MNC LLMOps (Lite)

운영자 메뉴얼

고객사: 롯데손해보험

Version: v 1.03.00LD

Doc. Version: v 1.2

2024.12

목차

[**문서 이력 3**](#_d240m38pqzov)

[**1. 시스템 구성 4**](#_5aufn3ftn2u9)

[1.1. MLOps 시스템 구성 이해 4](#_p3mw95ch4n4z)

[**2. DEV (모델개발, 학습 환경) 6**](#_rbi6m5t8n7cx)

[2.1. Jupyter 기능 시스템 아키텍처 6](#_rg11m35iw2ap)

[**3. Service (모델 서빙, 배치 실행 환경) 7**](#_r3grimz5x26m)

[3.1. API Service 기능 시스템 아키텍처 7](#_88vt6n7ygjqc)

[3.2. Batch Service 기능 시스템 아키텍처 8](#_4tja43osw3uj)

[**4. 시스템 장애 증상 별 대처 방안 9**](#_c4f01qvzeogx)

[4.1. 시스템 재기동 직후, 혹은 MLOps 일부 기능이 작동하지 않는 경우 9](#_iv0ro4nskxg2)

[4.2. 솔루션 접속 시 페이지를 찾을 수 없는 경우 10](#_1hx3gh9ic88g)

[4.3. MLOps Admin WEB 접속 불가능, Login 페이지가 안 뜨는 경우 11](#_ba5zkvcpeb6v)

[4.4. MLOps Admin WEB 접속 가능, 로그인이 안 되는 경우 12](#_drhedvhdkbg6)

[4.5. 시스템 모니터링 대시보드가 안 뜨는 경우 12](#_algrw9b77vu3)

[4.6. 배치 관련 기능이 작동하지 않는 경우 13](#_anfw6uafl5j0)

[4.7. Jupyter, API / 배치 서비스가 실행되지 않는 경우 14](#_gulxwfmwd01t)

[4.8. Dev 환경, API Service 혹은 Batch Service 구동 불가 15](#_yupi8hblh0aq)

[4.9. VS Code Jupyter Extension 동작 안 하는 경우 20](#_n6yyumecwys6)

