# Report 3

У цій лабораторній я зробив багато роботи:

- налаштував Redis-кластер із трьох master-вузлів
  Робота виконана мовою програмування Rust.

Посилання на GitHub [тут](https://github.com/uandere/apz3).

https://github.com/uandere/apz3

Щоб запустити проект за допомогою Docker, просто напишіть:

```
docker compose up
```

Як результат, бачимо, що усі сервіси підняті успішно:


Серед них - три екземпляри logging-service (і, відповідно, три екземпляри redis-клієнта), facade-service та messages-service.

Також, є сервіс `lab3-redis-cluster-configure`. Єдиною його метою є правильне налаштування Redis-кластера і з'єднання усіх трьох Redis-вузлів між собою:

```
redis-cluster-configure-1  | Master[0] -> Slots 0 - 5460
redis-cluster-configure-1  | Master[1] -> Slots 5461 - 10922
redis-cluster-configure-1  | Master[2] -> Slots 10923 - 16383
redis-cluster-configure-1  | M: 9d94c7cc9efc9ae6b9993c7c128ae8cbb8bf6715 172.20.0.5:6379
redis-cluster-configure-1  |    slots:[0-5460] (5461 slots) master
redis-cluster-configure-1  | M: 910e84977cb1b20817d0ec85621e777c3f999840 172.20.0.6:6379
redis-cluster-configure-1  |    slots:[5461-10922] (5462 slots) master
redis-cluster-configure-1  | M: 2f28fd243d1b76c6399bc935a5282398ca55494c 172.20.0.4:6379
redis-cluster-configure-1  |    slots:[10923-16383] (5461 slots) master
redis-cluster-configure-1  | >>> Nodes configuration updated
redis-cluster-configure-1  | >>> Assign a different config epoch to each node
redis-cluster-configure-1  | >>> Sending CLUSTER MEET messages to join the cluster
redis-cluster-configure-1  | Waiting for the cluster to join
redis-cluster-node-1-1     | 1:M 06 Apr 2024 10:47:42.772 # Cluster state changed: ok
redis-cluster-node-0-1     | 1:M 06 Apr 2024 10:47:42.780 # Cluster state changed: ok
redis-cluster-node-2-1     | 1:M 06 Apr 2024 10:47:44.007 # Cluster state changed: ok
redis-cluster-configure-1  | ......
redis-cluster-configure-1  | >>> Performing Cluster Check (using node 172.20.0.5:6379)
redis-cluster-configure-1  | M: 9d94c7cc9efc9ae6b9993c7c128ae8cbb8bf6715 172.20.0.5:6379
redis-cluster-configure-1  |    slots:[0-5460] (5461 slots) master
redis-cluster-configure-1  | M: 910e84977cb1b20817d0ec85621e777c3f999840 172.20.0.6:6379
redis-cluster-configure-1  |    slots:[5461-10922] (5462 slots) master
redis-cluster-configure-1  | M: 2f28fd243d1b76c6399bc935a5282398ca55494c 172.20.0.4:6379
redis-cluster-configure-1  |    slots:[10923-16383] (5461 slots) master
redis-cluster-configure-1  | [OK] All nodes agree about slots configuration.
redis-cluster-configure-1  | >>> Check for open slots...
redis-cluster-configure-1  | >>> Check slots coverage...
redis-cluster-configure-1  | [OK] All 16384 slots covered.
redis-cluster-configure-1 exited with code 0
```

Як бачимо, всі три Redis-вузли розділили між собою порти порівну, і отримали повідомлення `REDIS MEET` щоб приєднатися до кластера. Потім, кожен із вузлів погодився на свій набір портів і приєднався до єдиного кластера. Сервіс `lab3-redis-cluster-configure`, натомість, закінчив роботу і повернув код 0.

