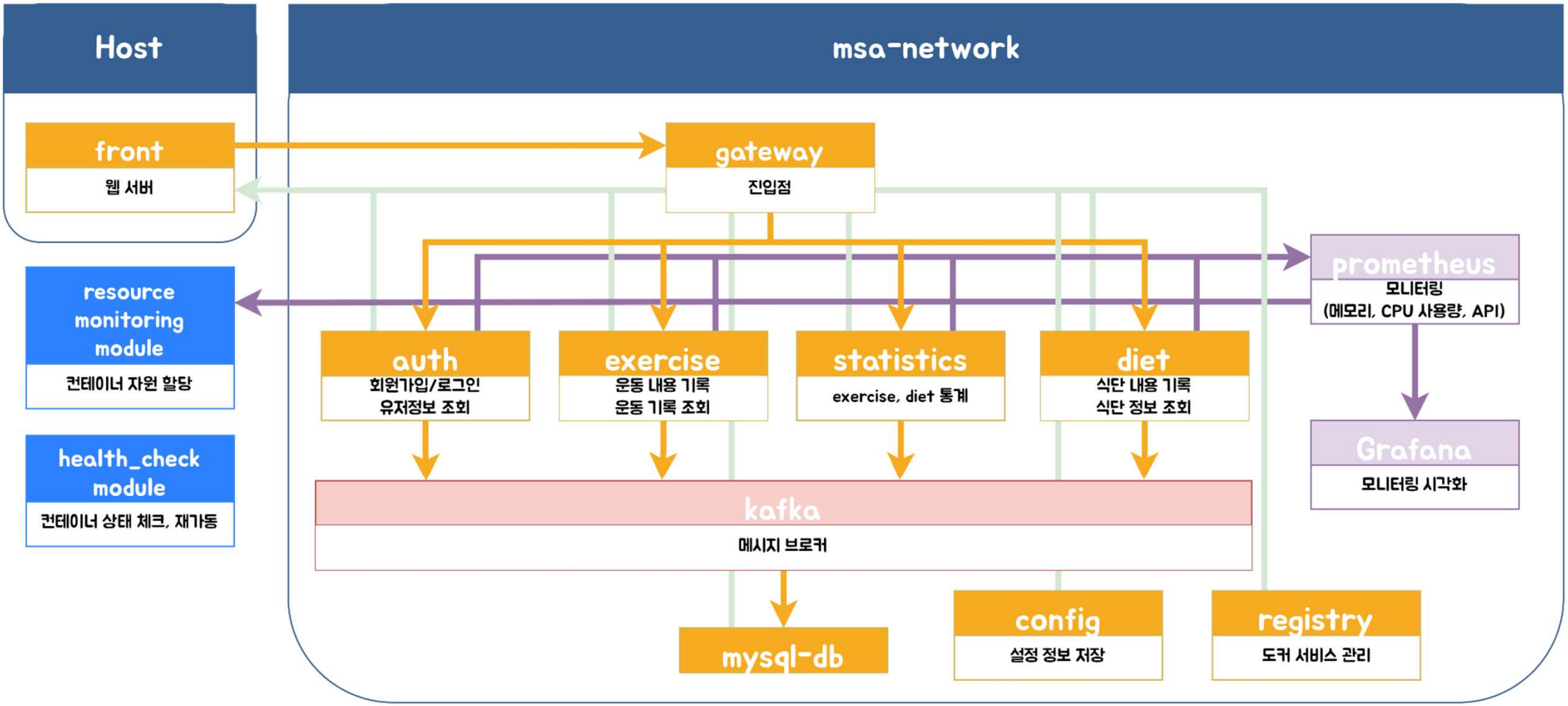
최종 설계안

이진트리구조

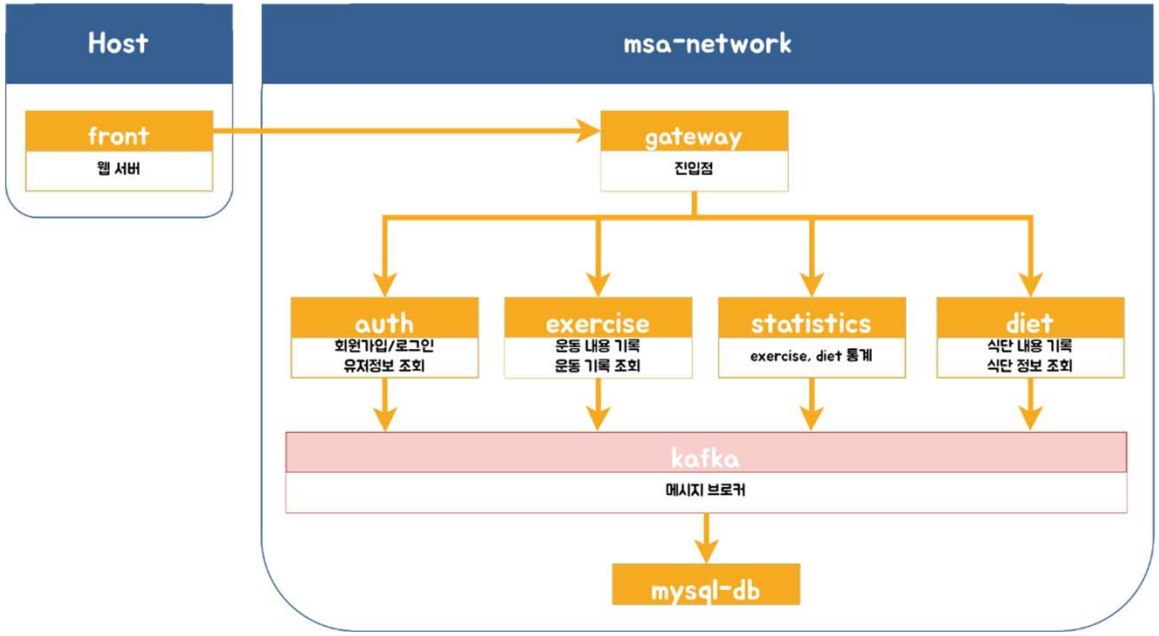
김동현, 김기현, 송경진, 최혜민, 허민

목차 Index

전체 구조도

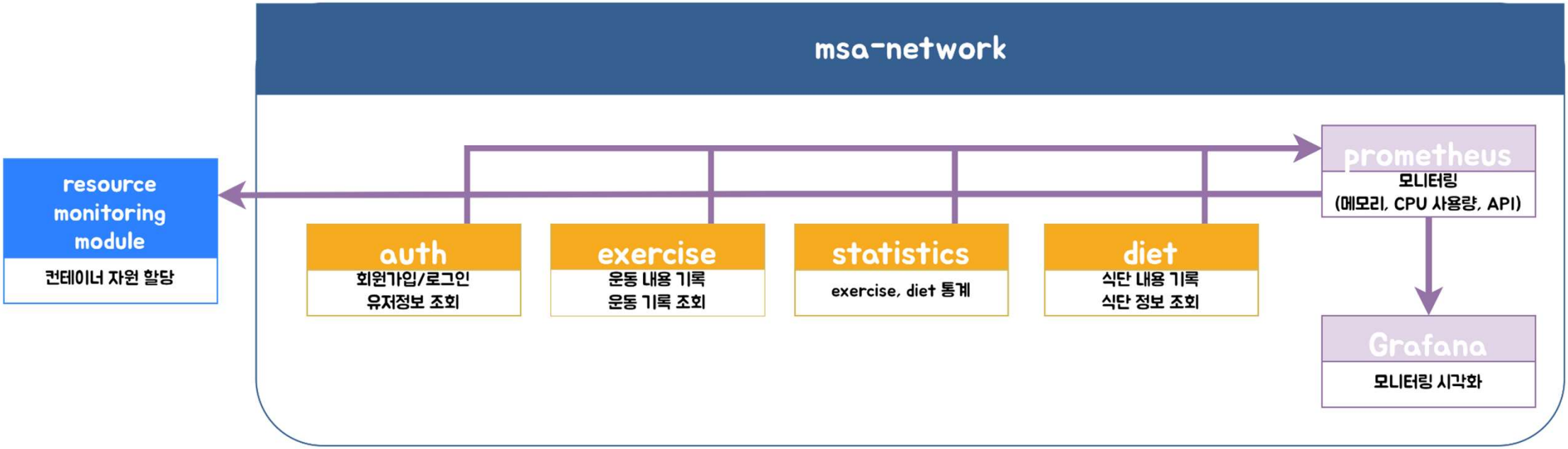

Host
msa-network
front
웹 서버
gateway
진입점
resource monitoring module
컨테이너 자원 할당
health_check module
컨테이너 상태 체크, 재가동
prometheus
모니터링
(메모리, CPU 사용량, API)
auth
회원가입/로그인
유저정보 조회
exercise
운동 내용 기록
운동 기록 조회
statistics
exercise, diet 통계
diet
식단 내용 기록
식단 정보 조회
Grafana
모니터링 시각화
kafka
메시지 브로커
mysql-db
config
설정 정보 저장
registry
도커 서비스 관리