# 

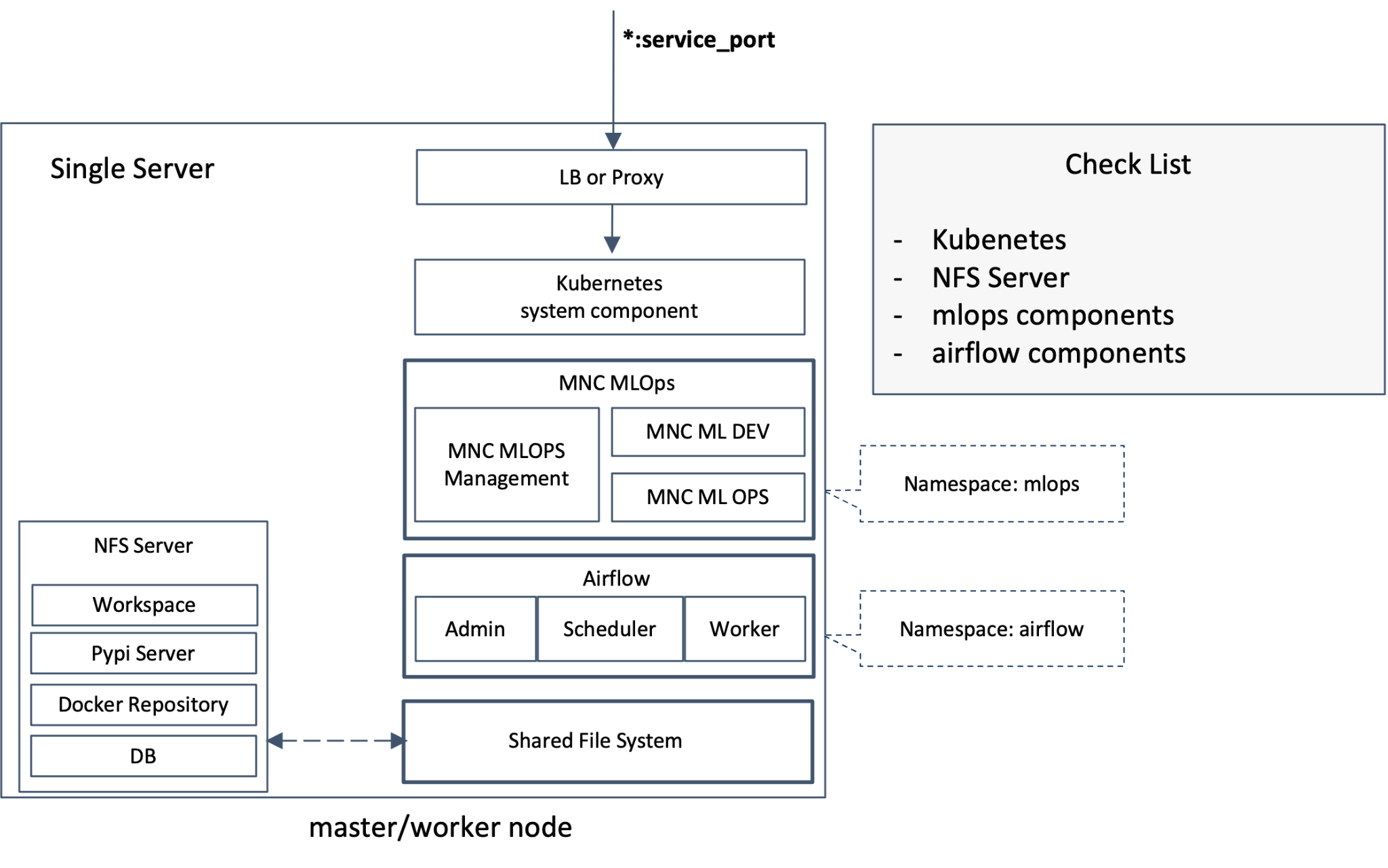
# 문서 이력

| **문서버전** | **개정 일자** | **작업자** | **설명** |
| --- | --- | --- | --- |
| v1.0 | 2024.12.12 | 임태수 | MLOps V1.03.00LD Release |
| v1.1 | 2024.12.23 | 임태수 | 장애대응 상황 추가 |
| v1.2 | 2024.12.24 | 함명호 | 알람서비스, 로깅 추가 |
|  |  |  |  |
|  |  |  |  |

# 시스템 구성

## MLOps 시스템 구성 이해

MNC MLOps 솔루션은 여러 개의 모듈 단위로 구성되어 있으며, 이 중 어느 한 모듈이라도 문제가 발생할 경우 솔루션이 정상 동작하지 않을 수 있다. 단일 서버에 설치된 경우 아래 그림과 같은 주요 계층으로 구성되며, 시스템 제기동 시 혹은 이상 발생 시 아래 체크 리스트에 따라 확인하도록 한다.

[그림] MLOps 시스템 구성도

주요 구성 요소와 확인 순서는 다음과 같다.

| 구성 요소 | 설명 |
| --- | --- |
| Kubernetes | mlops 는 kubernetes cluster상에 설치, 작동하도록 구성되어 있으며, 단일서버 구성의 경우도 동일하다. |
| NFS Server (NAS) | 모델 학습등을 위한 개인영역부터 mlops 구동을 위한 공유 파일을 관리한다. |
| mlops components | mlops를 구성하는 pods 및 volume 그리고 network 요소등이 포함된다. |
| airflow components | mlops에서 제공하는 스케쥴링 관련 기능을 담당한다. |

만약 솔루션 사용 중 이상이 발생한 경우, 어느 모듈에서 이상이 발생했는지를 파악하고 대처해야 한다.

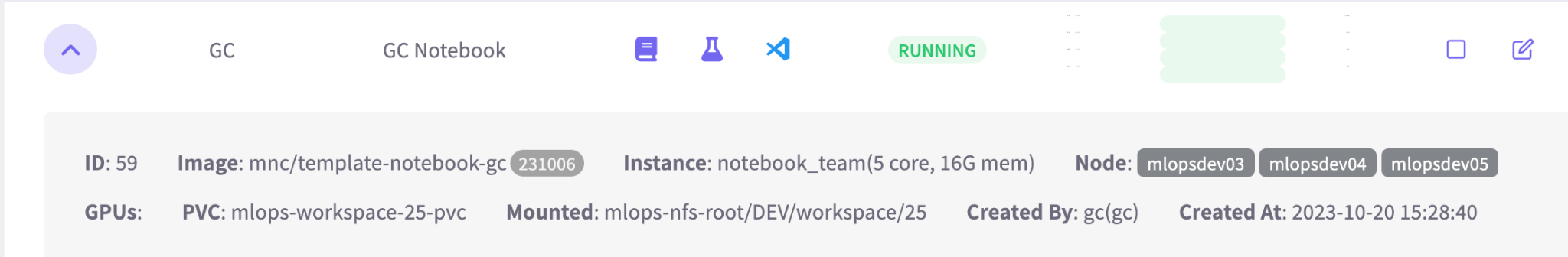
| ※ 주의  MNC MLOps는 Kubernetes 클러스터 상에서 구동하고, kubernetes는 호스트 시스템 재시작 후 구동 중인 컨테이너를 자동으로 재시작하도록 되어 있습니다. 다만, 일부 모듈의 경우 재시작 후 재설정을 요하기 때문에 본 문서에 명시된 방법으로 원인 규명이 어려운 경우 MNC에 문의 바랍니다. |
| --- |

# 

# DEV (모델개발, 학습 환경)

## Jupyter 기능 시스템 아키텍처

MLOps 솔루션의 Dev 기능은 NFS 내 특정 경로 상에 생성된 Jupyter Notebook 을 팀 단위 (Group) 로 접근 / 관리하는 기능이다.



[그림] MLOps Dev Jupyter 실행 화면 예시

MLOps 솔루션 상에서 Notebook 을 실행하면 ML Framework 및 개발환경이 구성되어 있는 컨테이너가 실행되며, 해당 컨테이너는 NFS 상의 특정 경로를 mount 하여 데이터를 영구 보존한다.



현재 솔루션 상에서 제공되는 개발환경은 Notebook, JupyterLab, VSCode 가 있으며, 솔루션에 탑재되어 있는 MLFlow 를 사용하기 위한 mlflow python library가 동봉되어 있다.