Давайте спробуємо закинути 10 повідомлень за допомогою POST. Вони збережуться у Redis-кластері. Для цього я підготував спеціальний скрипт `send_messages.sh`:

```
sh send_messages.sh
```

Результат - uuid усіх повідомлень, як і повинно бути.

```
2aeef981-0104-4b24-9d1d-05ef588fb89c
33463ed5-45d4-4125-a410-681ad1e596f2
dd4269db-ed45-499d-bb92-386a0f992f5d
d184e473-948e-4448-a616-8dbb8185a101
7ec57fb6-15e3-4685-962b-058f20ecd951
bfc7d85e-8cda-43d7-a7ed-57d261398e17
1f751dfa-992f-4869-be43-9a2865ebbc39
a76f1a46-fc61-4ae8-9907-b644f9b0a4f2
b326af92-f246-4676-a115-b8a2c911c8b4
19785c55-5cab-46b9-b23b-e6ca8e65bf75
```

Поглянемо, чи отримав наші повідомлення logging-service:

```
logging-service-1-1       | [2024-04-06T10:48:22Z INFO  rocket::server] POST /log application/json:
logging-service-1-1       | [2024-04-06T10:48:22Z INFO  rocket::server::_] Matched: (log_message) POST /log application/json
logging-service-1-1       | [2024-04-06T10:48:23Z INFO  logging_service] Logging message: UUID: 2aeef981-0104-4b24-9d1d-05ef588fb89c, Msg: Hello World - Iteration 1
logging-service-1-1       | [2024-04-06T10:48:23Z INFO  rocket::server::_] Outcome: Success(200 OK)
logging-service-1-1       | [2024-04-06T10:48:23Z INFO  rocket::server::_] Response succeeded.
logging-service-3-1       | [2024-04-06T10:48:23Z INFO  rocket::server] POST /log application/json:
logging-service-3-1       | [2024-04-06T10:48:23Z INFO  rocket::server::_] Matched: (log_message) POST /log application/json
logging-service-3-1       | [2024-04-06T10:48:23Z INFO  logging_service] Logging message: UUID: 33463ed5-45d4-4125-a410-681ad1e596f2, Msg: Hello World - Iteration 2
logging-service-3-1       | [2024-04-06T10:48:23Z INFO  rocket::server::_] Outcome: Success(200 OK)
logging-service-3-1       | [2024-04-06T10:48:23Z INFO  rocket::server::_] Response succeeded.
logging-service-3-1       | [2024-04-06T10:48:23Z INFO  rocket::server] POST /log application/json:
logging-service-3-1       | [2024-04-06T10:48:23Z INFO  rocket::server::_] Matched: (log_message) POST /log application/json
logging-service-3-1       | [2024-04-06T10:48:24Z INFO  logging_service] Logging message: UUID: dd4269db-ed45-499d-bb92-386a0f992f5d, Msg: Hello World - Iteration 3
logging-service-3-1       | [2024-04-06T10:48:24Z INFO  rocket::server::_] Outcome: Success(200 OK)
logging-service-3-1       | [2024-04-06T10:48:24Z INFO  rocket::server::_] Response succeeded.
logging-service-1-1       | [2024-04-06T10:48:24Z INFO  rocket::server] POST /log application/json:
logging-service-1-1       | [2024-04-06T10:48:24Z INFO  rocket::server::_] Matched: (log_message) POST /log application/json
logging-service-1-1       | [2024-04-06T10:48:25Z INFO  logging_service] Logging message: UUID: d184e473-948e-4448-a616-8dbb8185a101, Msg: Hello World - Iteration 4
logging-service-1-1       | [2024-04-06T10:48:25Z INFO  rocket::server::_] Outcome: Success(200 OK)
logging-service-1-1       | [2024-04-06T10:48:25Z INFO  rocket::server::_] Response succeeded.
logging-service-3-1       | [2024-04-06T10:48:25Z INFO  rocket::server] POST /log application/json:
logging-service-3-1       | [2024-04-06T10:48:25Z INFO  rocket::server::_] Matched: (log_message) POST /log application/json
logging-service-3-1       | [2024-04-06T10:48:26Z INFO  logging_service] Logging message: UUID: 7ec57fb6-15e3-4685-962b-058f20ecd951, Msg: Hello World - Iteration 5
logging-service-3-1       | [2024-04-06T10:48:26Z INFO  rocket::server::_] Outcome: Success(200 OK)
logging-service-3-1       | [2024-04-06T10:48:26Z INFO  rocket::server::_] Response succeeded.
logging-service-2-1       | [2024-04-06T10:48:26Z INFO  rocket::server] POST /log application/json:
logging-service-2-1       | [2024-04-06T10:48:26Z INFO  rocket::server::_] Matched: (log_message) POST /log application/json
logging-service-2-1       | [2024-04-06T10:48:26Z INFO  logging_service] Logging message: UUID: bfc7d85e-8cda-43d7-a7ed-57d261398e17, Msg: Hello World - Iteration 6
logging-service-2-1       | [2024-04-06T10:48:26Z INFO  rocket::server::_] Outcome: Success(200 OK)
logging-service-2-1       | [2024-04-06T10:48:26Z INFO  rocket::server::_] Response succeeded.
logging-service-1-1       | [2024-04-06T10:48:26Z INFO  rocket::server] POST /log application/json:
logging-service-1-1       | [2024-04-06T10:48:26Z INFO  rocket::server::_] Matched: (log_message) POST /log application/json
logging-service-1-1       | [2024-04-06T10:48:26Z INFO  logging_service] Logging message: UUID: 1f751dfa-992f-4869-be43-9a2865ebbc39, Msg: Hello World - Iteration 7
logging-service-1-1       | [2024-04-06T10:48:26Z INFO  rocket::server::_] Outcome: Success(200 OK)
logging-service-1-1       | [2024-04-06T10:48:26Z INFO  rocket::server::_] Response succeeded.
```

