

# RTGS 프로토타입 시스템 PoC 테스트 4차 계획 [로컬 개발환경 구성]

2025. 7

|                                   |    |
|-----------------------------------|----|
| I. 개요 .....                       | 1  |
| II. 시스템 구성 .....                  | 2  |
| III. 기본 동작 테스트 결과 .....           | 5  |
| IV. 향후 계획 .....                   | 9  |
| <br>(참고) 솔루션 및 응용프로그램 설치 내역 ..... | 10 |
| 1. OS .....                       | 10 |
| 2. VMWare .....                   | 12 |
| 3. JDK .....                      | 13 |
| 4. 개발도구 .....                     | 14 |
| 5. 배포도구 .....                     | 15 |
| 6. 네트워크 설정 .....                  | 16 |
| 7. 데이터베이스 .....                   | 19 |
| 8. 이벤트 브로커 .....                  | 27 |
| 9. 응용 프로그램 .....                  | 31 |

IT전략국 금융IT부 RTGS시스템팀

- IT전략국은 금융결제국과 함께 실시간충액결제 방식의 신속자금이체를 24×365 연중무휴로 제공하는 RTGS 시스템 구축을 계획\*

\* 「RTGS 방식 신속자금이체시스템 구축 종합계획」(결제정책팀-1169, 2023.12.28, 부총재보)  
 「소액RTGS시스템 구축을 위한 IT부문 기본계획」(결제시스템팀-3, 2024.1.2, 부총재보)  
 「소액 RTGS 사업 주요 추진 경과 및 향후 계획」(결제정책팀-479, 2024.6.10., 부총재보)  
 「RTGS시스템팀 주요 추진 내용 및 향후 계획」(RTGS시스템팀-35, 2025.3.6, 부총재보)

- 국내외 조사 사례를 바탕으로 IT전략국은 향후 RTGS 시스템 구축 시 필요한 IT요소기술과 아키텍처를 설계(2023.8월~2024.7월)\*

\* 「소액RTGS시스템 구축을 위한 기본 아키텍처」(RTGS시스템팀-62, 2024.5.20, 부장)  
 「소액RTGS시스템 구축 관련 기술검증 방안」(RTGS시스템팀-69, 2024.5.31, 부장)  
 「소액RTGS시스템 기술검증을 위한 프로토타입시스템 구축계획」(RTGS시스템팀-78, 2024.7.1, 부장)

- 이를 토대로 1~3차 PoC 테스트\*를 수행하며 2개 센터간 Active-Active 운영에 대해 성능·안정 및 확장성 측면에서 실현가능성을 검증, 향후 프로젝트에 대한 상세 요건 및 시사점을 도출(2024.8월~2025.6월)

\* 「소액RTGS시스템 기술검증을 위한 프로토타입 핵심프로세스 개발보고」(RTGS시스템팀-99, 2024.8.23, 부장)  
 「소액RTGS시스템 프로토타입 1차 테스트 완료 보고」(RTGS시스템팀-9, 2025.1.24., 부장)  
 「사업결의(RTGS 프로토타입 시스템에 대한 PoC테스트)」(RTGS시스템팀-85, 2025.4.30., 팀장)  
 「RTGS 프로토타입 시스템 PoC 테스트 3차 결과 보고」(RTGS시스템팀-143, 2025.7.29., 부장)

- 2026년부터 실시 예정인 BMT 사업을 앞두고 3개 센터 간 동시 운영과 당행 네트워크 적용 가능성 등 확인을 위해 추가 검증이 필요

⇒ 2025년 하반기에 로컬 환경에서 RTGS 프로토타입 시스템에 대한 4차 PoC 테스트를 수행하고자 당행 인터넷망에 개발 환경을 구성

「RTGS시스템 프로토타입」 PoC 테스트 추진

| 구분                  | 2023  |   |    |    |    | 2024 |   |   |   |   |         |   |   |   |    |    |    | 2025 |   |   |   |   |   |   |   |   |    |    |    |
|---------------------|-------|---|----|----|----|------|---|---|---|---|---------|---|---|---|----|----|----|------|---|---|---|---|---|---|---|---|----|----|----|
|                     | 8     | 9 | 10 | 11 | 12 | 1    | 2 | 3 | 4 | 5 | 6       | 7 | 8 | 9 | 10 | 11 | 12 | 1    | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|                     | 분석/설계 |   |    |    |    | 개발   |   |   |   |   | PoC 테스트 |   |   |   |    |    |    |      |   |   |   |   |   |   |   |   |    |    |    |
| IT요소기술 분석           |       |   |    |    |    |      |   |   |   |   |         |   |   |   |    |    |    |      |   |   |   |   |   |   |   |   |    |    |    |
| 국내외 사례조사            |       |   |    |    |    |      |   |   |   |   |         |   |   |   |    |    |    |      |   |   |   |   |   |   |   |   |    |    |    |
| 핵심 아키텍처 설계          |       |   |    |    |    |      |   |   |   |   |         |   |   |   |    |    |    |      |   |   |   |   |   |   |   |   |    |    |    |
| 프로토타입 시스템 개발        |       |   |    |    |    |      |   |   |   |   |         |   |   |   |    |    |    |      |   |   |   |   |   |   |   |   |    |    |    |
| 프로토타입 PoC 테스트(1~2차) |       |   |    |    |    |      |   |   |   |   |         |   |   |   |    |    |    |      |   |   |   |   |   |   |   |   |    |    |    |
| 프로토타입 PoC 테스트(3차)   |       |   |    |    |    |      |   |   |   |   |         |   |   |   |    |    |    |      |   |   |   |   |   |   |   |   |    |    |    |
| 프로토타입 PoC 테스트(4차)   |       |   |    |    |    |      |   |   |   |   |         |   |   |   |    |    |    |      |   |   |   |   |   |   |   |   |    |    |    |

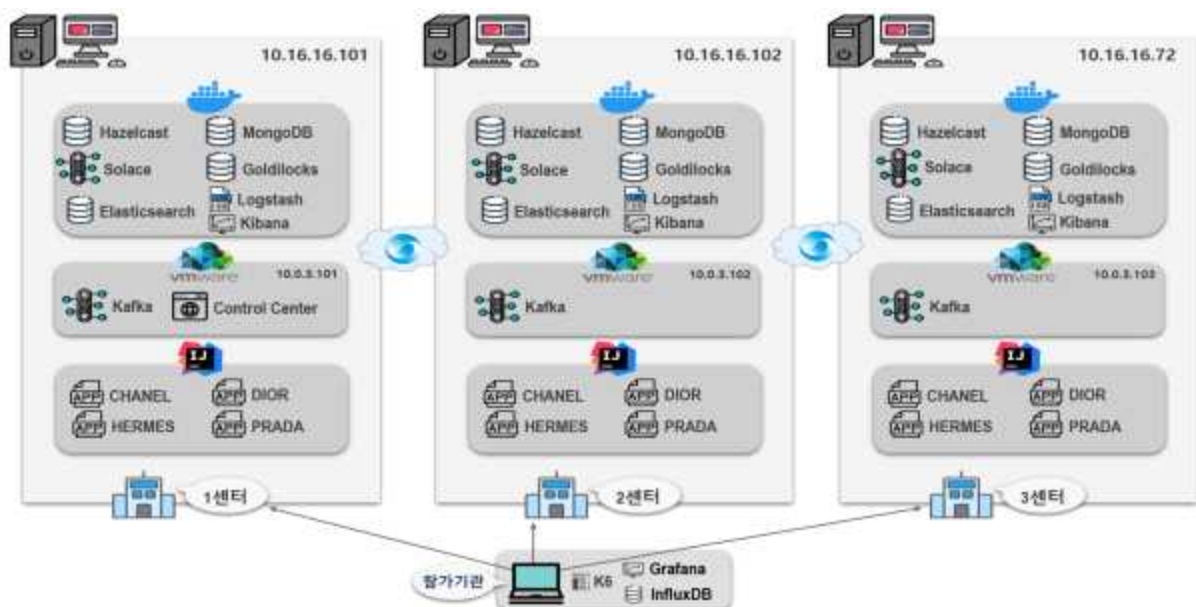
※ 프로토타입 PoC 테스트(4차)는 신규 프로젝트 추진과 관련하여 일정 및 내용이 조정될 수 있음

## II

## 시스템 구성

- RTGS 프로토타입 시스템을 당행 내부의 로컬 서버에서 인터넷을 통해 테스트 해보기 위해 별관 6층 사무실의 고성능 PC 3대를 구성하고, 각 PC를 하나의 센터로 취급
  - 각 PC에는 PoC 3차 테스트에서 구성한 서버 환경을 그대로 구성하되 로컬 환경의 고성능 PC는 클라우드에 비해 자원이 제한적이므로 모든 구성요소는 최소 사양으로 설치
  - 참가기관에서 각 센터로 요청을 보내는 것을 가정하기 위해 테스트 도구 (K6\*)는 노트북에 별도로 설치하여 실행
    - \* K6 : 웹 서비스의 성능과 안정성을 검증하기 위해 가상의 사용자가 실제처럼 접근하여 부하를 발생시키는 오픈소스 테스트 도구
  - 데이터베이스(Hazelcast, MongoDB, Goldilocks, ELK\*) 및 이벤트 브로커 Solace는 도커\*\* 컨테이너로 배포하며 Kafka는 가상머신(VM)으로 구성
    - \* ELK : Elasticsearch, Logstash, Kibana 세가지 오픈소스 도구로 로그 수집, 분석, 시각화를 위한 기술 스택
    - \*\* 도커(Docker) : 어플리케이션을 컨테이너라는 격리된 환경에서 실행할 수 있도록 해주는 오픈소스 도구
  - 자체 개발한 4가지 응용 프로그램인 CHANEL, DIOR, HERMES, PRADA의 경우 JVM(Java Virtual Machine)의 프로세스로 구성

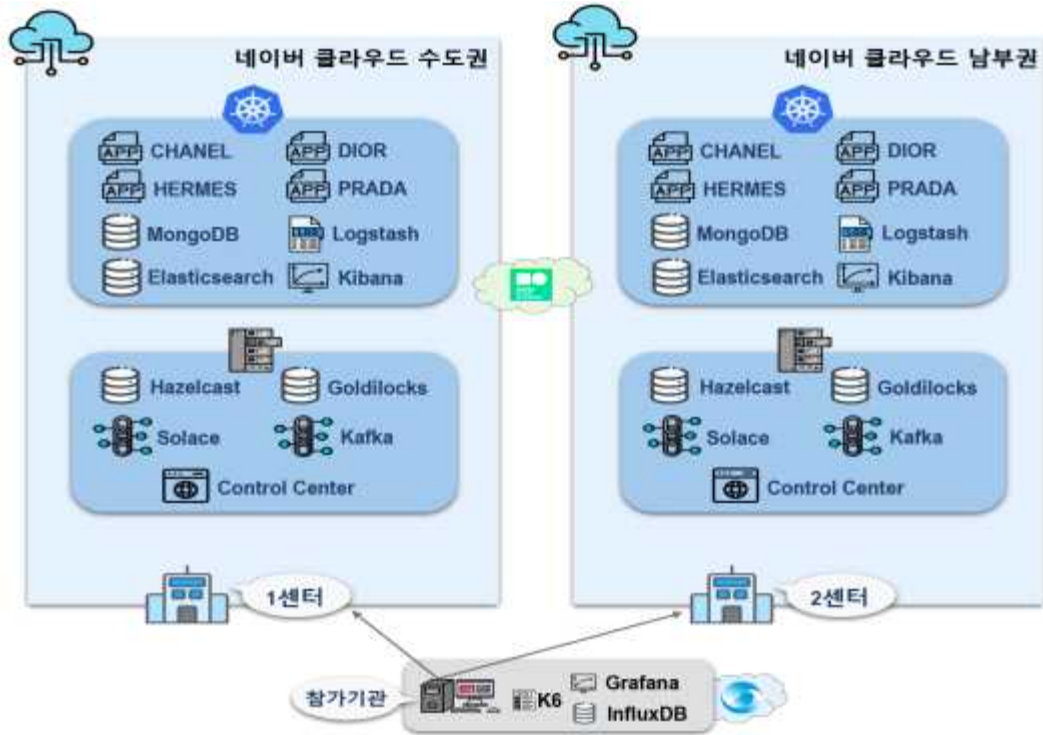
**로컬 환경 RTGS 프로토타입 시스템 구성도**



- 로컬 환경에서 수행할 PoC 4차 테스트는 클라우드 환경에서 수행한 PoC 3차 테스트와는 센터 구성 및 사용 도구 등에서 차이가 있음

※ 참고

**PoC 3차 테스트(클라우드 환경) RTGS 프로토타입 시스템 구성도**



**PoC 3차 테스트와 4차 테스트 비교**

| 구분             | PoC 3차 테스트                         | PoC 4차 테스트                 |
|----------------|------------------------------------|----------------------------|
| 실행 환경          | 클라우드(NCP, 네이버 공공 클라우드)             | 로컬(당행 고성능 PC)              |
| 센터 개수          | 2개 센터                              | 3개 센터                      |
| 배포 도구          | 쿠버네티스                              | 도커                         |
| 네트워크           | 네이버 클라우드망                          | 당행 인터넷망                    |
| Bastion 서버*    | 필요(외부에서 네이버 클라우드망으로 접근이 필요)        | 필요없음                       |
| K6 요청          | 당행 고성능 PC에서 네이버 클라우드망 서버로 요청       | 당행 노트북에서 고성능 PC로 요청        |
| CHANEL<br>노드 수 | 3개(자원 제약이 없으므로 노드 수를 증가시켜 병렬처리 가능) | 1개(자원 제약을 받으므로 최소 사양으로 설치) |
| CPU 코어수<br>합계  | 109 코어(1센터), 97코어(2센터)             | 25코어(1센터, 2센터, 3센터)        |
| 메모리 합계         | 380GB(1센터), 340GB(2센터)             | 62GB(1센터, 2센터, 3센터)        |