# Service (모델 서빙, 배치 실행 환경)

## API Service 기능 시스템 아키텍처

DEV 에서 생성된 모델을 서비스화 하여 외부 서비스와 연동할수 있도록 하는것을 서빙이라고 칭한다.

MLOps가 모델을 서빙하기 위하여 다음과 같은 기능을 제공한다.

* http를 통한 RESTful 통신
* json, image 전송 지원
* 서비스를 위한 커스텀 코드 지원
* 고성능 서빙을 위한 TFServer, Nvidia-Triton Server, MNC 자체 서버 지원
* 한개 서비스 내 여러개 모델 인퍼런스 구조 지원
* 모니터링을 위한 input/output 데이터 수집/감시

Model Serving Container의 기본 구조는 다음과 같다.



[그림] 모델 서비스 컨테이너 구성도

## Batch Service 기능 시스템 아키텍처

배치는 내부적으로 Airflow를 Scheduler로 채택하고 있으며, Airflow 버전 및 정보는 아래와 같다.

Airflow Specification

* Airflow Helm Chart 1.15.0
  + <https://airflow.apache.org/docs/helm-chart/1.15.0/index.html>
* Airflow 2.9.3
  + <https://airflow.apache.org/docs/apache-airflow/2.9.3/index.html>

배치 스케줄은 crontab 형태를 지원하고 있다. 배치 서비스는 내부적으로 Airflow DAG를 이용하여 Serving Container를 구동하는 형태로 동작하며, Airflow를 알고 있을 필요는 없지만 Airflow 동작 방식을 안다면 배치 서비스 구동 방식 이해에 도움이 된다.

배치 서비스의 동작 방식을 간결화하면 아래와 같다.



[그림] 모델 서비스 컨테이너 구성도

# 

# 알람 서비스

# 알람 서비스 아키텍처

알람 서비스는 쿠버네티스의 CronJob 오브젝트를 이용하며, 현재 5분마다 실행되어 서비스중인 API 서비스를 대상으로 healthcheck 요청을 보내고, 만약 응답이 없을시 장애로 간주하고 쪽지와 이메일로 장애 리포트를 전달한다.

알람 서비스를 구동하기 위한 파일은 2개로 구성되며, PDAMAP01 10.150.4.85 서버 접속후 /app/pkg 경로의 mlops-alarm.yaml, mlops-alarm-configmap.yaml 파일을 확인하면 된다.

mlops-alarm.yaml 파일은 CronJob을 구성하기 위한 파일이고, schedule: “\*/5 \* \* \* \*” 로 초기 세팅이 되어 있고, 5분에 한번씩 health check를 보내도록 되어 있다.

health check를 위한 프로그램은 mlops-alarm-configmap.yaml에 기술되어 있다. 이중 data 변수는 쪽지를 보내기 위한 파라미터이고, data\_mail은 메일을 보내기 위한 파라미터이다. 각각 수신자, 보내는 사람, 제목 등을 기술하고 있으며, 변경이 필요한 경우 이 파일을 수정하면 된다.

* 1. 알람 서비스 시작, 중지

PDAMAP01 10.150.4.85에 ssh 접속을 하여 다음의 명령을 실행하면 mlops-alarm pod의 마지막 실행 상태를 확인할수 있다.

| $ kubectl -n mlops get pods | grep alarm |
| --- |

해당 서버의 /app/pkg/ 폴더로 이동해서 알람 서비스의 중지를 위해서는 다음 명령을 실행하면 된다.

| $ kubectl delete -f mlops-alarm.yaml |
| --- |

그리고 알람 서비스를 시작하기 위해서 다음 명령을 실행하면 된다.