사용자 영역 구조도



- 헬스/영양 관리 애플리케이션 오픈소스 사용
- 사용자은 Host의 front 웹서버를 통해 접속
- front는 msa-network의 gateway를 통해 각 서비스를 제공

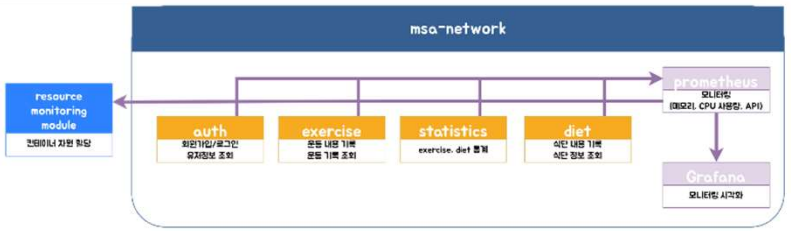
운영 영역 구조도 (1) - Resources Monitoring



- Grafana를 통해 자원을 시각적으로 확인

- Prometheus 컨테이너의 모니터링 정보를 활용하여 컨테이너 자원을 할당

운영 영역 구조도 (1) - Resources Monitoring

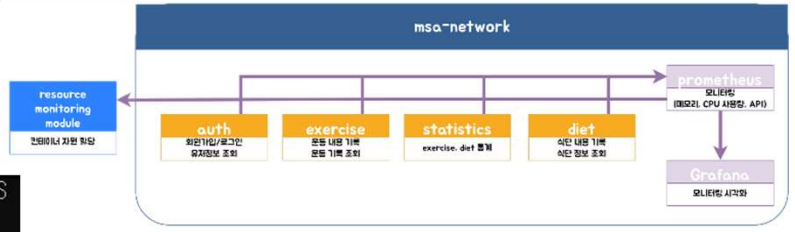


현재 컨테이너가 사용중인 메모리가 80%를 초과하면 5GB만큼 자원을 추가 할당한다.

반대로 현재 컨테이너가 사용중인 메모리가 20% 미만이면 5GB만큼 자원을 회수한다.

1. docker stats : 도커 자원 할당 확인
2. py DynamicAllocation.py : 동적 할당 모듈 실행
3. docker update --memory "1gb" --memory-swap "1gb" exercise : exercise 컨테이너 메모리 1gb로 낮추기