Дійсно, усе працює.

Тепер спробуємо отримати накопичені повідомлення з нашого кластера:

```
curl http://localhost:8000/
```

Результат:


Як бачимо, всі повідомлення дійсно лежать у Redis-кластері.

Але що, якщо ми хочемо побачити розподіл ключів по конкретних вузлах? Не проблема, для цього в нас на системі повинен бути встановлений redis. Після цього, ми можемо приєднатися до конкретного кластеру і подивитися те, що на ньому лежить:

```
→  lab3 git:(main) × redis-cli -p 9079
127.0.0.1:9079> keys *
1) "dd4269db-ed45-499d-bb92-386a0f992f5d"
127.0.0.1:9079>
→  lab3 git:(main) × redis-cli -p 9080
127.0.0.1:9080> keys *
1) "19785c55-5cab-46b9-b23b-e6ca8e65bf75"
2) "a76f1a46-fc61-4ae8-9907-b644f9b0a4f2"
3) "b326af92-f246-4676-a115-b8a2c911c8b4"
4) "2aeef981-0104-4b24-9d1d-05ef588fb89c"
127.0.0.1:9080>
→  lab3 git:(main) × redis-cli -p 9081
127.0.0.1:9081> keys *
1) "d184e473-948e-4448-a616-8dbb8185a101"
2) "33463ed5-45d4-4125-a410-681ad1e596f2"
3) "1f751dfa-992f-4869-be43-9a2865ebbc39"
4) "7ec57fb6-15e3-4685-962b-058f20ecd951"
5) "bfc7d85e-8cda-43d7-a7ed-57d261398e17"
127.0.0.1:9081>
```

Тепер, що буде, якщо відключити кілька Redis-вузлів? Тут потрібно

зауважити, що Redis має дещо складнішу конфігурацію, аніж Hazelcast. По перше, нам потрібно щонайменше три master-nodes для того, щоб система могла залишатися у визначеному стані. Якщо ми просто спробуємо зупинити один із Redis-вузлів:

```
docker-compose stop redis-cluster-node-0
```

... то ми не зможемо отримати ніяку відповідь від сервера (цей запит триватиме вічно):

```
→  lab3 git:(main) ✗ curl http://localhost:8000/
^C
```