| $ kubectl apply -f mlops-alarm.yaml |
| --- |

* 1. 알람 서비스 설정 변경

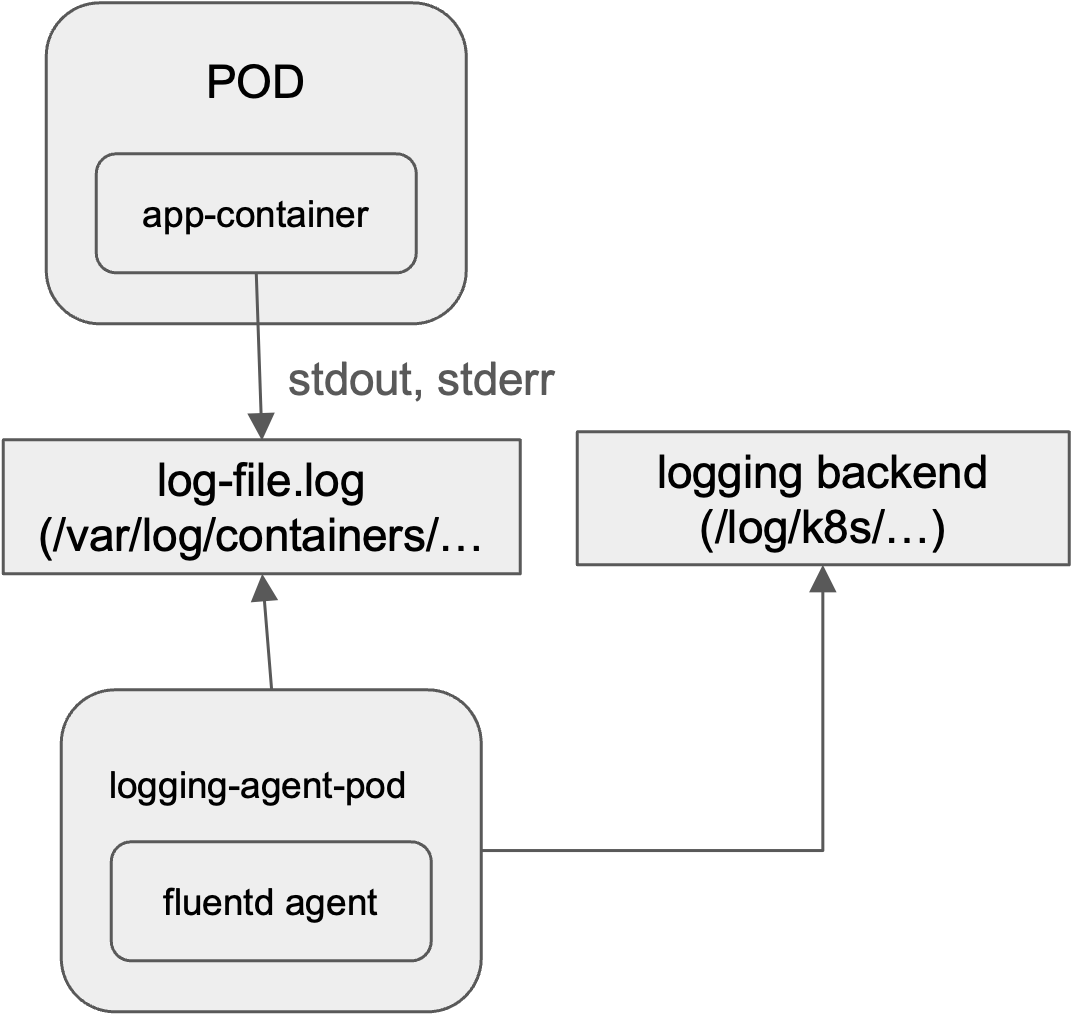
만약 수신자, 보내는 사람, 제목등의 변경이 필요하다면 mlops-alarm-configmap.yaml 의 변경이 필요한데, 변경후 아래의 명령을 실행하면 변경사항이 적용된다.

| $ kubectl apply -f mlops-alarm-configmap.yaml |
| --- |

# 로깅

# 로깅 아키텍처

fluentd는 오픈소스 로그 수집기이다. 서버에 쌓이고 있는 log file을 지정하여 수집해 로그 저장소에 최종적으로 저장한다. 다양한 로그 저장소를 지원하지만 본 프로젝트에서는 아래 그림과 같이 각 서버의 /log/k8s 경로에 저장을 한다.



pod의 로그는 /var/log/containers/ 에 쌓이게 되고, fluentd는 해당 경로의 log 파일을 수집하도록 설정되어있고, 로깅 backend는 /log/k8s/ 가 된다. 로깅 backend의 로그파일은 fluentd.log이고, 하루 경과시마다 압축 저장을 해놓는다.

# 로그 확인

오늘 발생한 로그는 해당 서버의 /log/k8s/fluentd.log를 확인하면 실시간으로 쌓이는 로그를 확인할수 있다. 과거의 로그는 날짜별로 압축되어 있어서 압축 해제후 확인을 하면 된다.

# 

# 시스템 장애 증상 별 대처 방안

## 시스템 재기동 직후, 혹은 MLOps 일부 기능이 작동하지 않는 경우

아래 명시된 증상, 혹은 그와 유사하게 솔루션 접속은 가능하나 일부 기능이 동작하지 않는 경우 MLOps 구성 요소의 상태를 확인한다.

* 모델 모니터링 대시보드 페이지 및 Data Health 그래프 표시 오류
* API / 배치 서비스 실시간 로그 표시 오류
* Jupyter, API 서비스의 컨테이너 별 리소스 사용량 표시 오류
* 로그인 불가

위와 같은 MLOps 일부 기능의 동작 오류의 경우, 대부분 MLOps 내 일부 모듈이 정상 동작하지 않는 상황이다. 이 때, 해당 모듈을 재구동 시키는 방법으로 긴급 조치를 할 수 있다.

MLOps 구동에 가장 기본이 되는 MLOps 모듈은 mlops namespace 상에 구축되어 있다. 아래 명령어를 통해 mlops namespace 내 모든 모듈의 구동 상태를 확인할 수 있다.

| $ kubectl -n mlops get deployments |
| --- |

[그림 x - 8] mlops namespace 내 패키지 (Deployment) 목록 일람 (예시)

\* 일부 모듈은 패키지 구성에 따라 설치되지 않았을 수 있다.

\*\* 붉은 테두리로 표시된 부분은 UI에서 실행한 Notebook, Service 컨테이너 명 예시이다.

만약 READY 상태가 1/1 이 아닌 모듈이 있다면 아직 로딩 중이거나 정상 구동이 되지 않은 것이며, 해당 상태가 지속될 경우 시스템 작업을 필요로 한다.