운영 영역 구조도 (1) - Resources Monitoring



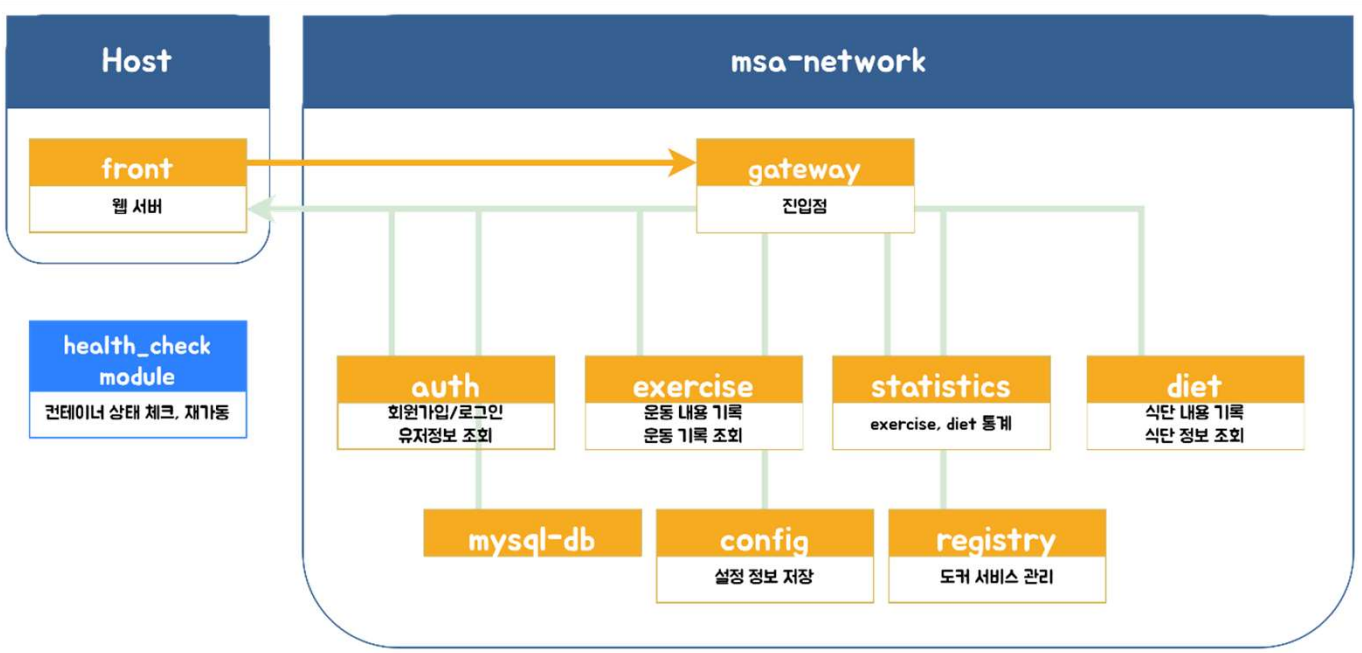
자원을 할당/회수 하기 전

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
c69a9935a45b	auth	2.22%	851.4MiB / 15.35GiB	5.42%	279kB / 1.19MB	0B / 0B	57
0c2333fbaa35	exercise	0.30%	873.9MiB / 15.35GiB	5.56%	277kB / 1.17MB	0B / 0B	59
856bcd517259	gateway	4.19%	872.2MiB / 15.35GiB	5.55%	2.71MB / 5.4MB	0B / 0B	99
ce221b54203c	diet	1.09%	856.2MiB / 15.35GiB	5.45%	268kB / 166kB	0B / 0B	55
aaa8b3896db1	statistics	4.42%	610.9MiB / 15.35GiB	3.89%	82.1kB / 54.2kB	0B / 0B	53
18ca98fd12ec	registry	3.39%	749.4MiB / 15.35GiB	4.77%	314kB / 222kB	0B / 0B	76
fb0edbbbf337	config	0.39%	538.3MiB / 15.35GiB	3.43%	112kB / 105kB	0B / 0B	43
3e124256e720	mysql-db	0.16%	260.3MiB / 15.35GiB	1.66%	11.4MB / 675kB	0B / 0B	78
3a534c51d6a2	prometheus	0.14%	62.88MiB / 15.35GiB	0.40%	5MB / 255kB	0B / 0B	22
80bbff3d9f56	grafana	0.02%	39.78MiB / 15.35GiB	0.25%	33.7kB / 32kB	0B / 0B	19
e9030cc6e027	front	0.00%	17.18MiB / 15.35GiB	0.11%	55.6kB / 0B	0B / 0B	17

자원을 할당/회수 한 후

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
c69a9935a45b	auth	0.21%	852.6MiB / 10GiB	8.33%	309kB / 1.43MB	0B / 0B	57
0c2333fbaa35	exercise	0.27%	875MiB / 10GiB	8.55%	307kB / 1.4MB	0B / 0B	59
856bcd517259	gateway	0.99%	876.9MiB / 10GiB	8.56%	3.23MB / 6.48MB	0B / 0B	99
ce221b54203c	diet	3.67%	858.1MiB / 10GiB	8.38%	297kB / 191kB	0B / 0B	55
aaa8b3896db1	statistics	0.32%	615.5MiB / 10GiB	6.01%	110kB / 79.9kB	0B / 0B	54
18ca98fd12ec	registry	2.45%	758.4MiB / 10GiB	7.41%	364kB / 260kB	0B / 0B	76
fb0edbbbf337	config	0.26%	539MiB / 10GiB	5.26%	121kB / 122kB	0B / 0B	43
3e124256e720	mysql-db	0.15%	263.1MiB / 10GiB	2.57%	11.4MB / 690kB	0B / 0B	78
3a534c51d6a2	prometheus	0.00%	67.91MiB / 10GiB	0.66%	6.02MB / 306kB	0B / 0B	22
80bbff3d9f56	grafana	0.26%	41.27MiB / 10GiB	0.40%	69.1kB / 79kB	0B / 0B	19
e9030cc6e027	front	0.00%	17.18MiB / 10GiB	0.17%	55.6kB / 0B	0B / 0B	17