\* Bastion 서버 : 외부에서 내부 네트워크에 접근할 수 있는 게이트웨이 역할을 하는 서버

## 로컬 환경 RTGS 프로토타입 솔루션 설치 사양

| 리전                            |                    | 테스트 케이스            |                   | 서비스                                | 3차 PoC (클라우드) |         | 4차 PoC (로컬) |         |
|-------------------------------|--------------------|--------------------|-------------------|------------------------------------|---------------|---------|-------------|---------|
|                               |                    |                    |                   |                                    | vCPU(core)    | 메모리(GB) | CPU(core)   | 메모리(GB) |
| PC-1<br>(1센터)<br>10.16.16.101 | 노드                 | ㉠-1, ㉡-1           | mongodb           | 2                                  | 8             | 2       | 4           |         |
|                               |                    | ㉠-1, ㉠-2, ㉡-1, ㉡-2 | chanel*           | 32                                 | 128           | 1       | 2           |         |
|                               |                    |                    | dior              | 2                                  | 4             | 1       | 2           |         |
|                               |                    |                    | hermes            | 2                                  | 4             | 1       | 2           |         |
|                               |                    |                    | prada             | 2                                  | 4             | 1       | 2           |         |
|                               |                    |                    | elasticsearch     | 1                                  | 4             | 1       | 2           |         |
|                               |                    | ㉠-1, ㉠-2, ㉡-1, ㉡-2 | logstash          | 1                                  | 2             | 1       | 2           |         |
|                               |                    |                    | kibana            | 1                                  | 2             | 1       | 2           |         |
|                               |                    |                    | rtgs-solace01     | 16                                 | 64            | 2       | 8           |         |
|                               |                    | ㉡-1, ㉡-2           | solace-bestion    | 2                                  | 8             |         |             |         |
|                               |                    |                    | gd-bastion        | 2                                  | 8             |         |             |         |
|                               | VM**<br>10.0.3.101 | ㉠-2, ㉡-2           | goldilocks-c01    | 8                                  | 16            | 2       | 8           |         |
|                               |                    |                    | goldilocks-c02    | 8                                  | 16            | 2       | 8           |         |
|                               |                    |                    | hazelcast-bastion | 2                                  | 4             |         |             |         |
|                               |                    |                    | rtgs-hazelcast01* | 8                                  | 32            | 2       | 4           |         |
| tg-br-001                     |                    |                    | 4                 | 16                                 | 8             | 16      |             |         |
| tg-br-002                     |                    |                    | 4                 | 16                                 |               |         |             |         |
| tg-br-003                     |                    |                    | 4                 | 16                                 |               |         |             |         |
| tg-kc-001                     | 2                  | 8                  |                   |                                    |               |         |             |         |
| ㉠-1, ㉡-1, ㉢                   | tg-cc-001          | 4                  | 16                |                                    |               |         |             |         |
|                               | tg-bastion         | 2                  | 4                 |                                    |               |         |             |         |
| 합계                            |                    |                    |                   |                                    | 109           | 380     | 25          | 62      |
| PC-2<br>(2센터)<br>10.16.16.102 | 노드                 | ㉠-1, 2-1           | mongodb           | 2                                  | 8             | 2       | 4           |         |
|                               |                    | ㉠-1, ㉠-2, ㉡-1, ㉡-2 | chanel*           | 32                                 | 128           | 1       | 2           |         |
|                               |                    |                    | dior              | 2                                  | 4             | 1       | 2           |         |
|                               |                    |                    | hermes            | 2                                  | 4             | 1       | 2           |         |
|                               |                    |                    | prada             | 2                                  | 4             | 1       | 2           |         |
|                               |                    |                    | elasticsearch     | 1                                  | 4             | 1       | 2           |         |
|                               |                    | ㉠-1, ㉠-2, ㉡-1, ㉡-2 | logstash          | 1                                  | 2             | 1       | 2           |         |
|                               |                    |                    | kibana            | 1                                  | 2             | 1       | 2           |         |
|                               |                    |                    | rtgs-solace11     | 16                                 | 64            | 2       | 8           |         |
|                               |                    | ㉡-1, ㉡-2           | goldilocks-s01    | 8                                  | 16            | 2       | 8           |         |
|                               |                    |                    | goldilocks-s02    | 8                                  | 16            | 2       | 8           |         |
|                               | VM*<br>10.0.3.102  | ㉠-1, ㉡-1, ㉢        | rtgs-hazelcast11* | 8                                  | 32            | 2       | 4           |         |
|                               |                    |                    | tg-br-004         | 4                                  | 16            | 8       | 16          |         |
|                               |                    |                    | tg-br-005         | 4                                  | 16            |         |             |         |
|                               |                    |                    | tg-br-006         | 4                                  | 16            |         |             |         |
| tg-kc-002                     |                    |                    | 2                 | 8                                  |               |         |             |         |
| 합계                            |                    |                    |                   |                                    | 97            | 340     | 25          | 62      |
| PC-3<br>(3센터)<br>10.16.16.72  | 노드                 | ㉠-1, ㉡-1           | mongodb           | PoC 3차 테스트는 2개<br>센터 구성으로<br>진행되었음 |               | 2       | 4           |         |
|                               |                    | ㉠-1, ㉠-2, ㉡-1, ㉡-2 | chanel            |                                    |               | 1       | 2           |         |
|                               |                    |                    | dior              |                                    |               | 1       | 2           |         |
|                               |                    |                    | hermes            |                                    |               | 1       | 2           |         |
|                               |                    |                    | prada             |                                    |               | 1       | 2           |         |
|                               |                    |                    | elasticsearch     |                                    |               | 1       | 2           |         |
|                               |                    | ㉠-1, ㉠-2, ㉡-1, ㉡-2 | logstash          |                                    |               | 1       | 2           |         |
|                               |                    |                    | kibana            |                                    |               | 1       | 2           |         |
|                               |                    |                    | rtgs-solace21     |                                    |               | 2       | 8           |         |
|                               |                    | ㉡-1, ㉡-2           | goldilocks-k01    |                                    |               | 2       | 8           |         |
|                               |                    |                    | goldilocks-k02    |                                    |               | 2       | 8           |         |
|                               | VM**<br>10.0.3.103 | ㉠-1, ㉡-1, ㉢        | rtgs-hazelcast21  |                                    |               | 2       | 4           |         |
|                               |                    |                    | tg-br-007         |                                    |               | 4       | 16          |         |
|                               |                    |                    | tg-br-008         |                                    |               | 4       | 16          |         |
|                               |                    |                    | tg-br-009         |                                    |               | 4       | 16          |         |
| tg-kc-003                     |                    |                    | 2                 |                                    |               | 8       |             |         |
| 합계                            |                    |                    |                   |                                    |               |         | 25          | 62      |

\* 클라우드 환경에서는 병렬 처리를 위해 노드 3개로 구성되었음

\*\* 현재 Kafka Confluent의 경우 모든 센터 같은 사양의 VM으로 구성되어 있으므로 vCPU, 메모리 동일

### 고성능 PC 사양

| 구분       | IP주소         | OS         | CPU     | Memory |
|----------|--------------|------------|---------|--------|
| 고성능 PC-1 | 10.16.16.101 | Windows 11 | i9 24코어 | 128GB  |
| 고성능 PC-2 | 10.16.16.102 |            |         |        |
| 고성능 PC-3 | 10.16.16.72  |            |         |        |



## MongoDB 조회 화면

[illegible]

### 3개 센터 Hazelcast 계좌 조회 화면

### 3개 센터 MongoDB 계좌 조회 화면

The image displays three screenshots of a network traffic analysis tool (Wireshark) showing IP addresses, with annotations indicating the source of the traffic. The top-left screenshot is labeled "1센터 ip" (1st Center IP), the top-right is labeled "2센터 ip" (2nd Center IP), and the bottom is labeled "3센터 ip" (3rd Center IP). A central green oval contains the text "3개 센터 계좌 일치" (3 centers account match), with arrows pointing from the three screenshots to it, suggesting that the IP addresses from these three sources are being compared for account matching.

- o 테스트 케이스 1-2 (Kafka Confluent + Goldilocks + Goldilocks) 수행 결과, 별도의 노트북에서 부하발생기(K6)로 요청한 51건에 대해 3개 센터 모두 정상적으로 기능처리 확인 완료

### K6 실행 화면

```

execution: local
script: ./localtest.js
output:

scenarios: (100.00%) 1 scenario, 1000 max VUs, 31s max duration (incl. graceful stop):
  * constant_rps: 50.00 iterations/s for 1s (maxVUs: 500-1000, gracefulStop: 30s)

TOTAL RESULTS
checks_total          51      0.5(172)%
checks_succeeded      100.00% 51 out of 51
checks_failed         0.00%  0 out of 51
✓ is status 200

HTTP
http_req_duration     avg=10.5ms  p(90)=13.22ms p(95)=13.75ms min=7.34ms max=16.67ms
                    { expected_response:true }
http_req_failed       0.00%  0 out of 51
http_reqs             51      0.5(172)%

EXECUTION
iteration_duration     avg=11.4ms  p(90)=13.93ms p(95)=15.48ms min=7.67ms max=26.71ms
iterations            51      0.5(172)%
vus                   0      0.0(0)%
vus_max               500    0.0(0)%

NETWORK
data_received         87 kB  0.0 MB/s
data_sent             233 kB  0.0 MB/s

running (01.0s), 0000/0500 VUs, 51 complete and 0 interrupted iterations

```

### Goldilocks 인메모리 DB 조회 화면

|                          |                |              |                                        |                             |       |             |  |  |
|--------------------------|----------------|--------------|----------------------------------------|-----------------------------|-------|-------------|--|--|
| <input type="checkbox"/> | goldilocks-bok |              |                                        |                             | 5.72% | 3 hours ago |  |  |
| <input type="checkbox"/> | goldilocks-c02 | 6c7de1f5848b | <a href="#">goldilocks-bok.22c-1_5</a> | <a href="#">22582.22581</a> | 2.75% | 3 hours ago |  |  |
| <input type="checkbox"/> | goldilocks-c01 | 4f6908dfe1f4 | <a href="#">goldilocks-bok.22c-1_5</a> | <a href="#">22581.22581</a> | 2.97% | 3 hours ago |  |  |

Showing 14 Items

최종 상태ACCC 확인

|           |                                       |      |     |      |      |            |            |                            |                   |        |   |
|-----------|---------------------------------------|------|-----|------|------|------------|------------|----------------------------|-------------------|--------|---|
| 42_325489 | 202542169736139872216464285987791328  | ACCC | 63  | 8778 | 1328 | 999999937  | 1000000184 | 2025-07-22 16:46:41.969625 | 2025-07-22 16:46: | gc-c01 | X |
| 42_299797 | 2025992579944737072216464281912908588 | ACCC | 15  | 1268 | 8088 | 999999961  | 9999999575 | 2025-07-22 16:46:41.785263 | 2025-07-22 16:46: | gc-c02 | X |
| 42_228802 | 2025662165825688672216464197912888588 | ACCC | 35  | 1268 | 8588 | 999999976  | 9999999660 | 2025-07-22 16:46:41.889359 | 2025-07-22 16:46: |        |   |
| 42_174885 | 2025984744238857072216464195919478870 | ACCC | 194 | 1047 | 8878 | 999999980  | 1000000241 | 2025-07-22 16:46:41.644186 | 2025-07-22 16:46: |        |   |
| 42_143878 | 2025489447867711872216464193993221264 | ACCC | 11  | 9322 | 1268 | 9999999948 | 1000000011 | 2025-07-22 16:46:41.840826 | 2025-07-22 16:46: |        |   |
| 42_114347 | 2025911374862550872216464182218518870 | ACCC | 47  | 1851 | 8878 | 999999953  | 1000000847 | 2025-07-22 16:46:41.524198 | 2025-07-22 16:46: |        |   |
| 41_916249 | 2025358444361610672216464183993221328 | ACCC | 41  | 9322 | 1328 | 999999959  | 1000000041 | 2025-07-22 16:46:41.525285 | 2025-07-22 16:46: |        |   |
| 41_942435 | 202552836896429487221646418798588118  | ACCC | 75  | 8588 | 1184 | 999999925  | 1000000075 | 2025-07-22 16:46:41.566426 | 2025-07-22 16:46: |        |   |
| 42_820381 | 2025426891451108872216464239985261314 | ACCC | 165 | 8528 | 1314 | 1000000024 | 1000000191 | 2025-07-22 16:46:42.312812 | 2025-07-22 16:46: |        |   |
| 42_708238 | 202537585795438867221646424198078218  | ACCC | 76  | 8678 | 1211 | 1000000058 | 1000000269 | 2025-07-22 16:46:42.331293 | 2025-07-22 16:46: |        |   |

RAM 14.95 GB CPU 0.44% Disk 27.21 GB used (limit 1006.85 GB)

Terminal New version available

## Goldilocks 원장 DB 조회 화면

[illegible]

### 3개 센터 Goldilocks 인메모리 DB 계좌 조회 화면

The screenshot displays the AWS IAM console's 'Groups' page. At the top, there are three tabs: 'group-admins', 'group-devs', and 'group-prod'. The 'group-devs' tab is selected. Below the tabs, there are two sections: 'Groups' and 'Users'. The 'Groups' section lists three groups: 'group-admins', 'group-devs', and 'group-prod'. The 'group-devs' group is highlighted. The 'Users' section lists three users: 'user-admins', 'user-devs', and 'user-prod'. The 'user-devs' user is highlighted. Annotations include: '1센터 DB' pointing to the 'group-devs' group, '2센터 DB' pointing to the 'group-prod' group, '3개 센터 계좌 일치' pointing to the 'user-devs' user, and '3센터 DB' pointing to the 'user-prod' user.