MLOps 기본 모듈 상태 조회를 조회하는 또 다른 방법으로는 POD 조회가 있다. 아래 명령어를 통해 현재 실행 중인 Kubernetes POD 목록을 확인할 수 있으며, Running 상태가 아닌 경우 별도의 시스템적 조치가 필요할 수 있다.

| $ kubectl -n mlops get pods |
| --- |

[그림 x - 9] mlops namespace 내 POD 목록 일람 (예시)

\* 일부 모듈은 패키지 구성에 따라 설치되지 않았을 수 있다.

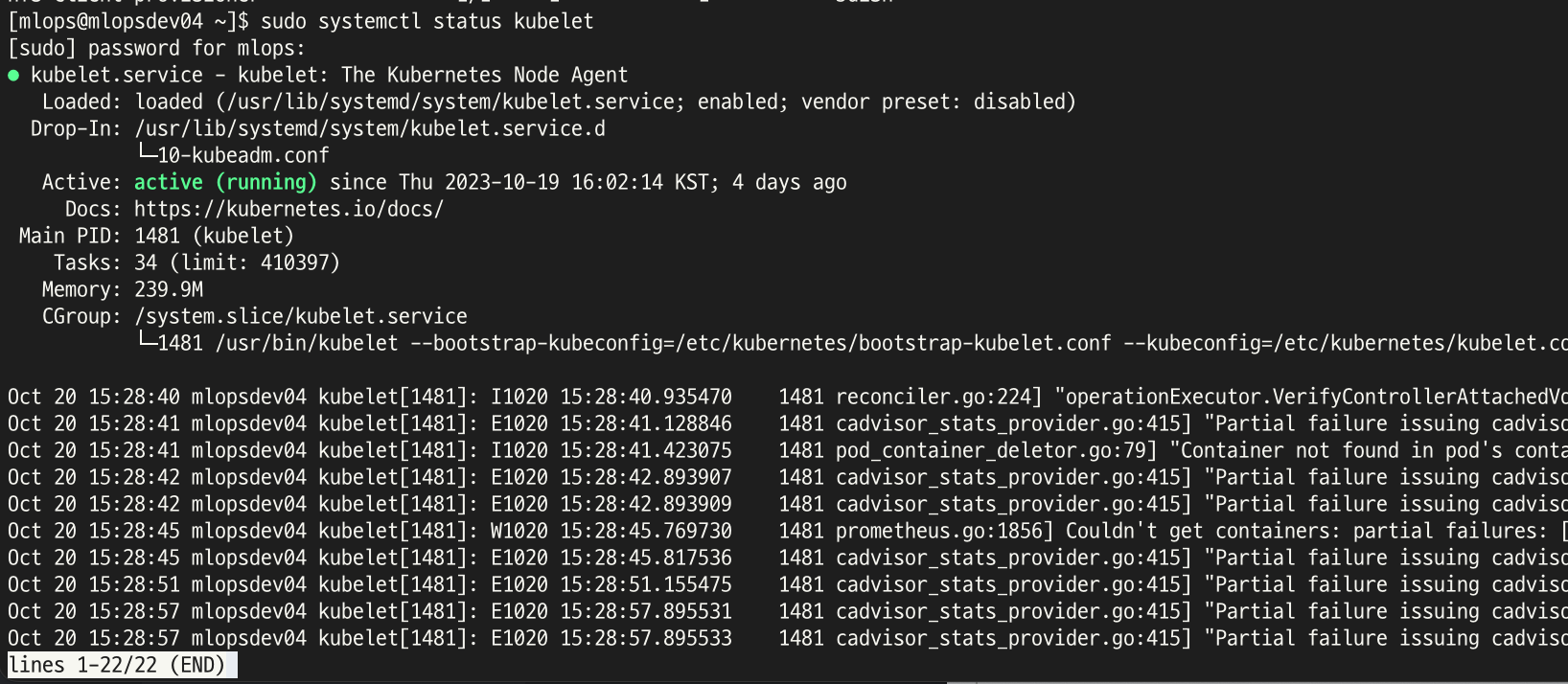
\*\* 붉은 테두리로 표시된 부분은 UI에서 실행한 Notebook, Service 컨테이너 명 예시이다.

## 솔루션 접속 시 페이지를 찾을 수 없는 경우

솔루션 URL에 접근했을 때 로그인 화면이나 404에러 화면 등이 표시되지 않고 페이지를 찾을 수 없는 경우, Kubernetes 클러스터 자체의 문제일 수 있다.

이를 쉽게 진단해보기 위한 방법으로 kubelet 서비스 상태 조회가 있다.

| $ sudo systemctl status kubelet |
| --- |

[그림 x - 13] kubelet 서비스 상태 조회

Kubernetes 클러스터 자체의 정상 동작 확인 방안으로는 kubelet 서비스 조회가 있다. (sudo 권한 필요)

만약 kubelet 서비스가 정상 동작하지 않는 경우 인프라 작업이 필요하며, 시스템 관리자에게 문의해야 한다. kubelet을 함부로 재시작할 경우, Kubernetes 클러스터 전체에 문제가 생길 수 있으므로, 시스템 관리자가 아니라면 관련 인프라 작업은 하지 않도록 한다.

## MLOps Admin WEB 접속 불가능, Login 페이지가 안 뜨는 경우

사용자 환경으로부터 MLOps Server의 Admin Web으로 접속이 가능한 환경인지 방화벽 등을 확인한다.