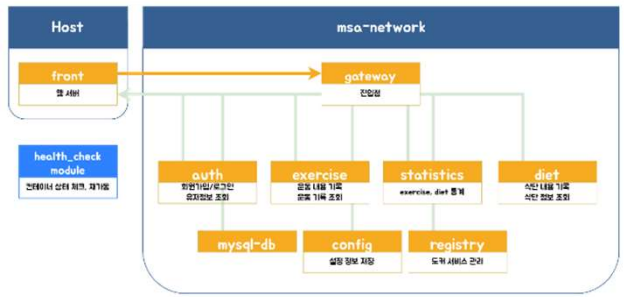
운영 영역 구조도 (2) - HealthCheck



- msa-network에 있는 컨테이너는 front컨테이너에 curl 요청을 통해 healthcheck

- health_check module을 통해 컨테이너들의 상태를 체크
  -> 이상이 생기면 재가동

운영 영역 구조도 (2) - HealthCheck



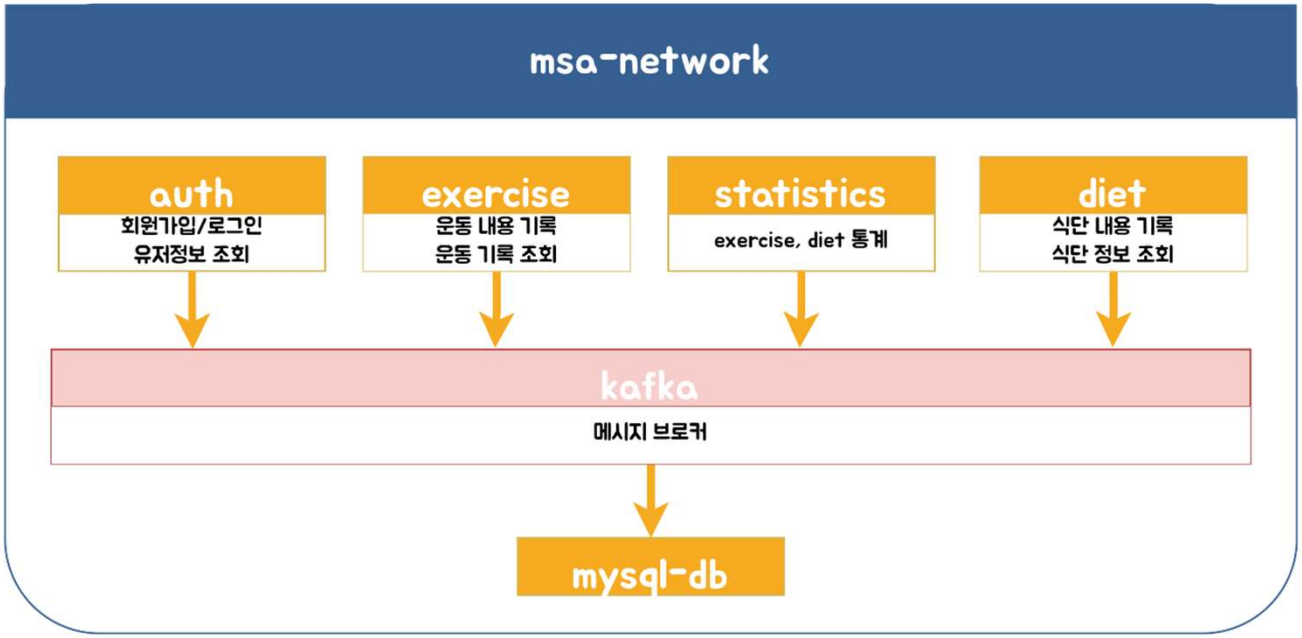
컨테이너가 종료된 상태

```
CONTAINER ID   IMAGE                  COMMAND                  CREATED          STATUS                      PORTS
     NAMES
eb5502b75fe2   qpdh1924/exercise      "java -XX:+UnlockExp…"   About an hour ago   Up 39 minutes (healthy)      0.0.0.0:8083->8080/tcp
    msa_exercise_1
cc9cbd6823bf   qpdh1924/statistics    "java -XX:+UnlockExp…"   About an hour ago   Up 39 minutes (healthy)      0.0.0.0:8084->8080/tcp
    msa_statistics_1
80fa6fc6a19a   qpdh1924/diet          "java -XX:+UnlockExp…"   About an hour ago   Up About an hour (healthy)   0.0.0.0:8082->8080/tcp
    msa_diet_1
6b9d001418ab   qpdh1924/auth          "java -XX:+UnlockExp…"   About an hour ago   Up About an hour (healthy)   0.0.0.0:8081->8080/tcp
    msa_auth_1
e1573b359a10   qpdh1924/gateway       "java -XX:+UnlockExp…"   About an hour ago   Up About an hour (healthy)   0.0.0.0:8080->8080/tcp
    msa_gateway_1
796a6e07e0b6   ce19f003/registry      "java -XX:+UnlockExp…"   About an hour ago   Up About an hour (healthy)   8080/tcp, 0.0.0.0:9761-
>8761/tcp   msa_registry_1
82c9752c94a3   qpdh1924/config-server "java -XX:+UnlockExp…"   About an hour ago   Up About an hour (healthy)   8080/tcp, 0.0.0.0:9888-
>8888/tcp   msa_config_1
f6eb5503506c   ce19f003/mysql:5.7     "docker-entrypoint.s…"   About an hour ago   Exited (0) 52 seconds ago
    mysql-db
```