... і це при тому, що сервер наш запит отримує і опрацьовує, але він не в змозі отримати відповідь від Redis-кластера:

```
logging-service-2-1        | [2024-04-06T11:29:29Z INFO  rocket::server] GET /logs:
logging-service-2-1        | [2024-04-06T11:29:29Z INFO  rocket::server::_] Matched: (get_logs) GET /logs
```

Ми можемо зрозуміти, що кластер не в робочому стані по логах контейнерів:

```
redis-cluster-node-0-1     | 1:signal-handler (1712402941) Received SIGTERM scheduling shutdown...
redis-cluster-node-0-1     | 1:M 06 Apr 2024 11:29:01.452 # User requested shutdown...
redis-cluster-node-0-1     | 1:M 06 Apr 2024 11:29:01.452 * Calling fsync() on the AOF file.
redis-cluster-node-0-1     | 1:M 06 Apr 2024 11:29:01.452 # Redis is now ready to exit, bye bye...
redis-cluster-node-0-1 exited with code 0
redis-cluster-node-0-1 exited with code 0
redis-cluster-node-2-1     | 1:M 06 Apr 2024 11:29:08.087 * Marking node 910e84977cb1b20817d0ec85621e777c3f999840 as failing (quorum reached).
redis-cluster-node-2-1     | 1:M 06 Apr 2024 11:29:08.088 # Cluster state changed: fail
redis-cluster-node-1-1     | 1:M 06 Apr 2024 11:29:08.090 * Marking node 910e84977cb1b20817d0ec85621e777c3f999840 as failing (quorum reached).
redis-cluster-node-1-1     | 1:M 06 Apr 2024 11:29:08.090 # Cluster state changed: fail
```

Як бачимо, після завершення роботи `redis-cluster-node-0-1`, інші два його побратими не змогли змінити стан кластера, і тому перестали відповідати на запити. Хоча, кожен з них досі містить усі свої ключі, як і до того:

```
→  lab3 git:(main) ✗ redis-cli -p 9080
127.0.0.1:9080> keys *
1) "19785c55-5cab-46b9-b23b-e6ca8e65bf75"
2) "a76f1a46-fc61-4ae8-9907-b644f9b0a4f2"
3) "b326af92-f246-4676-a115-b8a2c911c8b4"
4) "2aeef981-0104-4b24-9d1d-05ef588fb89c"
```

Просто тепер ключі не синхронізуються через кластер.

Щоб це виправити, потрібно зробити наступну річ: в налаштуваннях кластеру (файл `path/redis_conf_folder/redis-cluster-create.sh`) треба змінити кількість реплік з 0 на 1:

```
redis-cli --cluster create $node_0_ip:6379 $node_1_ip:6379 $node_2_ip:6379 --cluster-replicas 0 --cluster-yes
```

```
redis-cli --cluster create $node_0_ip:6379 $node_1_ip:6379 $node_2_ip:6379 --cluster-replicas 1 --cluster-yes
```

А також додати три slave-вузли (по одному на кожен master-вузол) до нашого кластеру у `docker-compose.yml`:

```
redis-cluster-node-3:
  image: redis:6.0-alpine
  command: redis-server /usr/local/etc/redis/redis.conf
  ports:
    - "9082:6379"
  volumes:
    - ${PWD}/path/redis_conf_folder:/usr/local/etc/redis

redis-cluster-node-4:
  image: redis:6.0-alpine
  command: redis-server /usr/local/etc/redis/redis.conf
  ports:
    - "9083:6379"
  volumes:
    - ${PWD}/path/redis_conf_folder:/usr/local/etc/redis

redis-cluster-node-5:
  image: redis:6.0-alpine
  command: redis-server /usr/local/etc/redis/redis.conf
  ports:
    - "9084:6379"
  volumes:
    - ${PWD}/path/redis_conf_folder:/usr/local/etc/redis
```