* 단일서버 환경의 경우

목적지 IP: MLOps Server IP

목적지 포트 : 30080

* HA Cluster의 경우

목적지 IP: HA 구성 Load Balancer의 IP 혹은 VIP

목적지 포트: 30080

**\*\* 설치 시 설정된 목적지 포트를 이용하여 확인 하여야 함.**

## MLOps Admin WEB 접속 가능, 로그인이 안 되는 경우

로그인 화면에서 로그인이 되지 않는 경우, 사용자 계성 설정 옵션 중 IP whitelist가 활성화 되었는지 확인하고 활성화 된 경우 설정 IP와 사용자 환경 IP를 확인한다. (admin 권한 접속 필요)

Admin 권한 등으로 접속 또한 불가한 경우, mlops 구성요소 확인이 필요하다.

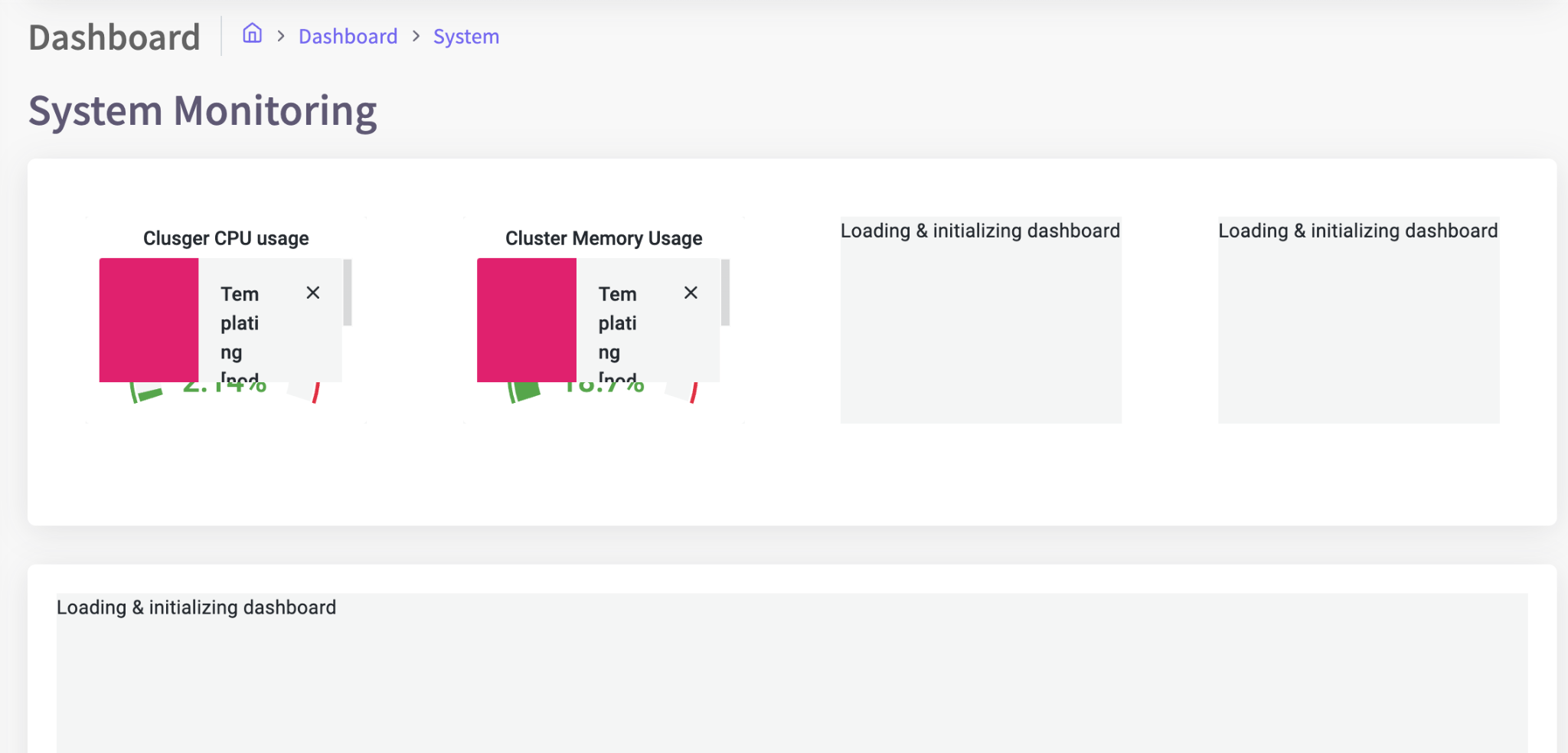
본 문서 내 [1.2.1. 시스템 재기동 직후, 혹은 MLOps 일부 기능이 작동하지 않는 경우] 를 참고하여 mlops 구성 요소들의 정상 구동 여부를 확인한다.

사내 SSO와 계정이 통합되어 있는 경우, 플랫폼 내에서 설정한 비밀번호가 아닌 SSO 시스템 상의 비밀번호를 입력해야 한다.