결함 포용 모듈로 컨테이너를 다시 실행한 상태

```
CONTAINER ID   IMAGE                  COMMAND                  CREATED          STATUS                      PORTS
          NAMES
eb5502b75fe2   qpdh1924/exercise      "java -XX:+UnlockExp…"   About an hour ago   Up 40 minutes (healthy)      0.0.0.0:8083->8080/tcp
          msa_exercise_1
cc9cbd6823bf   qpdh1924/statistics    "java -XX:+UnlockExp…"   About an hour ago   Up 40 minutes (healthy)      0.0.0.0:8084->8080/tcp
          msa_statistics_1
80fa6fc6a19a   qpdh1924/diet          "java -XX:+UnlockExp…"   About an hour ago   Up About an hour (healthy)   0.0.0.0:8082->8080/tcp
          msa_diet_1
6b9d001418ab   qpdh1924/auth          "java -XX:+UnlockExp…"   About an hour ago   Up About an hour (healthy)   0.0.0.0:8081->8080/tcp
          msa_auth_1
e1573b359a10   qpdh1924/gateway       "java -XX:+UnlockExp…"   About an hour ago   Up About an hour (healthy)   0.0.0.0:8080->8080/tcp
          msa_gateway_1
796a6e07e0b6   ce19f003/registry      "java -XX:+UnlockExp…"   About an hour ago   Up About an hour (healthy)   8080/tcp, 0.0.0.0:9761-
>8761/tcp    msa_registry_1
82c9752c94a3   qpdh1924/config-server "java -XX:+UnlockExp…"   About an hour ago   Up About an hour (healthy)   8080/tcp, 0.0.0.0:9888-
>8888/tcp    msa_config_1
f6eb5503506c   ce19f003/mysql:5.7     "docker-entrypoint.s…"   About an hour ago   Up 3 seconds (health: starting)   0.0.0.0:3306->3306/tcp,
33060/tcp   mysql-db
```