### 3개 센터 Goldilocks 원장 DB 계좌 조회 화면

## IV

## 향후 계획

- 현재 RTGS 프로토타입 시스템의 아키텍처는 제약사항을 개선하여 **2025년 하반기 PoC 추가 테스트를 진행하고 있으며**, PoC 사업 결과를 바탕으로 **2026 ~ 2027년 BMT\* 사업을 추진할 계획**

\* BMT는 벤치마크 테스트(BenchMark Test)로 하드웨어 인프라에 소요되는 성능이 어느 정도인지 비교평가하기 위해 수행하는 테스트

### < 2025년 하반기 PoC 테스트(4차) 계획\* >

\* 2025.하반기 계획은 신규 프로젝트 추진과 관련하여 일정 및 내용이 조정될 수 있음

- 로컬 환경에서 테스트 케이스 [2] 유형에 대한 동작 테스트 진행 및 모든 테스트 케이스에 대한 **성능 및 복원력 테스트** 실시
- 이벤트 브로커 **Solace**에 대한 **3센터 추가 구성(Avative-Active-Stanby)** 논의 및 추가 테스트를 진행
- ISO20022 전문의 BMI 생성 정확도 개선, 예외 처리 및 로그 분석 개선, 테스트 도구의 적용 변수 다양화 등 각 **센터 간 데이터 정합성 확인용 로직 보완 및 추가**

### 2025년 RTGS시스템팀 추진 계획 및 일정

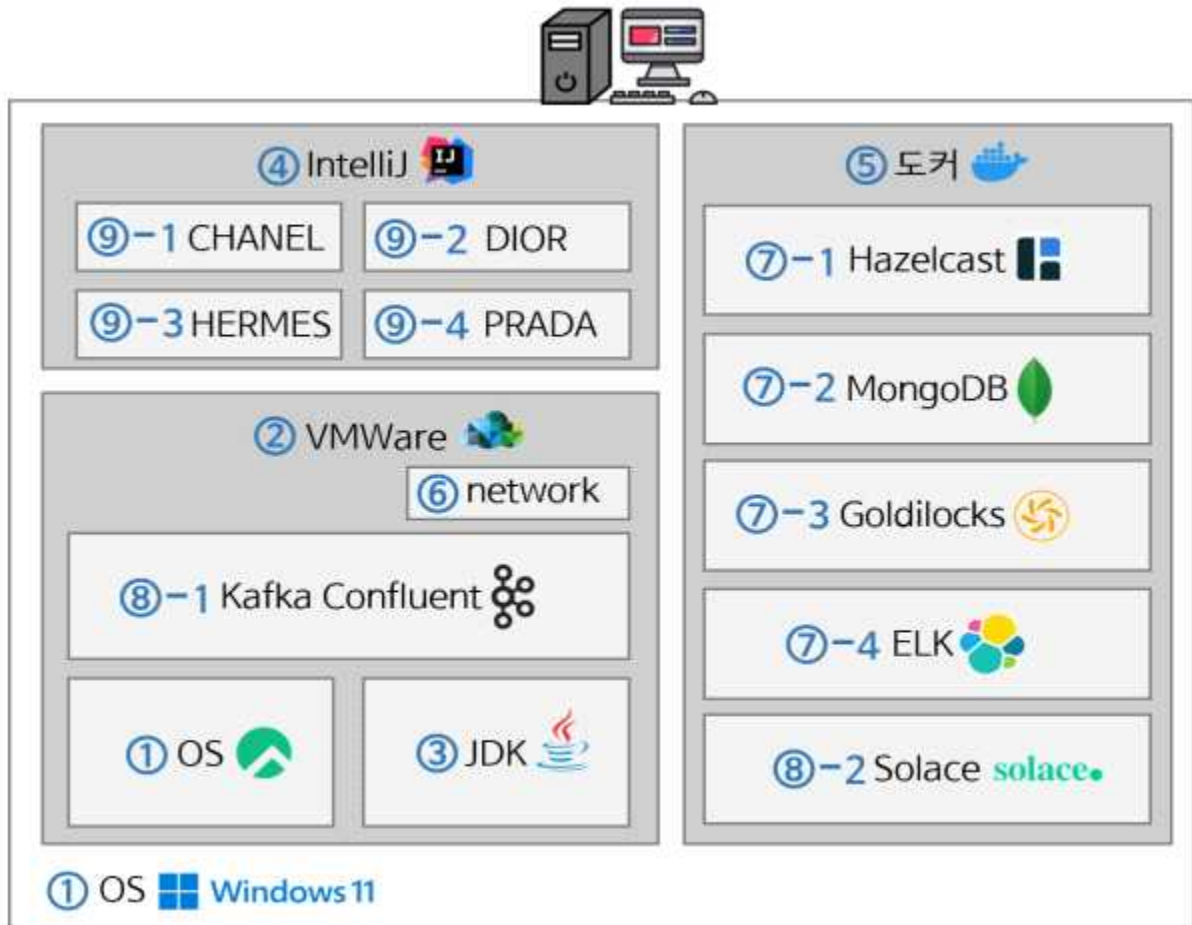
|             | 구분            | 1월 | 2월 | 3월 | 4월 | 5월 | 6월 | 7월 | 8월 | 9월 | 10월 | 11월 | 12월 |
|-------------|---------------|----|----|----|----|----|----|----|----|----|-----|-----|-----|
| PoC 테스트 2차* | 테스트 실시        |    |    |    |    |    |    |    |    |    |     |     |     |
|             | 프로토타입 시스템 보완  |    |    |    |    |    |    |    |    |    |     |     |     |
| PoC 테스트 3차  | 테스트 요구사항 정의   |    |    |    |    |    |    |    |    |    |     |     |     |
|             | 사업결의 및 계약체결   |    |    |    |    |    |    |    |    |    |     |     |     |
|             | 테스트 사업 실시     |    |    |    |    |    |    |    |    |    |     |     |     |
|             | 테스트 결과 정리     |    |    |    |    |    |    |    |    |    |     |     |     |
| PoC 테스트 4차  | 당행 로컬 환경 구성   |    |    |    |    |    |    |    |    |    |     |     |     |
|             | 기본 동작 테스트     |    |    |    |    |    |    |    |    |    |     |     |     |
|             | Solace 3센터 구성 |    |    |    |    |    |    |    |    |    |     |     |     |
|             | 성능 및 복원력 테스트  |    |    |    |    |    |    |    |    |    |     |     |     |
|             | 로직 보완 및 추가    |    |    |    |    |    |    |    |    |    |     |     |     |

\* PoC 테스트 1차 보고는 2024.8월 실시

## 〈참 고〉

- 솔루션 및 응용 프로그램 설치 내역은 크게 1번부터 9번까지로 구분되며 Kafka Confluent는 VM으로, 4가지 응용 프로그램은 프로세스로, 그 외 나머지 솔루션은 도커로 실행

### 로컬 환경 솔루션 설치 요약



## 1. OS

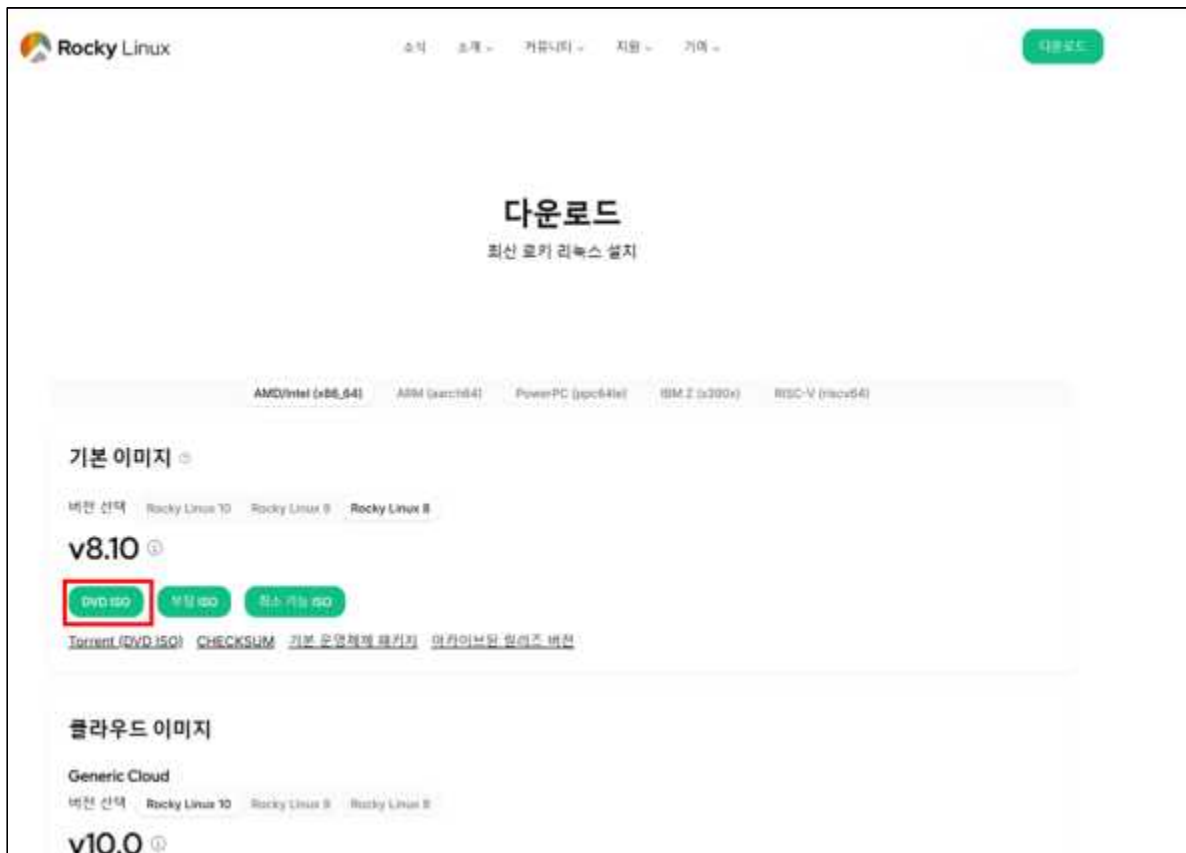
- OS(Operating System, 운영체제)는 고성능 PC에서는 Windows 11을 사용하며, Kafka Confluent 기동을 위한 VM의 OS로는 Rocky\*를 설치

\* Rocky : CentOS 배포 종료 이후 탄생한 RHEL(미국 소프트웨어 솔루션 제공 기업인 RedHat이 만든 리눅스) 호환 오픈소스이며 RedHat 계열과 안정적인 호환이 가능한 Kafka 설치 시 주로 사용

- o Windows 11은 당행 인터넷망에 기본 설치되어 있으므로 수동 설치의 생략

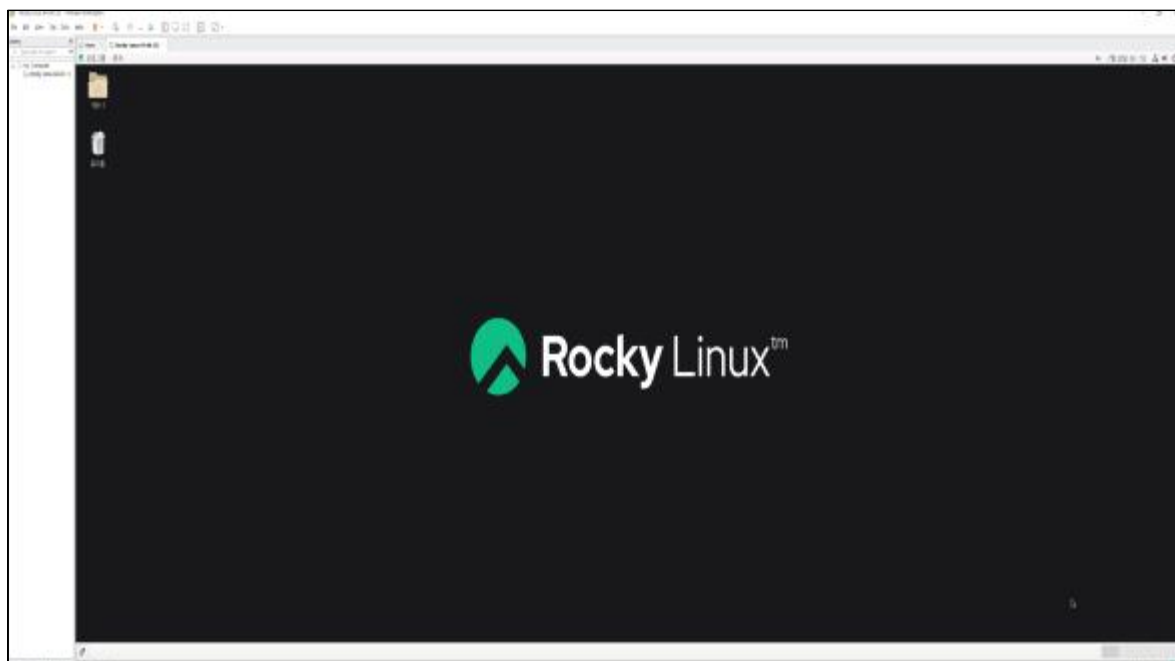
o Rocky는 Kafka Confluent 권장 사양인 8.10버전을 설치

### Rocky 8.10 다운로드\* 화면



\* 다운로드 링크 : <https://rockylinux.org/ko-KR/download>

### Rocky 8.10 설치 완료 화면



## 2. VMWare

- Kafka Confluent는 도커 배포용 이미지를 지원하지 않으므로 VM으로 구성해야 하며 이를 위해 VMWare\*를 설치하고 게스트 OS를 Rocky로 세팅

\* VMWare : 미국 소프트웨어 기업에서 개발한 가상화 플랫폼 도구이며 서버, 데스크탑, 네트워크 자원을 가상화하여 하나의 물리 장비 위에 여러 OS를 동시에 실행할 수 있도록 지원

- o VMWare는 Workstation Pro 17.6.3 버전을 설치

### VMWare 다운로드\* 화면



\* 다운로드 링크 :

<https://support.broadcom.com/group/ecx/productdownloads?subfamily=VMware%20Workstation%20Pro&freeDownloads=true>

- o Kafka Confluent를 설치하는 각 서버는 로컬 시간이 모두 동일해야 하기 때문에 'timedatectl'로 시간 확인 후, 다르다면 서버 간 시간을 동기화

- timedatectl set-timezone Asia/Seoul (KST 시간을 기준으로 설정)

### VM 시간확인 화면

```
[rtgs-1@localhost~]$timedatectl
Local time: 금 2025-07-18 09:52:09 KST
Universal time: 금 2025-07-18 00:52:09 UTC
RTC time: 금 2025-07-18 00:52:10
Time zone: Asia/Seoul (KST, +0900)
System clock synchronized: no
NTP service: active
RTC in local TZ: no
[rtgs-1@localhost~]$
```