## 시스템 모니터링 대시보드가 안 뜨는 경우

시스템 모니터링 대시보드의 각 컴포넌트의 로딩이 끝나지 않거나 아래 스크린샷과 같이 오류 화면이 나타날 경우, 시스템 모니터링 대시보드에 값을 보내주는 모듈이 시스템 재기동 후 초기화되었을 가능성이 크다.

이 경우, 해당 모듈을 재설정해줘야 하는데, 이는 수동으로만 가능하므로 MNC에 문의해야 한다.



[그림 x - 24] 시스템 모니터링 대시보드 동작 오류 화면

## 배치 관련 기능이 작동하지 않는 경우

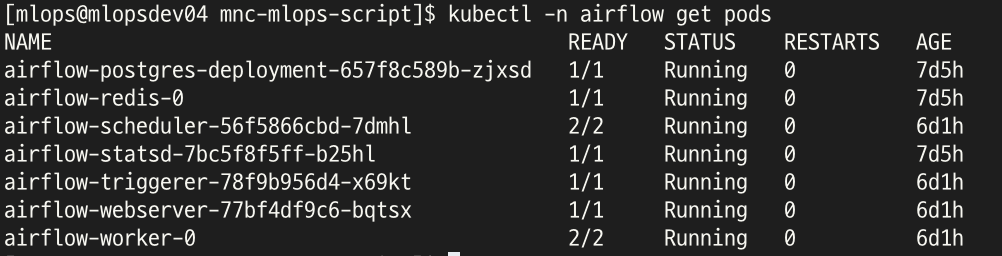
아래 명시된 예시 혹은 그와 유사하게 배치 관련 기능이 정상 동작하지 않는 경우, Airflow 모듈의 장애를 짐작해볼 수 있다.

* 배치 파이프라인에 등록된 DAG가 작동하지 않는 경우.
* 배치 서비스가 스케줄 된 시간에 작동하지 않는 경우.
* 배치의 실행 이력이나 최근 100개 실행 기록 등이 표시되지 않는 경우

Airflow는 MLOps 내 Batch Scheduler 및 Executor를 담당하는 중요 모듈이다. 이미 검증된 Apache 재단의 제품이며, 별도로 패키징되어 있기 때문에 다른 namespace에 설치되어 있다.

아래 명령어를 통해 현재 구동 중인 airflow namespace의 모듈 목록을 확인할 수 있다.

| $ kubectl -n airflow get pods |
| --- |

[그림 x - 10] airflow namespace 내 POD 목록 일람 (예시)

위에 표기된 POD들 중 STATUS가 Running이 아닌 POD가 있다면 Airflow가 정상 동작하지 않는다. 배치 스케줄러에 문제가 생겼을 가능성이 크기 때문에 시스템 관리자, 혹은 MNC에 문의해야 한다.

## Jupyter, API / 배치 서비스가 실행되지 않는 경우

Jupyter, API / 배치 서비스 등 실행 시 새로운 컨테이너가 생성되는 기능이 동작하지 않는 경우, 정확한 문제 원인을 위해서는 솔루션 전체 구성에 대한 이해가 필요하다. 하지만, 그 중에서 발생 빈도가 높고 쉽게 대처할 수 있는 몇 가지 원인 들이 있다.

가장 쉽게 확인해볼 수 있는 원인은 파일시스템 (NFS) 용량이다.



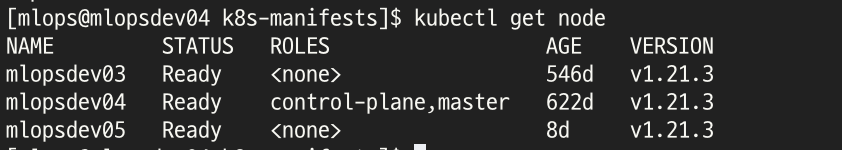
[그림 x - 11] MLOps 시스템 모니터링 대시보드 상 사용 중인 파일시스템 사용량

Dashboard > System Monitoring 메뉴에 들어가면 클러스터 파일시스템 사용량을 확인할 수 있다. Kubernetes는 특정 Node의 디스크 사용량이 85%를 넘어갈 경우, 해당 Node에 떠 있는 모든 POD를 Evict하고 새로운 POD 실행을 막아버린다. 때문에 디스크 사용량을 85% 미만으로 잘 관리해야 한다.

다만, MLOps 솔루션 상에서는 실제 사용 중인 파일시스템 용량만 표시되지만, Kubernetes 클러스터의 실제 Disk Pressure 상태를 조회하고 싶은 경우, 아래 명령어를 통해 확인할 수 있다.

노드 이름 확인

| $ kubectl get node |
| --- |

[그림 x - 12] Kubernetes 클러스터 내 노드 목록

노드 목록에서 노드 이름 확인 후, 아래 명령어를 통해 해당 노드의 정보를 확인할 수 있다.

| $ kubectl describe node [nodename] |
| --- |

Kubernetes 클러스터가 Disk Pressure 상태인 경우, 아래 그림처럼 DiskPressure Status가 True로 표시된다. 만약 현재 클러스터 상태가 아래 그림과 같이 Disk Pressure 상태일 경우 모든 POD가 동작하지 않을 가능성이 있으니 NFS가 실행되고 있는 디스크의 용량을 확보해야 한다.