운영 영역 구조도 (3) - Message Queuing Server

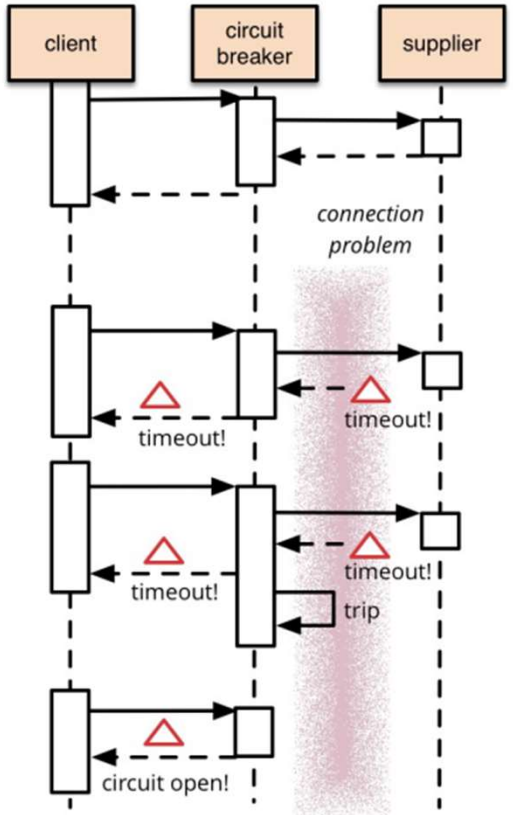


- 각 서비스가 mysql에 직접적으로 요청하는 것이 아닌

  메시지 큐를 통해 서비스가 이루어짐

- 중간에 서비스가 다운되어도 데이터베이스의 무결성을 보장

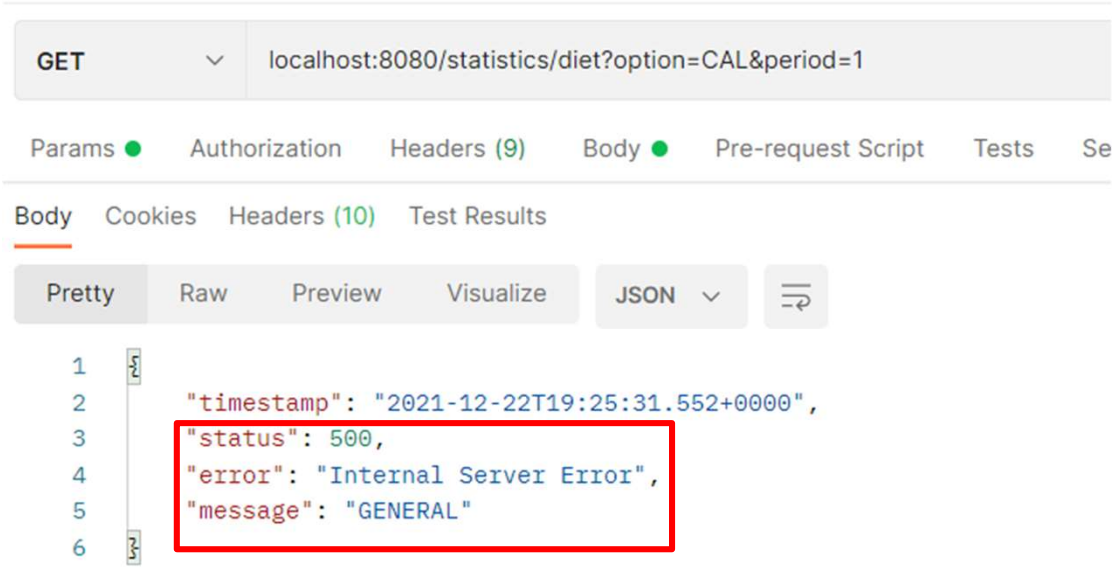
운영 영역 구조도 (4) - Circuit Breaker



- MicroSerive간에 API통신 시 서비스 하나가 Down될 에러가 하나의 서비스만 고립적으로 문제가 발생하는 것이 아닌 연쇄적으로 발생

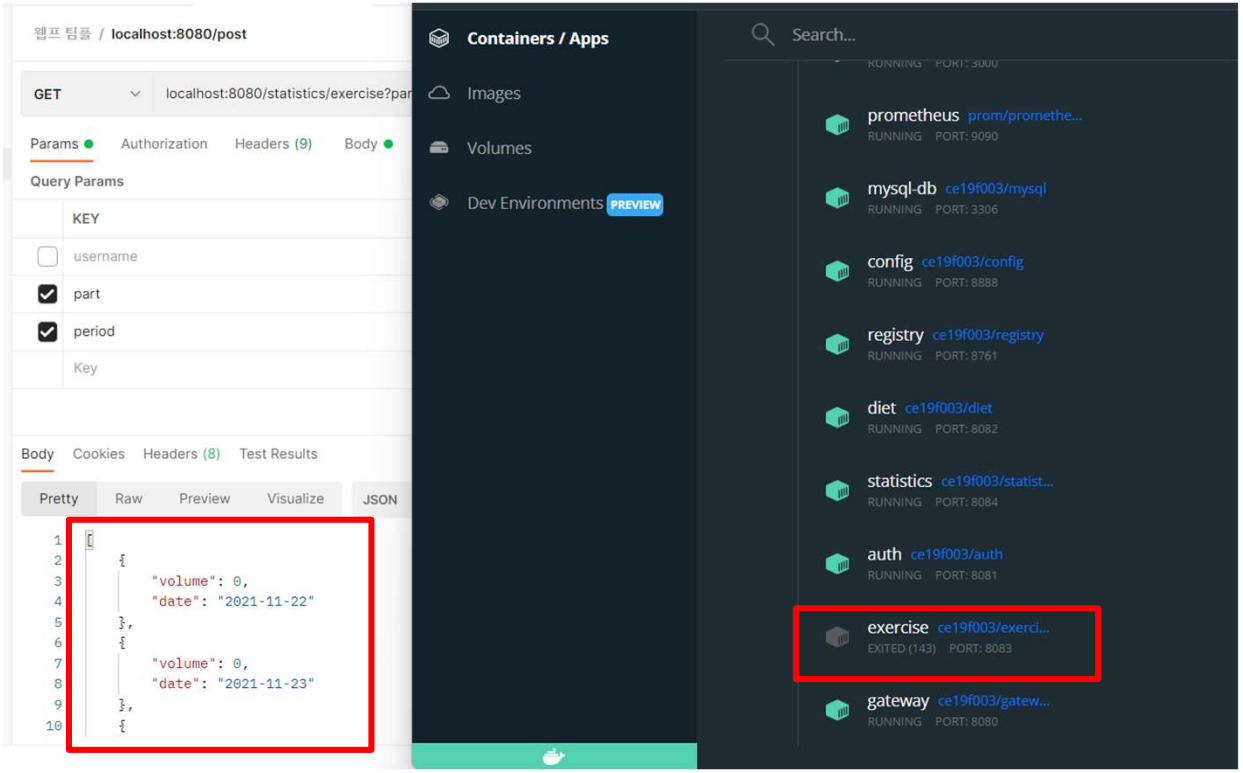
- 이런 서비스 장애를 고립시키기 위해 다운된 MicroService에 요청을 보내지 않고, Default값을 보내 정상 작동하는 것 처럼 보여준다.

운영 영역 구조도 (4) - Circuit Breaker



- circuitbreaker가 적용되지 않은 컨테이너가 종료되었을 때 요청 시 오류

운영 영역 구조도 (4) - Circuit Breaker



- exercise 컨테이너가 중지된 상태에서 postman으로 request 시 default 값이 나오는 상황

프로젝트 추진 일정

1주차 — 1 자료조사 — 2 주제선정 — 3 구조설계

2주차 — 4 개발환경구축 — 5 DB 구축

3주차

4주차

5주차

프로젝트 추친 일정