### 3. JDK

- JDK(Java Development Kit)는 Java 어플리케이션을 개발하고 실행하기 위한 통합 개발 도구이며, Kafka Confluent를 실행하기 위한 필수 구성요소
- 업체 권장 사양인 OpenJDK 17.0.15 버전을 선택하며, VM의 Rocky에 설치될 것이므로 리눅스 운영체제에 맞는 파일을 다운로드

#### JDK 다운로드\* 화면



\* 다운로드 링크 : <https://adoptium.net/temurin/releases/?version=17&os=any&arch=any>

- 빌드 도구(Gradle), 어플리케이션 서버 등은 JAVA\_HOME 환경변수를 참조하여 JDK의 위치를 파악하기 때문에 bashrc 파일에 환경변수를 등록

#### JDK 환경변수 설정 화면

```
[rtgs-1@localhost~]$cat ~/.bashrc
export PS1="[\u@\h\w]\$ "
export JAVA_HOME=$HOME/java/jdk-17.0.15+6
export PATH=$JAVA_HOME/bin:$PATH
export JAVA_HOME=$HOME/java/jdk-17.0.15+6
export PATH=$JAVA_HOME/bin:$PATH
[rtgs-1@localhost~]$
```

- 사용 명령어
  - tar -xvzf OpenJDK17U-jdk\_x64\_linux\_hotspot\_17.0.15\_6.tar.gz
  - echo 'export JAVA\_HOME=\$HOME/Downloads/jdk-17.0.15+6' >> ~/.bashrc
  - echo 'export PATH=\$JAVA\_HOME/bin:\$PATH' >> ~/.bashrc
  - source ~/.bashrc

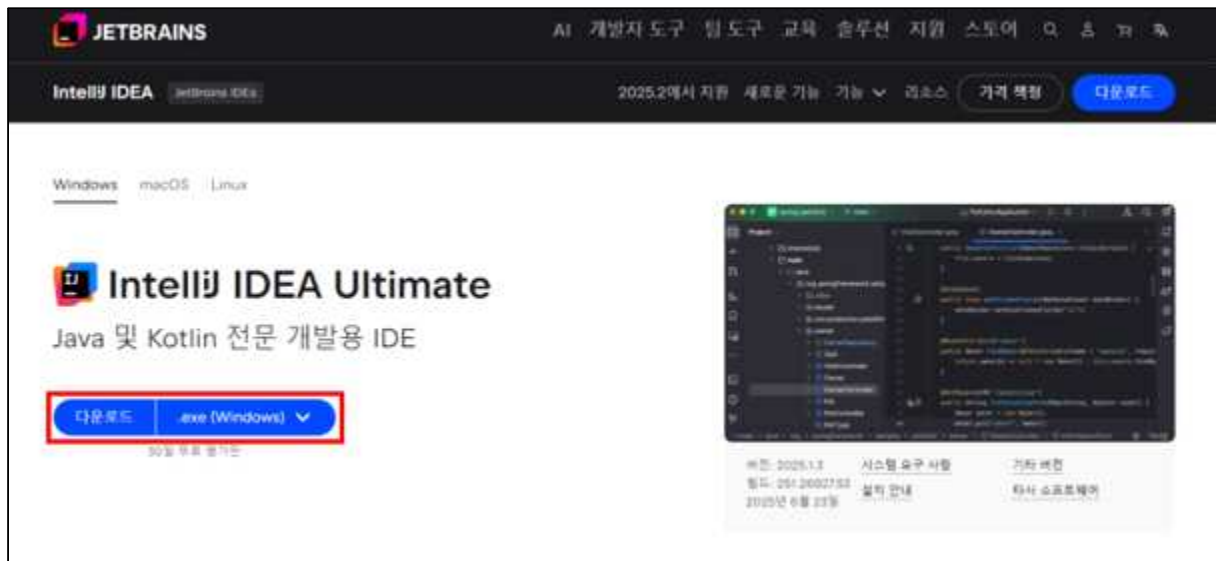
## 4. 개발도구

- 응용 프로그램(CHANEL, DIOR, HERMES, PRADA) 개발 및 실행을 위한 IDE\*로 IntelliJ IDEA Ultimate를 사용

\* IDE(Integrated Development Environment) : 소프트웨어 개발에 필요한 여러 도구를 하나의 프로그램에 통합한 환경으로 코드 작성부터 실행, 디버깅, 배포까지 한 곳에서 처리하도록 도와주는 도구

- 응용 프로그램 개발은 Kotlin 언어를 사용하였으므로 Java/Kotlin 기반 전문 IDE인 IntelliJ IDEA Ultimate 사용이 적합

### IntelliJ IDEA Ultimate 다운로드\* 화면



\* 다운로드 링크 : <https://www.jetbrains.com/ko-kr/idea/download/?section=windows>

- IntelliJ IDEA Ultimate에서 사용할 JDK는 Temurin 21버전을 사용

### IntelliJ IDEA Ultimate JDK 설정 화면



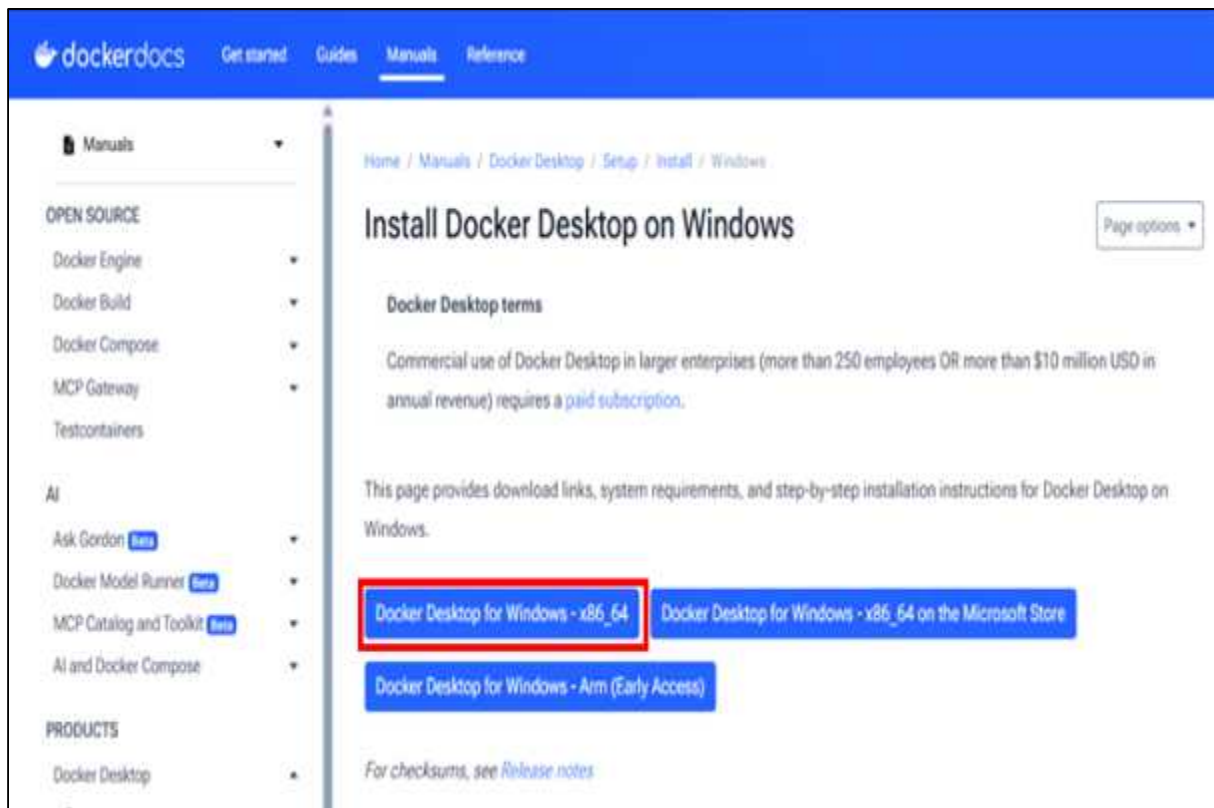
## 5. 배포도구

- 각 솔루션 및 응용 프로그램을 배포하여 하나의 서버(컨테이너 형태)로 실행하기 위해 도커를 사용
- 로컬 컴퓨터에 도커를 쉽게 설치하고 사용할 수 있도록 GUI 기반 통합 도구인 도커 Desktop을 사용

### 도커 Desktop 구성요소

| 구성요소                | 설명                                     |
|---------------------|----------------------------------------|
| 도커 Engine           | 도커 컨테이너를 실행하는데 필요한 핵심 런타임 도구           |
| 도커 CLI              | 배포 명령어를 Command Line 기반으로 입력할 수 있는 도구  |
| 도커 Compose          | 여러 컨테이너를 정의하고 실행할 수 있는 도구              |
| GUI 대시보드            | 실행 중인 컨테이너, 이미지, 네트워크 등을 시각적으로 관리하는 도구 |
| Volume/File Sharing | 호스트 PC와 컨테이너 간의 파일 공유 설정               |

### 도커 Desktop 다운로드\* 화면



\* 다운로드 링크 : <https://docs.docker.com/desktop/setup/install/windows-install/>

## 6. 네트워크 설정

□ 각 솔루션 및 응용 프로그램들은 서로 통신이 가능해야 하므로 각기 다른 센터의 고성능 PC간 통신, 한 센터 내 고성능 PC와 VM간 통신, 각기 다른 센터의 VM간 통신 모두 가능하도록 구성

○ 고성능 PC와 VM간 같은 대역의 브릿지\* 모드로 설정하며 V-ip(가상 ip)를 10.0.3.x 대역으로 고정

\* 브릿지(Bridge) 모드 : VM이 호스트 PC의 물리적 네트워크 어댑터(랜카드)를 통해 외부 네트워크에 직접 연결되어 마치 별도의 물리적 컴퓨터처럼 네트워크에 참여

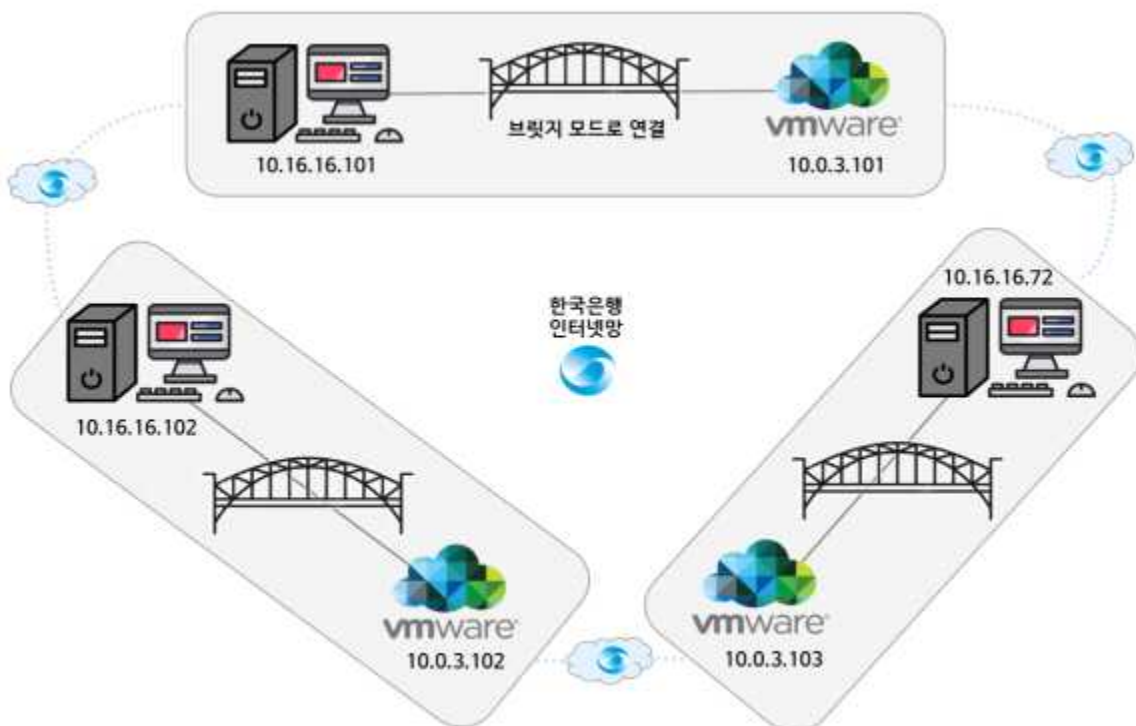
**VMWare 네트워크 모드 비교표**

|          | 브릿지 모드                  | NAT* 모드                         | Host-only** 모드                  |
|----------|-------------------------|---------------------------------|---------------------------------|
| ip 할당 방식 | 외부 네트워크(호스트 PC)에서 직접 받음 | NAT 네트워크에서 자동 할당                | Host-only 네트워크에서 자동 할당          |
| 외부 접속 여부 | 가능                      | 포트포워딩 설정 시 가능                   | 불가능                             |
| 용도       | VM을 서버처럼 운영 할 때 사용      | 일반적인 개발 환경으로 안전하게 외부 접속 필요 시 사용 | 테스트/로컬 개발 등 보안 상 격리된 환경 필요 시 사용 |