Після цього, якщо ми видалимо один із вузлів:

```
docker-compose stop redis-cluster-node-0
```

... то кластер автоматично переконфігурується, і дані залишаться в цілісному стані:

```
redis-cluster-node-4-1     | 1:S 06 Apr 2024 14:33:59.156 * Connecting to MASTER 172.20.0.5:6379
redis-cluster-node-4-1     | 1:S 06 Apr 2024 14:33:59.156 * MASTER <-> REPLICA sync started
redis-cluster-node-3-1     | 1:S 06 Apr 2024 14:34:04.165 * Marking node 5449b47ddb54896124986049625749fb447d6d9 as failing (quorum reached).
redis-cluster-node-3-1     | 1:S 06 Apr 2024 14:34:04.165 # Cluster state changed: fail
redis-cluster-node-1-1     | 1:M 06 Apr 2024 14:34:04.167 * FAIL message received from dc0660ea739b74425762af3c98d947e5204ca18c about 5449b47ddb54896124986049625749fb447d6d9
redis-cluster-node-1-1     | 1:M 06 Apr 2024 14:34:04.167 # Cluster state changed: fail
redis-cluster-node-4-1     | 1:S 06 Apr 2024 14:34:04.167 * FAIL message received from dc0660ea739b74425762af3c98d947e5204ca18c about 5449b47ddb54896124986049625749fb447d6d9
redis-cluster-node-2-1     | 1:M 06 Apr 2024 14:34:04.169 * FAIL message received from dc0660ea739b74425762af3c98d947e5204ca18c about 5449b47ddb54896124986049625749fb447d6d9
redis-cluster-node-4-1     | 1:S 06 Apr 2024 14:34:04.167 # Cluster state changed: fail
redis-cluster-node-2-1     | 1:M 06 Apr 2024 14:34:04.169 # Cluster state changed: fail
redis-cluster-node-5-1     | 1:S 06 Apr 2024 14:34:04.168 * FAIL message received from dc0660ea739b74425762af3c98d947e5204ca18c about 5449b47ddb54896124986049625749fb447d6d9
redis-cluster-node-5-1     | 1:S 06 Apr 2024 14:34:04.168 # Cluster state changed: fail
redis-cluster-node-4-1     | 1:S 06 Apr 2024 14:34:04.269 # Start of election delayed for 505 milliseconds (rank #0, offset 809).
redis-cluster-node-4-1     | 1:S 06 Apr 2024 14:34:04.782 # Starting a failover election for epoch 7.
redis-cluster-node-1-1     | 1:M 06 Apr 2024 14:34:04.785 # Failover auth granted to 44497efe629e78931b51f075316126ef5fd21f23 for epoch 7
redis-cluster-node-2-1     | 1:M 06 Apr 2024 14:34:04.786 # Failover auth granted to 44497efe629e78931b51f075316126ef5fd21f23 for epoch 7
redis-cluster-node-4-1     | 1:S 06 Apr 2024 14:34:04.788 # Failover election won: I'm the new master.
redis-cluster-node-4-1     | 1:S 06 Apr 2024 14:34:04.788 # configEpoch set to 7 after successful failover
redis-cluster-node-4-1     | 1:M 06 Apr 2024 14:34:04.788 * Discarding previously cached master state.
redis-cluster-node-4-1     | 1:M 06 Apr 2024 14:34:04.788 # Setting secondary replication ID to e8f973f8cdaef87460e0e0aa39cf8020822468d4, valid up to offset: 810. New replicati
on ID is 5232f8d3ef56c3ad524905031bde6a2c6ed056c8
redis-cluster-node-4-1     | 1:M 06 Apr 2024 14:34:04.789 # Cluster state changed: ok
redis-cluster-node-3-1     | 1:S 06 Apr 2024 14:34:04.790 # Cluster state changed: ok
redis-cluster-node-2-1     | 1:M 06 Apr 2024 14:34:04.790 # Cluster state changed: ok
redis-cluster-node-1-1     | 1:M 06 Apr 2024 14:34:04.790 # Cluster state changed: ok
redis-cluster-node-5-1     | 1:S 06 Apr 2024 14:34:04.790 # Cluster state changed: ok
```