1주차

2주차

6 역할분담 7 msa 개발 — 3주차

8 시나리오 9 TEST — 4주차

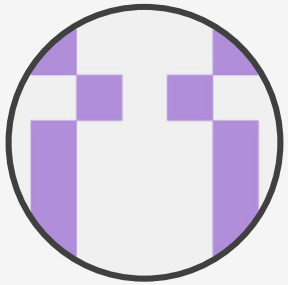
9 TEST &발표준비 — 5주차

최종 역할 분배

김동현

1. Notion 관리
2. HealthCheck를 통한 컨테이너 결함포용 문제
3. docker-compose : Init SQL를 통한 데이터베이스 뼈대 생성, 더미데이터 삽입
4. docker Network 설정

최종 역할 분배



김기현

1. kafka 연결을 통한 DB Query요청 안정화
2. Sprint micrometer를 통한 각 서비스의 api, 서버의 자원상태를 체크하는 log 수집해서 prometheus로 전송
3. resilience4j를 통한 circuitbreaker 설정
4. nginx 이미지 위로 front코드 컨테이너화

최종 역할 분배

송경진

1. 식단 데이터 타입 생성 및 컨트롤러에 식단 데이터 생성 함수 구현
2. 운동 데이터 타입 생성 및 컨트롤러에 운동 데이터 생성 함수 구현
3. 식단, 운동 추가 페이지 구현

최종 역할 분배

최혜민

1. 시나리오 1 구현
2. ALERT MANAGER를 이용한 EMAIL 전송

최종 역할 분배

허 민

1. 프론트엔드 서비스 분석
2. 기존 리액트 레거시 코드 DB 이슈 핸들링
3. 기존 리액트 컴포넌트 렌더링 이슈 핸들링
4. 사용자 기반 컴포넌트 UI 문서화

감사합니다.