\* NAT(Network Address Translation) 모드 : 하나의 공인 IP로 여러 장치가 인터넷을 공유하는 방식

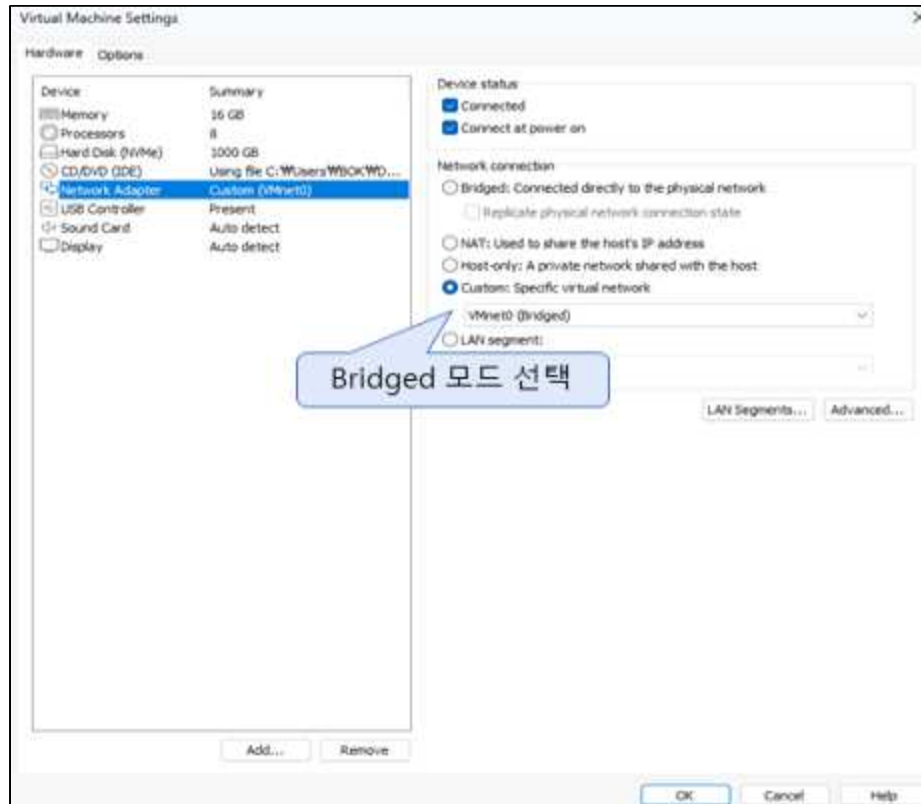
\*\* Host-only 모드 : 호스트와 VM만 통신 가능한 격리된 네트워크

### 네트워크 구성도

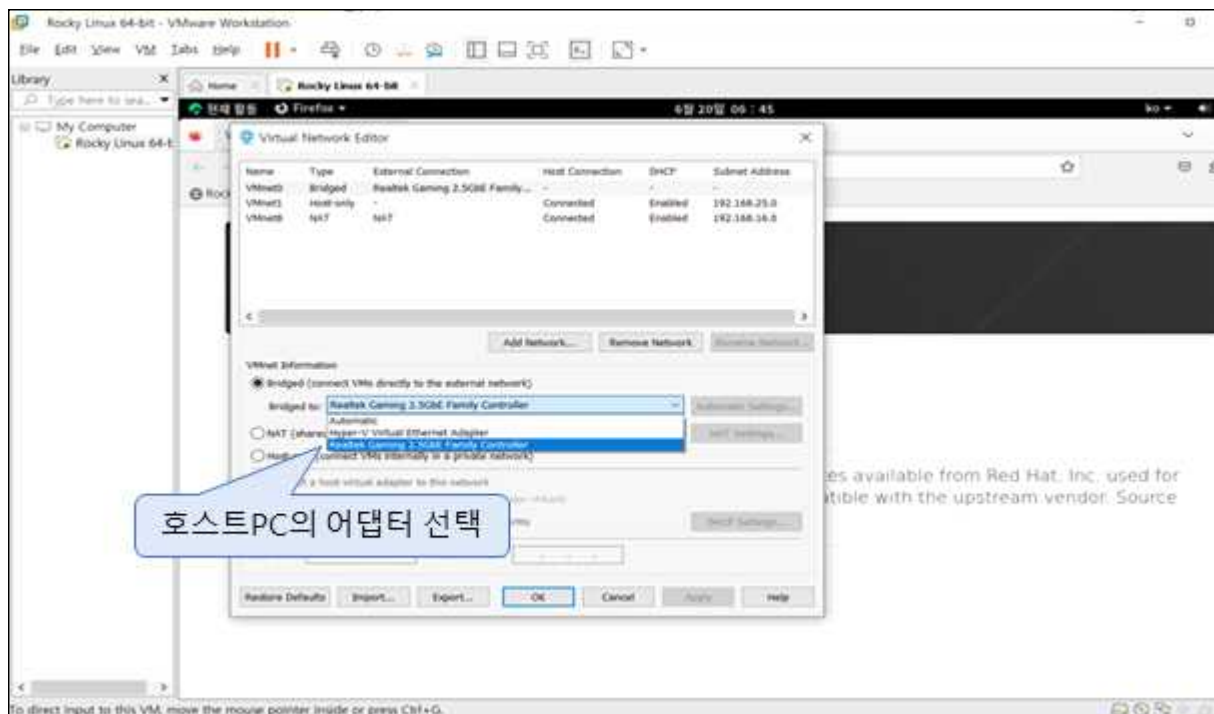


- o VMWare 설정 화면에서 네트워크(Bridge) 선택하고 연결될 브릿지로 호스트 PC의 어댑터 선택

### VM Settings 화면



### Virtual Network Editor 화면



o VM 터미널에서 nmcli\* 명령어를 사용하여 v-ip를 설정

\* nmcli(Network Manager CLI) : 리눅스 시스템에서 네트워크를 명령어로 관리할 수 있게 해주는 도구

- nmcli con show (이더넷 인터페이스명 확인, 예 : ens160)
- nmcli con mod ens160 ipv4.addresses 10.0.3.101/24 (ip주소 설정)
- nmcli con mod ens160 ipv4.gateway 10.0.3.1 (게이트웨이 설정)
- nmcli con mod ens160 ipv4.dns 8.8.8.8 (dns 설정)
- nmcli con mod ens160 ipv4.method manual (ip주소 자동 할당아닌 고정ip 모드로 설정)
- nmcli con down ens160 (네트워크 설정 끄기)
- nmcli con up ens160 (네트워크 설정 켜기)
- nmcli con mod ens160 connection.autoconnect yes (재부팅 시에도 이전 네트워크 설정 유지)
- sudo systemctl stop firewalld (방화벽 끄기)

### VM ip설정 완료 화면

```
[rtgs-1@localhost~]$ifconfig
ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.3.101 netmask 255.255.255.0 broadcast 10.0.3.255
    inet6 fe80::918f:d5d5:a0eb:b8d5 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:b0:53:b7 txqueuelen 1000 (Ethernet)
    RX packets 134 bytes 24216 (23.6 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 62 bytes 12137 (11.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 48 bytes 4962 (4.8 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 48 bytes 4962 (4.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 192.168.122.1 netmask 255.255.255.0 broadcast 192.168.122.255
    ether 52:54:00:66:34:3f txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

[rtgs-1@localhost~]$
```

## 7. 데이터베이스

□ 설치할 데이터베이스는 Hazelcast, MongoDB, Goldilocks, ELK가 있음

□ (7-1) IMDG인 Hazelcast 사용을 위해 도커 컨테이너로 배포

o 도커 네트워크 생성

- docker network create hazelcast-network

o 도커 파일 배포 (compose-hazel.yaml)

- docker compose -f compose-hazel.yaml up

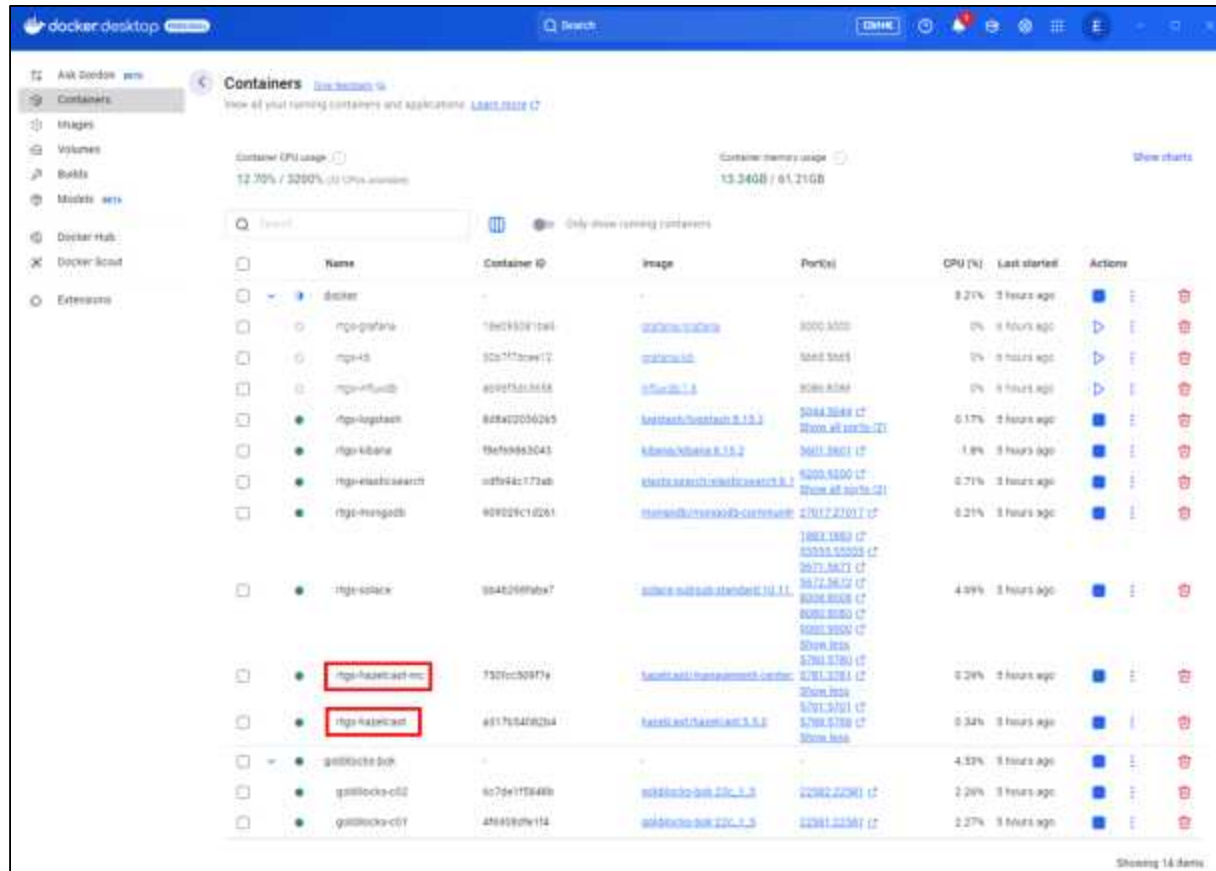
### compose-hazel.yaml 파일

```
services:
  hazelcast:
    image: hazelcast/hazelcast:5.5.8
    container_name: rtgs-hazelcast
    restart: always
    deploy:
      resources:
        limits:
          memory: 4g
          cpus: "1"
    environment:
      HAZELCAST_NETWORK_PUBLICADDRESS: 10.16.16.100:5701
      HAZELCAST_CLUSTERNAME: rtgs-hazelcast
      PROMETHEUS_PORT: 5788
      JAVA_OPTS: -Dhazelcast.shutdownhook.policy=GRACEFUL
      -Dhazelcast.graceful.shutdown.max.wait=10
    ports:
      - "5701:5701"
      - "5788:5788"
    networks:
      - hazelnet

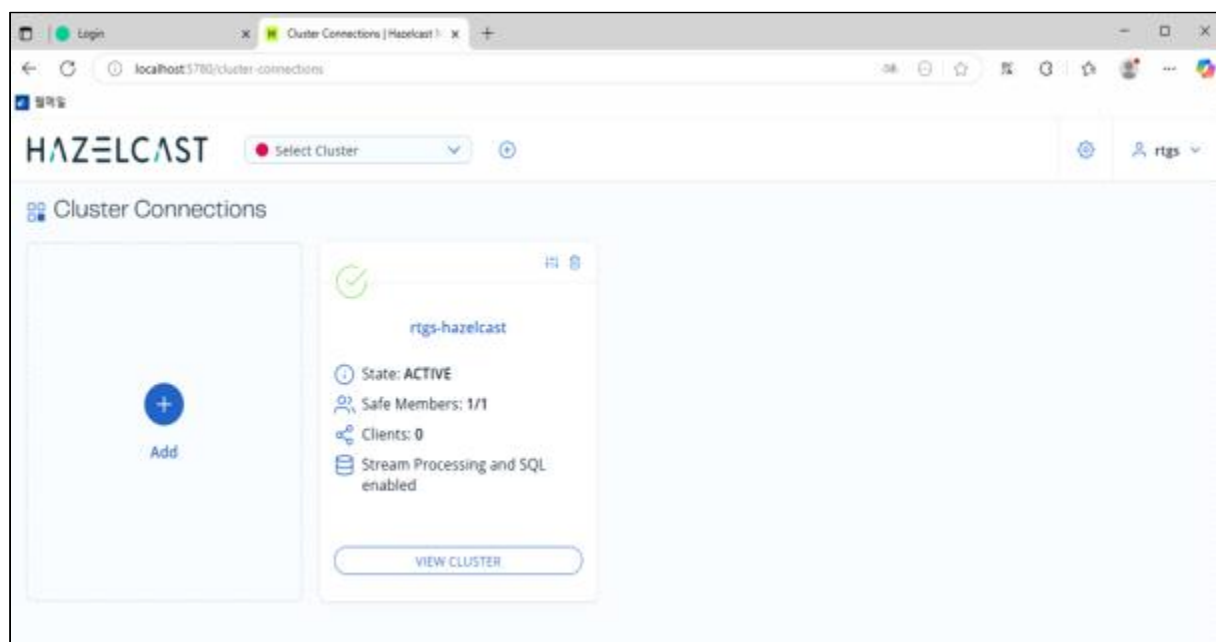
  hazelcast-mc:
    depends_on:
      - hazelcast
    image: hazelcast/management-center:5.5.8
    container_name: rtgs-hazelcast-mc
    restart: always
    deploy:
      resources:
        limits:
          memory: 4g
          cpus: "1"
    environment:
      MC_ADMIN_USER: rtgs
      MC_ADMIN_PASSWORD: hsk1357!! # Password must not contain the underscore. Password must not contain more than two sequential digits.
      MC_ALLOW_MULTIPLE_LOGIN: true
      MC_HTTP_PORT: 5780
      MC_HEALTH_CHECK_PORT: 5781
      MC_DEFAULT_CLUSTER: rtgs-hazelcast
      MC_DEFAULT_CLUSTER_MEMBERS: 10.16.16.101:5701
    volumes:
      - hazeldata:/data
    ports:
      - "5780:5780"
      - "5781:5781"
```

- o 도커 컨테이너 확인 및 Hazelcast Management Center(localhost:5780) 접속 확인

### 도커 Desktop 화면



### Hazelcast Management Center 접속 화면





## DataGrip 연결 설정 화면

The screenshot shows the 'Connect to new MongoDB instance' window in MongoDB Compass. The 'Driver' tab is active. The 'Host' field is set to 'localhost', and the 'Port' field is set to '27017'. The 'Username' field is 'rtgs' and the 'Password' field is 'rtgs1234!'. A callout bubble points to the 'Connect' button with the text '성공 시 접속완료' (Successful connection). The 'Database' field is empty, and the 'URL' field shows 'mongodb://localhost:27017/'.