Перевіримо присутність усіх даних:

```
➜  lab3 git:(main) ✗ curl http://localhost:8000/
[{"uuid":"184665c0-cb66-4e9b-b946-a49013b3856c","msg":"Hello World - Iteration 4"},{"uuid":"634d27aa-2eda-4050-b324-28a43a605c19","msg":"Hello World - Iteration 3"},{"uuid":"cb
818670-6c02-42cc-a5a0-4b21a6d66d23","msg":"Hello World - Iteration 2"},{"uuid":"997474e9-9acc-47d3-b8cb-9a9b76dd65de","msg":"Hello World - Iteration 10"},{"uuid":"3696882b-c80e
-4295-89ef-c560fae20323","msg":"Hello World - Iteration 8"},{"uuid":"6807c33e-c309-49ad-98e7-57e2bdbd0fcb","msg":"Hello World - Iteration 6"},{"uuid":"7793a385-72cf-43c8-86ac-3
fddd97b2a45","msg":"Hello World - Iteration 5"},{"uuid":"bcfbd246-130a-40f3-9ea9-53de7a469fbb","msg":"Hello World - Iteration 7"},{"uuid":"d6b281de-4d6f-424e-a7a2-8c1c80fe1da0"
,"msg":"Hello World - Iteration 1"},{"uuid":"1bf18269-80cc-4382-a049-985c92d30eb2","msg":"Hello World - Iteration 9"}]%
```

Ось розподіл ключів (разом із реплікацією) по окремих нодах:

```
➜  lab3 git:(main) ✗ redis-cli -p 9080
127.0.0.1:9080> keys *
1) "184665c0-cb66-4e9b-b946-a49013b3856c"
2) "634d27aa-2eda-4050-b324-28a43a605c19"
3) "cb818670-6c02-42cc-a5a0-4b21a6d66d23"
127.0.0.1:9080>
➜  lab3 git:(main) ✗ redis-cli -p 9081
127.0.0.1:9081> keys *
1) "997474e9-9acc-47d3-b8cb-9a9b76dd65de"
2) "3696882b-c80e-4295-89ef-c560fae20323"
3) "6807c33e-c309-49ad-98e7-57e2bdbd0fcb"
127.0.0.1:9081>
➜  lab3 git:(main) ✗ redis-cli -p 9082
127.0.0.1:9082> keys *
1) "3696882b-c80e-4295-89ef-c560fae20323"
2) "997474e9-9acc-47d3-b8cb-9a9b76dd65de"
3) "6807c33e-c309-49ad-98e7-57e2bdbd0fcb"
127.0.0.1:9082>
➜  lab3 git:(main) ✗ redis-cli -p 9083
127.0.0.1:9083> keys *
1) "7793a385-72cf-43c8-86ac-3fddd97b2a45"
2) "bcfbd246-130a-40f3-9ea9-53de7a469fbb"
3) "d6b281de-4d6f-424e-a7a2-8c1c80fe1da0"
4) "1bf18269-80cc-4382-a049-985c92d30eb2"
127.0.0.1:9083>
➜  lab3 git:(main) ✗ redis-cli -p 9084
127.0.0.1:9084> keys *
1) "cb818670-6c02-42cc-a5a0-4b21a6d66d23"
2) "184665c0-cb66-4e9b-b946-a49013b3856c"
3) "634d27aa-2eda-4050-b324-28a43a605c19"
127.0.0.1:9084>
```

Що ж буде, якщо відключити кілька вузлів відразу? Відбудеться наступне:

якщо відключати лише master-вузли, то нічого поганого не відбудеться, за умови існування реплікації. Якщо ж відключати і master-вузли, і їх репліки та робити це одночасно, то частина даних буде втрачена, і кластер не зможе відповідати на запити через наявність inconsistency в системі.