## DataGrip 접속 화면

### □ (7-3) 인메모리 및 원장DB인 Goldilocks 사용을 위해 도커 컨테이너로 배포

#### o 도커 파일 배포 (compose-gd.yaml)

- docker compose -f compose-gd.yaml up
- 1센터는 goldilocks-c01/c02, 2센터는 goldilocks-s01/s02, 3센터는 goldilocks-k01/k02으로 컨테이너 이름을 설정

#### compose-gd.yaml 파일

```
name: goldilocks-bok

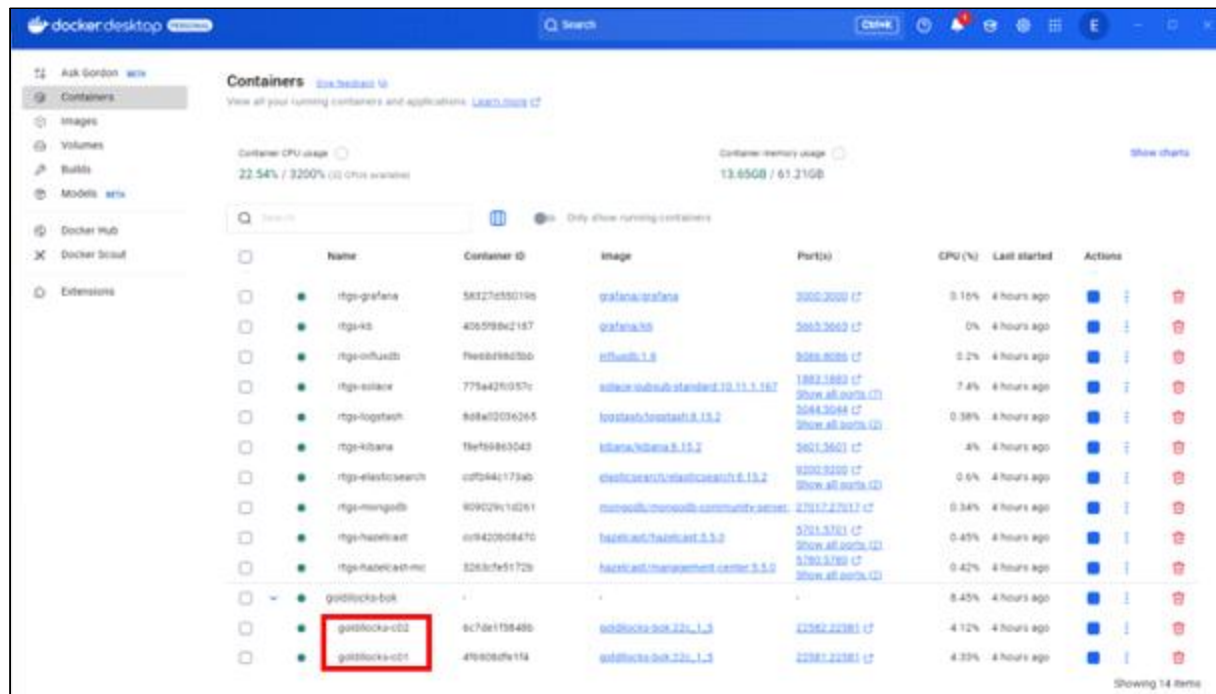
services:
  goldilocks-c01:
    image: localhost/goldilocks-bok:22c_1_5
    container_name: goldilocks-c01
    hostname: goldilocks-c01
    environment:
      - CONTAINER_TYPE=memory
    ports:
      - "22581:22581"
    volumes:
      - goldilocks_c01_data:/home/sunje/goldilocks_data
    restart: unless-stopped
    cpus: 8
    mem_limit: 8G

  goldilocks-c02:
    image: localhost/goldilocks-bok:22c_1_5
    container_name: goldilocks-c02
    hostname: goldilocks-c02
    environment:
      - CONTAINER_TYPE=disk
    ports:
      - "22582:22581"
    volumes:
      - goldilocks_c02_data:/home/sunje/goldilocks_data
    restart: unless-stopped
    cpus: 8
    mem_limit: 8G

volumes:
  goldilocks_c01_data: {}
  goldilocks_c02_data: {}
```

- ## o 도커 컨테이너 확인 및 cli 접속 확인

## 도커 Desktop 화면



## Goldilocks cli 접속 화면



- `docker exec -it goldilocks-c01 bash` (Goldilocks 도커 컨테이너 접속, 인메모리 DB)
- `docker exec -it goldilocks-c02 bash` (Goldilocks 도커 컨테이너 접속, 디스크 DB)
- `gsqlnet test test -dsn GOLDDILOCKS` (컨테이너 내부에서 DB 접속)

## □ (7-4) 전문DB 및 시각화 툴인 ELK 사용을 위해 도커 컨테이너로 배포

### o 도커 네트워크 생성

- docker network create elastic-network

### o 도커 파일 배포 (compose-elk.yaml)

- docker compose -f compose-elk.yaml up

### compose-elk.yaml 파일

```
services:
  logstash:
    MONITORING_ENABLED: true
    MONITORING_ELASTICSEARCH_HOSTS: http://10.16.16.101:9200
    volumes:
      - ./logstash/pipeline:/usr/share/logstash/pipeline/
      - ./logstash/config:/usr/share/logstash/config/
    deploy:
      resources:
        limits:
          memory: 2g
          cpus: '1'
      ports:
        - "5044:5044"
        - "9600:9600"

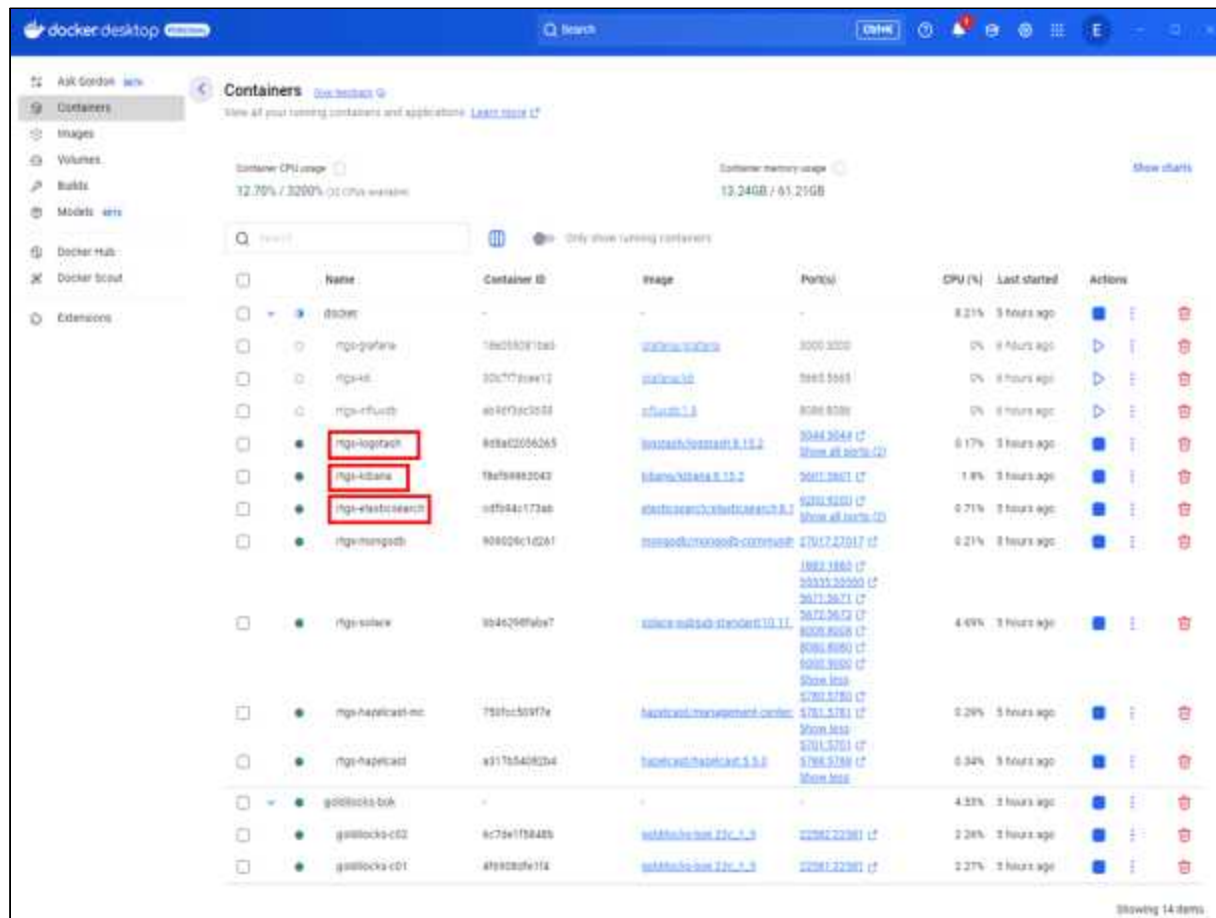
  kibana:
    depends_on:
      - elasticsearch
    image: docker.elastic.co/kibana/kibana:8.15.2
    container_name: rtgs-kibana
    restart: always
    environment:
      - ELASTICSEARCH_HOSTS=http://10.16.16.101:9200 # Link to Elasticsearch
      - xpack.security.enabled=false # disable security for simplicity
    deploy:
      resources:
        limits:
          memory: 2g
          cpus: '1'
      ports:
        - "5601:5601" # Kibana
    networks:
      - elastnet

volumes:
  elasticdata:
    driver: local

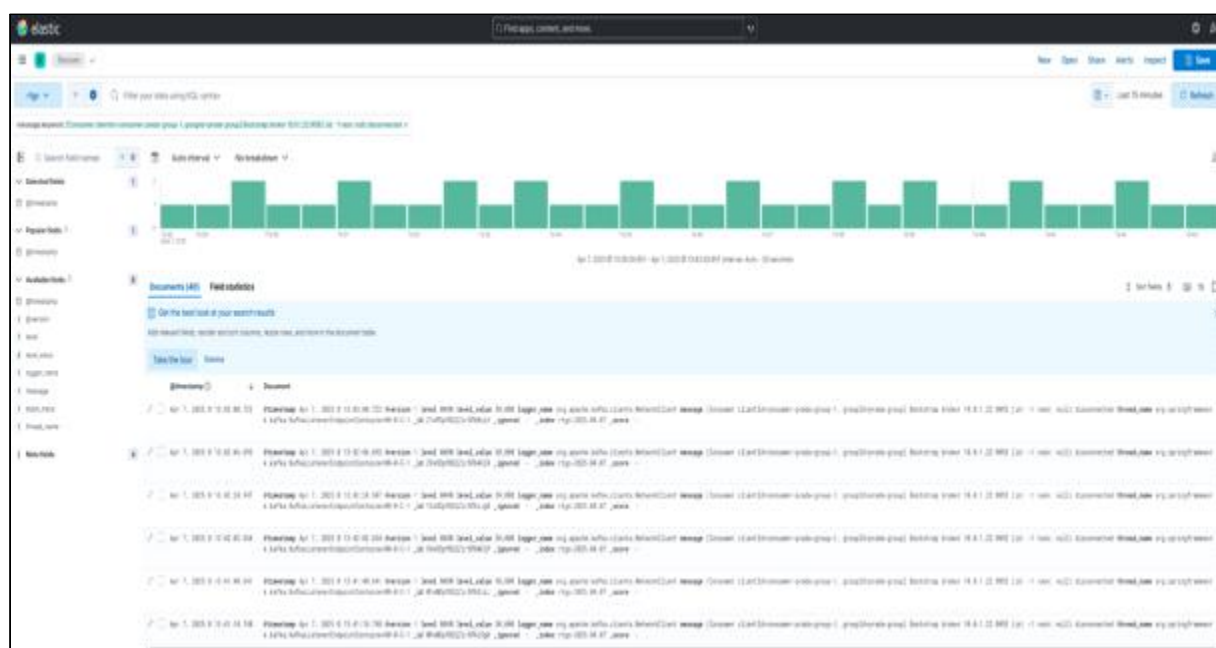
networks:
  elastnet:
    name: elastic-network
    external: true
```

### o 도커 컨테이너 확인 및 Kibana(localhost:5601) 접속 확인

## 도커 Desktop 화면

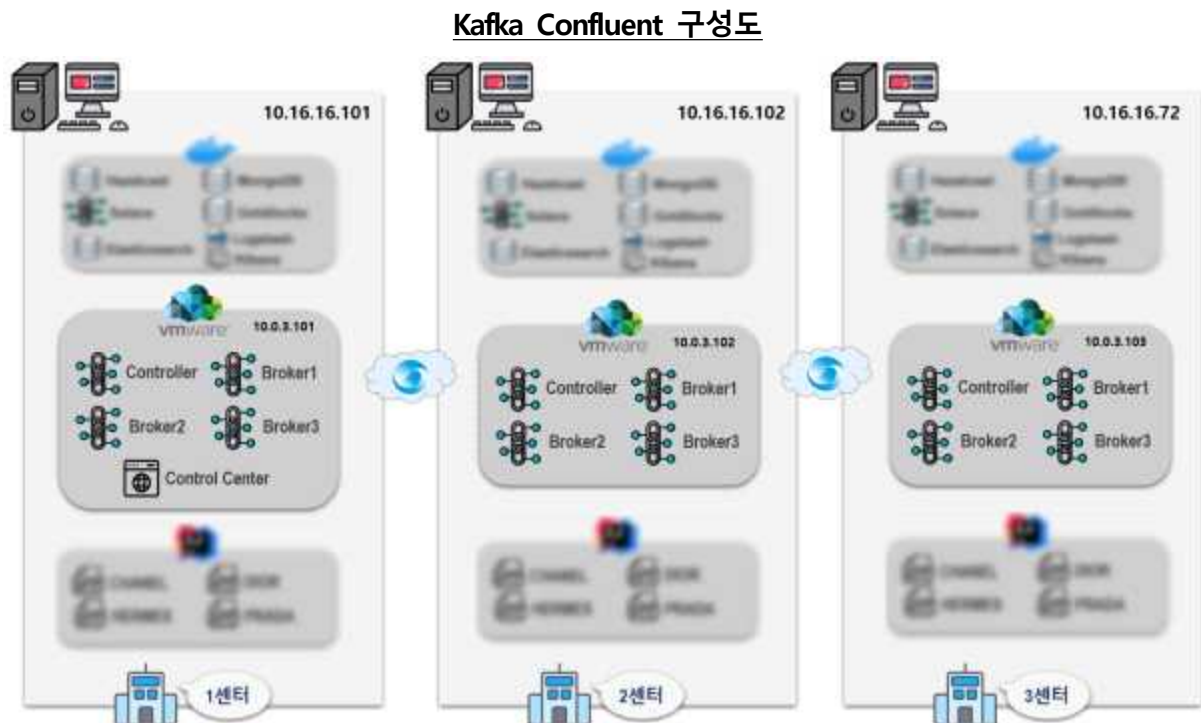


## Kibana 접속 화면



## 8. 이벤트 브로커

- Kafka Confluent는 3개 센터(고성능 PC)에 브로커 3대, 컨트롤러 3대, 컨트롤 센터 1대(1센터만 구성)으로 구성



o Kafka Confluent 컨트롤러, 브로커, 컨트롤 센터 기동

- cd /confluent/engn/confluent/scripts
- ./start-controller.sh (컨트롤러 기동)
- ./start-broker1.sh (브로커1 기동)
- ./start-broker2.sh (브로커2 기동)
- ./start-broker3.sh (브로커3 기동)
- ./start-control-center.sh (컨트롤 센터 기동)

o Kafka Confluent 컨트롤러, 브로커, 컨트롤 센터 기동 중지

- ./stop-controller.sh (컨트롤러 중지)
- ./stop-broker1.sh (브로커1 중지)
- ./stop-broker2.sh (브로커2 중지)

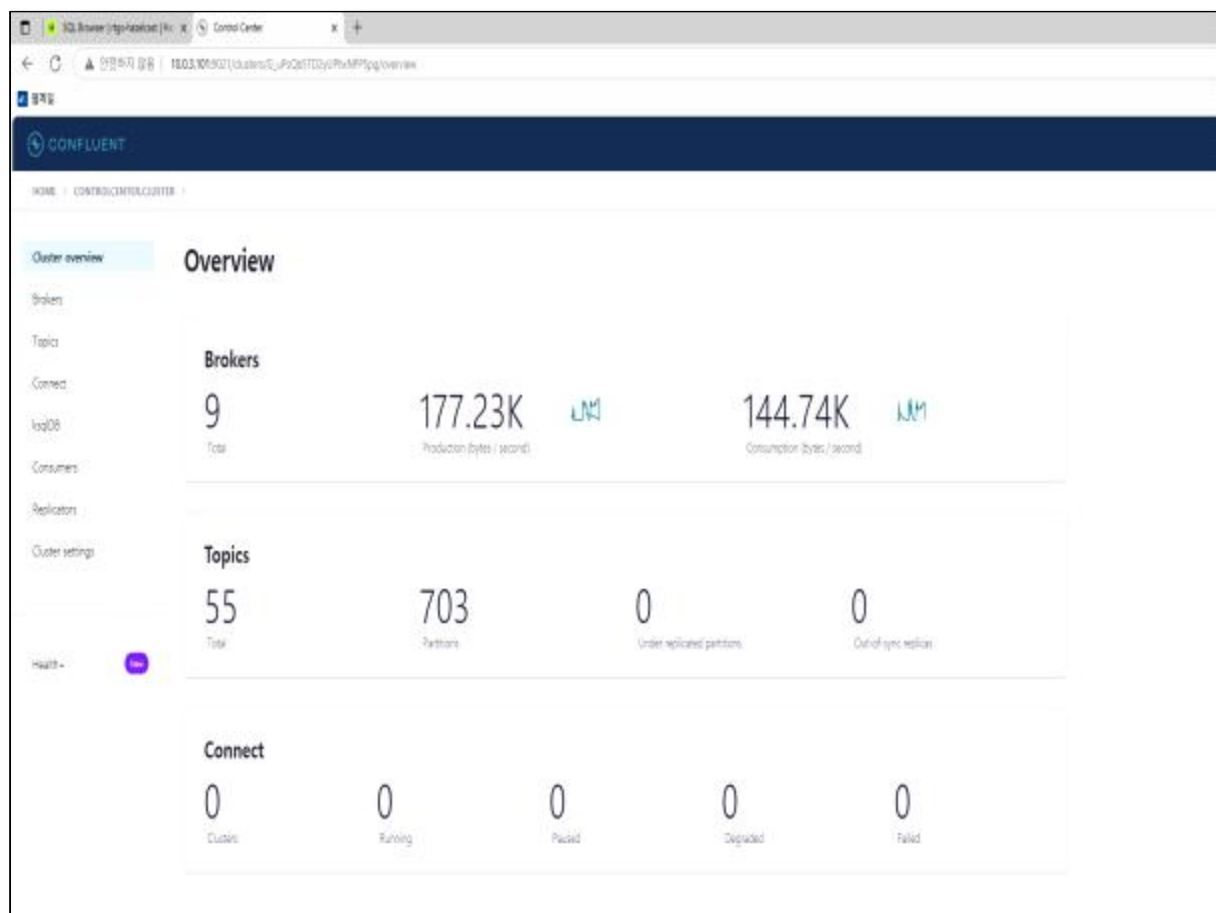
- ./stop-broker3.sh (브로커3 중지)
- ./stop-control-center.sh (컨트롤 센터 중지)

o Kafka Confluent 메타 데이터 초기화(에러로 인해 데이터 손상되었을 시)

- `rm -rf /confluent/data/{controller, broker1, broker2, broker3}/*` (모든 데이터 디렉토리 삭제)
- `ls -al /confluent/data/{controller, broker1, broker2, broker3}/` (숨김파일 확인 후 삭제)
- `cd /confluent/engn/confluent/bin`
- `./kafka-storage format --config ../properties/{controller, broker1, broker2, broker3}.properties --cluster-id G_uPsQb5TD2yUPhxNPP5pg` (데이터 포맷)

o 컨트롤 센터에서 정상 기동 확인

### 컨트롤 센터 화면



□ Solace의 경우 현재 2개 센터로 구성되어 있으며 3개 센터 구성 방안은 논의 중

o 도커 이미지 생성

- docker load -i solace-pubsub-standard-10.11.1.167-docker.tar.gz

o 도커 파일 배포 (compose-solace.yaml)

- docker compose -f compose-solace.yaml up

### compose-solace.yaml 파일



o Solace 큐, 토픽, 브릿지 설정

- docker cp ./rtgs-solace01\_backup\_20250618.cli

rtgs-solace:/usr/sw/jail/cliscripts/ (1센터 도커 설정파일 복사)

- docker cp ./rtgs-solace11\_backup\_20250618.cli

rtgs-solace:/usr/sw/jail/cliscripts/ (2센터 도커 설정파일 복사)

- docker exec -it rtgs-solace /bin/bash (도커 컨테이너 진입)

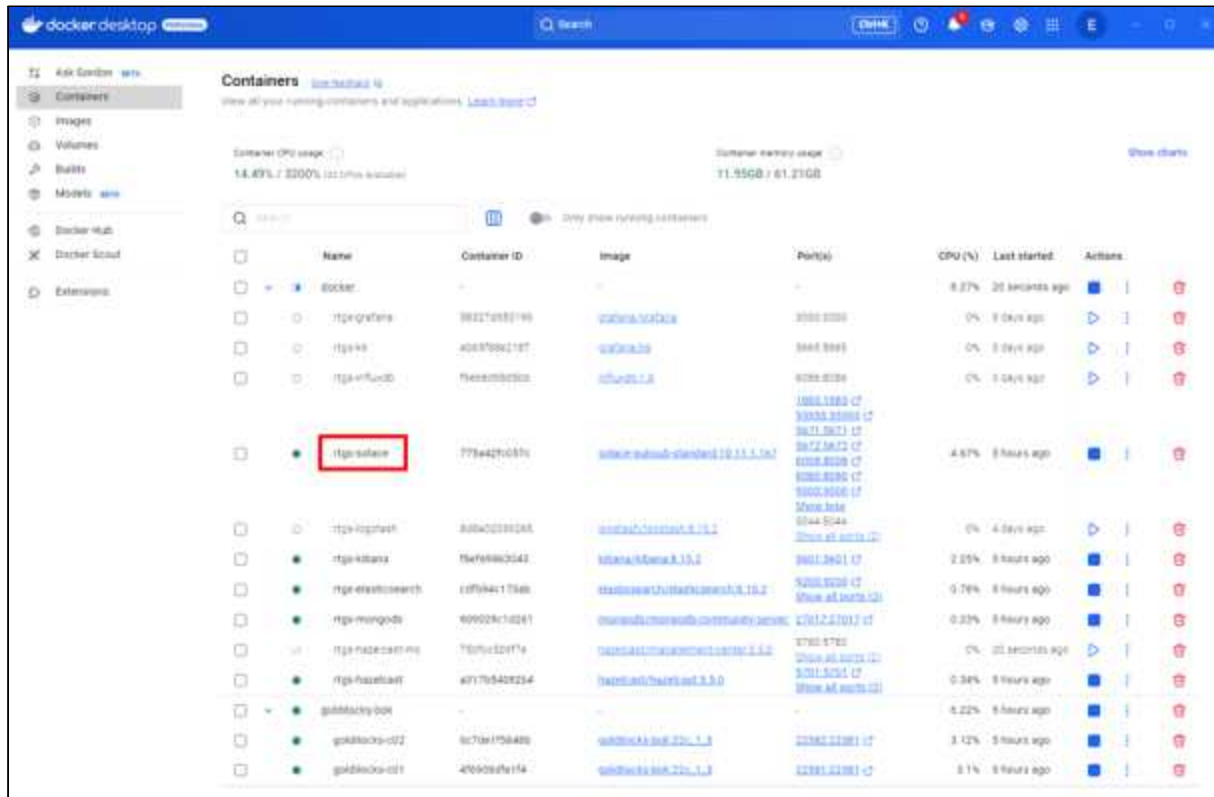
- cli (커맨드 라인 진입)

- source script rtgs-solace01\_backup\_20250618.cli (1센터 설정파일 적용)

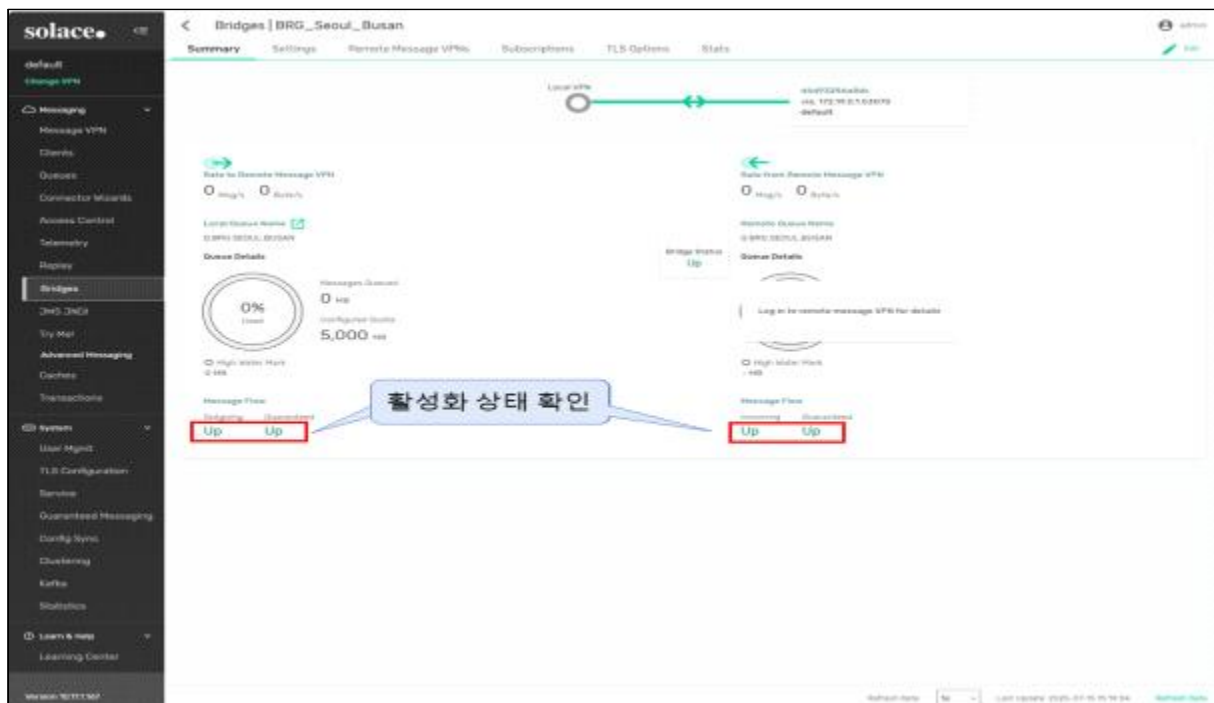
- source script rtgs-solace11\_backup\_20250618.cli (2센터 설정파일 적용)

o 도커 컨테이너 확인 및 Solace 모니터링 센터(localhost:8080) 접속 확인

## 도커 Desktop 화면



## Solace 모니터링 센터 접속 화면



## 9. 응용 프로그램

□ 응용 프로그램은 CHANEL(전문 송수신), DIOR(접수 동기화), HERMES(결제 처리), PRADA(결과 동기화)가 있으며 IntelliJ로 소스코드를 실행

o CHANEL(Contact Hub for Advanced Networked Entry Liaison) : 참가기관(송신)으로부터 거래요청 건을 접수받아 참가기관(송신, 수신)에게 처리결과를 통보하는 서비스

① ISO20022 전문에서 핵심 전문을 추출·생성, ② 인메모리DB에 핵심 전문을 접수(RCVD)하여 유효성 체크 후 전문상태값을 승인상태(ACCT)로 수정, ③핵심 전문을 이벤트 브로커의 접수 토픽에 등록

o DIOR(Data Integration and Organization Relay) : 참가기관(송신)에게 접수한 거래건을 다중 센터에서 동일하게 처리하기 위한 접수 동기화 서비스

④ 접수 토픽에 등록된 핵심 전문을 입수, ⑤ 접수 토픽에서 입수한 전문은 모두 대기상태(PDNG)로 원장DB에 저장

o HERMES(High-Efficiency Rapid Memory Execution for Settlement) : 각 센터에서 동일한 거래를 순서대로 처리하여 각 센터 내에 위치한 원장 데이터베이스에 저장하는 결제 처리 서비스

⑥ 접수 토픽에 등록된 핵심 전문을 입수, ⑦ 입수된 핵심 전문을 대기상태(PDNG)로 저장하고 계좌 데이터의 잔액 정보를 수정한 후 동 전문의 상태값을 예약상태(ACSP)로 수정, ⑧핵심 전문을 이벤트 브로커의 결과 토픽에 등록

o PRADA(Persistent Repository Administrator for Data Aggregation) : 각 센터에서 처리한 거래처리 결과에 대한 결과 동기화 서비스

⑨ 결과 토픽에 등록된 핵심 전문을 입수, ⑩ 원장DB의 계좌 데이터의 잔액 정보를 수정하고 처리상태값을 입금처리완료(ACCC)로 수정, ⑪ 인메모리 DB의 핵심 전문의 처리상태값을 입금처리완료(ACCC)로 수정

### 전문 송수신 프로그램(Chanel)

| 구분             | 소스코드                      | 기능요약                               | 비고*      |
|----------------|---------------------------|------------------------------------|----------|
| chanel         | ChanelApplication         | 전문송수신 응용시스템 초기화 실행                 |          |
| application    | ChanelService             | 전문송수신 서비스                          | (수정)     |
| domain         | Transfer                  | 자금이체 엔티티(송금) 데이터 관리                | (수정)     |
| infrastructure | ChanelExceptionHandler    | 전문송수신 예외처리                         |          |
| config         | ChanelConfig              | 전문송수신 환경설정                         |          |
|                | HazelcastConfig           | 인메모리DB 환경설정                        | 1-1, 2-1 |
|                | KafkaProducerConfig       | 이벤트브로커 프로듀서 환경설정                   | 1-1, 1-2 |
|                | GoldilocksMemoryConfig    | Goldilocks 인메모리DB 환경설정             | 1-1, 2-2 |
|                | SolaceConfig              | 이벤트브로커 환경설정                        | 2-1, 2-2 |
|                | HazelcastProperties       | 인메모리DB 환경설정                        | 1-1, 2-1 |
|                | TransferRepository        | Goldilocks 인메모리DB 쿼리 정의            | 1-1, 2-2 |
| presentation   | BmiGenerator              | 거래고유식별자 전문번호(BMI) 생성               |          |
|                | XmlValidator              | XML 유효성 체크(pacs.002)               |          |
|                | ChanelController          | 전문송수신 제어 처리                        |          |
|                | RequestEnvelope           | XML 객체 변환처리                        |          |
|                | ResponseEnvelope          | XML 응답 처리(pacs.002)                |          |
| resources      | application.yml           | 응용시스템 환경설정                         | (수정)     |
|                | application-hazelcast.yml | 인메모리 응용시스템 환경설정                    | 1-1, 2-1 |
|                | application-kafka.yml     | 이벤트브로커 응용시스템 환경설정                  | 1-1, 1-2 |
|                | application-solace.yml    | 이벤트브로커 응용시스템 환경설정                  | 2-1, 2-2 |
|                | logback-spring.xml        | Logstash 로그관리                      |          |
|                | head.001.001.03.xsd       | ISO20022 전문헤더(BAH) 체크              |          |
|                | pacs.002.001.13.xsd       | ISO20022 전문(응답) 본문 체크              |          |
|                | pacs.008.001.11.xsd       | ISO20022 전문(요청) 본문 체크              |          |
|                | pacs008template.xsd       | ISO20022 전문(요청) 본문 템플릿             |          |
|                | .gitignore                | 컨테이너 환경에서 프로젝트 빌드 준비               |          |
| 적용배포           | build.gradle.kts          | Spring Boot 기반 Kotlin 프로젝트 설정      |          |
|                | Dockerfile                | Java Spring Boot 빌드 실행 컨테이너 이미지 생성 |          |
|                | Dockerfile.gradle-cache   | Gradle 빌드 프로세스 캐시 이미지 생성           |          |
|                | gradlew                   | 프로젝트 빌드와 실행 스크립트                   |          |
|                | gradlew.bat               | Gradle Wrapper 실행 배치 스크립트          |          |
|                | settings.gradle.kts       | Gradle 프로젝트의 루트 설정                 |          |

\* 케이스별 수정사항 및 적용 대상

### 접수 동기화 프로그램(Dior)

| 구분             |            | 소스코드                      | 기능요약                                  | 비고       |
|----------------|------------|---------------------------|---------------------------------------|----------|
| dior           |            | DiorApplication           | 접수동기화    응용시스템    초기화 실행              |          |
| application    |            | DiorService               | Kafa 기반    접수동기화    서비스               | 1-1, 1-2 |
|                |            | SolaceSubscriber          | Solace 기반    접수동기화    서비스             | 2-1, 2-2 |
| domain         |            | Transfer                  | 자금이체    엔티티(송금)    데이터 관리             | (수정)     |
| infrastructure | config     | MongoConfig               | 몽고DB    환경설정                          | 1-1, 2-1 |
|                |            | HazelcastConfig           | 인메모리DB    환경설정                        | 1-1, 2-1 |
|                |            | KafkaConsumerConfig       | 이벤트브로커    컨슈머    환경설정                 | 1-1, 1-2 |
|                |            | GoldilocksDiskConfig      | Goldilocks    원장DB    환경설정            | 1-2, 2-2 |
|                |            | GoldilocksMemoryConfig    | Goldilocks    인메모리DB    환경설정          | 1-2, 2-2 |
|                | properties | HazelcastProperties       | 인메모리DB    환경설정                        | 1-1, 2-1 |
|                | repository | DiskTransferRepository    | Goldilocks    원장DB에서    사용할 쿼리 정의     | 1-2, 2-2 |
|                |            | MemoryTransferRepository  | Goldilocks    인메모리DB에서    사용할 쿼리 정의   | 1-2, 2-2 |
| resources      |            | application.yml           | 응용시스템    환경설정                         | (수정)     |
|                |            | application-database.yml  | 응용시스템    DB    환경설정                   | 1-1, 2-1 |
|                |            | application-hazelcast.yml | 인메모리    응용시스템    환경설정                 | 1-1, 2-1 |
|                |            | application-kafka.yml     | 메시지브로커    응용시스템    환경 설정              | 1-1, 1-2 |
|                |            | application-solace.yml    | 이벤트브로커    응용시스템    환경 설정              | 2-1, 2-2 |
|                |            | logback-spring.xml        | Logstash    로그관리                      |          |
| 적용배포           |            | .gitignore                | 컨테이너    환경에서    프로젝트 빌드 준비            |          |
|                |            | build.gradle.kts          | Spring Boot 기반    Kotlin    프로젝트 설정   |          |
|                |            | Dockerfile                | Java Spring Boot 빌드 실행    컨테이너 이미지 생성 |          |
|                |            | Dockerfile.gradle-cache   | Gradle 빌드    프로세스 캐시    이미지 생성        |          |
|                |            | gradlew                   | 프로젝트 빌드와 실행    스크립트                   |          |
|                |            | gradlew.bat               | Gradle Wrapper 실행    배치    스크립트       |          |
|                |            | settings.gradle.kts       | Gradle    프로젝트의    루트 설정              |          |

## 결제 처리 프로그램(Hermes)

| 구분             |            | 소스코드                      | 기능요약                                                 | 비고       |
|----------------|------------|---------------------------|------------------------------------------------------|----------|
| hermes         |            | HermesApplication         | 결제처리 응용시스템 초기화 실행                                    |          |
| application    |            | HermesService             | Kafka 기반 결제처리 서비스                                    | 1-1, 1-2 |
|                |            | SolaceSubscriber          | Solace 기반 결제처리 서비스                                   | 2-1, 2-2 |
|                |            | TransferService           | Goldilocks에서 트랜잭션 데이터를 관리할 함수 정의                     | 1-2, 2-2 |
| domain         |            | Transfer                  | 자금이체 엔티티(송금) 데이터 관리                                  | (수정)     |
|                |            | Ledger                    | DB를 기반으로 원장 엔티티 계좌(Account) 정보를 관리하고, 잔액을 실시간으로 업데이트 | (수정)     |
| infrastructure | config     | HermesConfig              | 결제처리 환경설정                                            |          |
|                |            | HazelcastConfig           | 인메모리DB 환경설정                                          | 1-1, 2-1 |
|                |            | KafkaConsumerConfig       | 이벤트브로커 컨슈머 환경설정                                      | 1-1, 1-2 |
|                |            | KafkaProducerConfig       | 이벤트브로커 프로듀서 환경설정                                     | 1-1, 1-2 |
|                |            | SolaceConfig              | 이벤트브로커 환경설정                                          | 2-1, 2-2 |
|                |            | GoldilocksMemoryConfig    | Goldilocks 인메모리DB 환경설정                               | 1-2, 2-2 |
|                | properties | HazelcastProperties       | 인메모리DB 환경설정                                          | 1-1, 2-1 |
|                | repository | TransferRepository        | Goldilocks 인메모리DB에서 사용할 쿼리 정의                        | 1-2, 2-2 |
| resources      |            | application.yml           | 응용시스템 환경설정                                           | (수정)     |
|                |            | application-hazelcast.yml | 인메모리 응용시스템 환경설정                                      | 1-1, 2-1 |
|                |            | application-kafka.yml     | 이벤트브로커 응용시스템 환경설정                                    | 1-1, 1-2 |
|                |            | application-solace.yml    | 이벤트브로커 응용시스템 환경설정                                    | 2-1, 2-2 |
|                |            | logback-spring.xml        | Logstash 로그관리                                        |          |
| 적용배포           |            | .gitignore                | 컨테이너 환경에서 프로젝트 빌드 준비                                 |          |
|                |            | build.gradle.kts          | Spring Boot 기반 Kotlin 프로젝트 설정                        |          |
|                |            | Dockerfile                | Java Spring Boot 빌드 실행 컨테이너 이미지 생성                   |          |
|                |            | Dockerfile.gradle-cache   | Gradle 빌드 프로세스 캐시 이미지 생성                             |          |
|                |            | gradlew                   | 프로젝트 빌드와 실행 스크립트                                     |          |
|                |            | gradlew.bat               | Gradle Wrapper 실행 배치 스크립트                            |          |
|                |            | settings.gradle.kts       | Gradle 프로젝트의 루트 설정                                   |          |

### 결과 동기화 프로그램(Prada)

| 구분                     |            | 소스코드                      | 기능요약                               | 비고       |
|------------------------|------------|---------------------------|------------------------------------|----------|
| prada                  |            | PradaApplication          | 결과동기화 응용시스템 초기화 실행                 |          |
| application            |            | PradaService              | Kafka 기반 결과동기화 서비스                 | 1-1, 1-2 |
|                        |            | SolaceSubscriber          | Solace 기반 결과동기화 서비스                | 2-1, 2-2 |
|                        |            | TransferService           | Goldilocks에서 트랜잭션 데이터를 관리할 함수 정의   | 1-2, 2-2 |
| domain                 |            | Transfer                  | 자금이체 엔티티                           | (수정)     |
|                        |            | Ledger                    | 원장 엔티티                             | (수정)     |
| infra<br>struc<br>ture | config     | PradaConfig               | 결과동기화 환경설정                         |          |
|                        |            | HazelcastConfig           | 인메모리DB 환경설정                        | 1-1, 2-1 |
|                        |            | KafkaConsumerConfig       | 이벤트브로커 컨슈머 환경설정                    | 1-1, 1-2 |
|                        |            | SolaceConfig              | 이벤트브로커 환경설정                        | 2-1, 2-2 |
|                        |            | GoldilocksDiskConfig      | Goldilocks 원장DB 환경설정               | 1-2, 2-2 |
|                        |            | GoldilocksMemoryConfig    | Goldilocks 인메모리DB 환경설정             | 1-2, 2-2 |
|                        | properties | HazelcastProperties       | 인메모리DB 환경설정                        | 1-1, 2-1 |
|                        | repository | DiskTransferRepository    | Goldilocks 원장DB에서 사용할 쿼리 정의        | 1-2, 2-2 |
|                        |            | MemoryTransferRepository  | Goldilocks 인메모리DB에서 사용할 쿼리 정의      | 1-2, 2-2 |
| resources              |            | application.yml           | 응용시스템 환경설정                         | (수정)     |
|                        |            | application-database.yml  | 응용시스템 DB 환경설정                      | 1-1, 2-1 |
|                        |            | application-hazelcast.yml | 인메모리 응용시스템 환경설정                    | 1-1, 2-1 |
|                        |            | application-kafka.yml     | 이벤트브로커 응용시스템 환경설정                  | 1-1, 1-2 |
|                        |            | application-solace.yml    | 이벤트브로커 응용시스템 환경설정                  | 2-1, 2-2 |
|                        |            | logback-spring.xml        | Logstash 로그관리                      |          |
| 적용배포                   |            | .gitignore                | 컨테이너 환경에서 프로젝트 빌드 준비               |          |
|                        |            | build.gradle.kts          | Spring Boot 기반 Kotlin 프로젝트 설정      |          |
|                        |            | Dockerfile                | Java Spring Boot 빌드 실행 컨테이너 이미지 생성 |          |
|                        |            | Dockerfile.gradle-cache   | Gradle 빌드 프로세스 캐시 이미지 생성           |          |
|                        |            | gradlew                   | 프로젝트 빌드와 실행 스크립트                   |          |
|                        |            | gradlew.bat               | Gradle Wrapper 실행 배치 스크립트          |          |
|                        |            | settings.gradle.kts       | Gradle 프로젝트의 루트 설정                 |          |