[그림 x - 12] 실제 Disk Pressure 상태인 Kubernetes Node (예시)

## Dev 환경, API Service 혹은 Batch Service 구동 불가

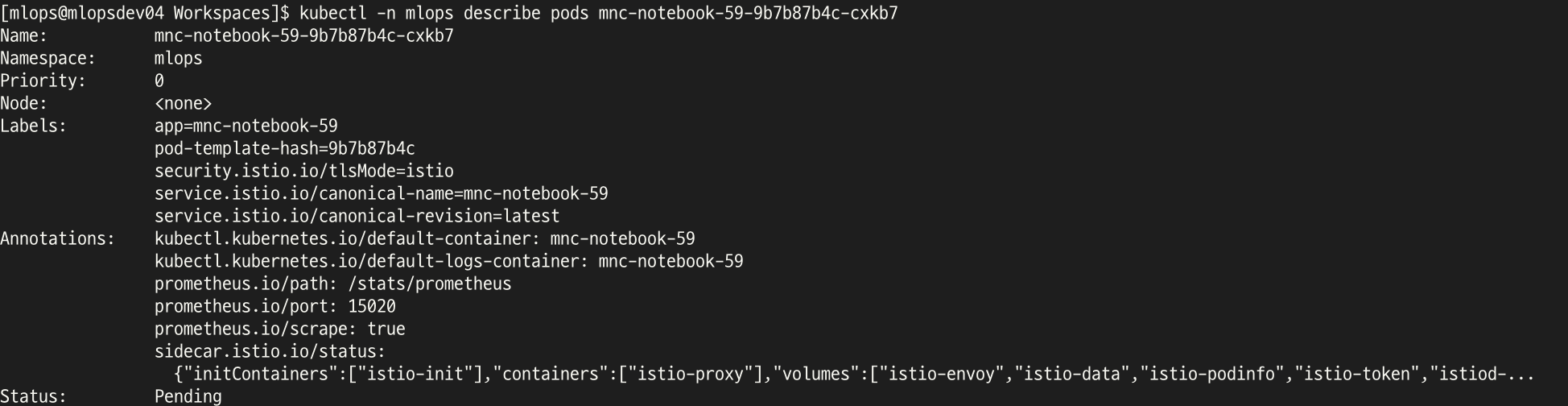
기존 사용하던 Dev (Jupyter) 환경, 혹은 신규로 구성하고자 하는 Jupyter의 구동이 불가한 경우, 또 기존 서비스 혹은 신규 API / 배치 서비스 구성 후 서비스 실행 시 구동이 불가한 상황이 있다. 위 기능은 모두 새로운 컨테이너를 생성하는 절차가 필요한데, 이 과정에서 컨테이너 생성이 불가한 경우가 있다.

아래 명령어를 통해 현재 정상 구동되고 있지 않은 POD와 원인을 파악할 수 있다.

| $ kubectl -n mlops get pods  $ kubectl -n mlops describe pod [podname] |
| --- |

상단에 있는 명령어로 확인한 POD 목록 중 STATUS가 READY가 아닌 POD의 상태를 하단의 명령어로 확인할 수 있다.

[그림 x - 14] 정상 구동중이 아닌 POD 확인 화면

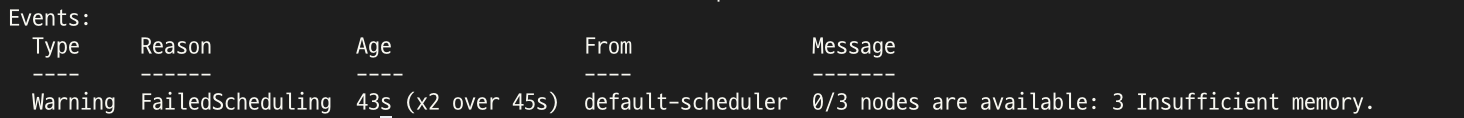
[그림 x - 15] POD 상태 확인 화면

* 시스템 내 리소스 부족

가장 가능성이 높은 원인은 현재 시스템의 리소스가 서비스의 컨테이너에서 요청한 만큼 충분하지 않은 경우이다. 서비스 구동 시 서비스 구성 정보 중 리소스 설정 값을 Kubernetes 클러스터에 요청하도록 되어 있는데, 리소스가 충분하지 않은 경우 Kubernetes에서는 컨테이너를 생성하지 않는다.

아래 명령어로 POD의 상태를 확인했을 때 Resource 부족으로 인한 실패 알림이 뜬다면 POD가 요청한 리소스를 확보할 수 없어서 실행이 되지 않은 것이다.

| $ kubectl -n mlops describe pod [podname] |
| --- |



[그림 x - 16] Memory 부족으로 인한 POD 실행 실패 화면

CPU, 메모리 등 시스템 리소스 부족으로 인한 POD 실행 실패의 경우, 리소스를 점거 중인 다른 서비스나 외부 프로그램 사용을 중지한 후 다시 시도한다.

Dev 환경, 서비스에 할당해야 하는 시스템 리소스는 아래 화면에서 확인 가능하다.

Instance Type은 CPU와 메모리 할당량을 표시한다.

GPU의 경우 할당한 GPU의 Index가 표기되며, 어떤 노드의 몇 번 GPU인지를 확인할 수 있다.



[그림 x - 17] Dev Jupyter에서 할당해야 하는 리소스 확인



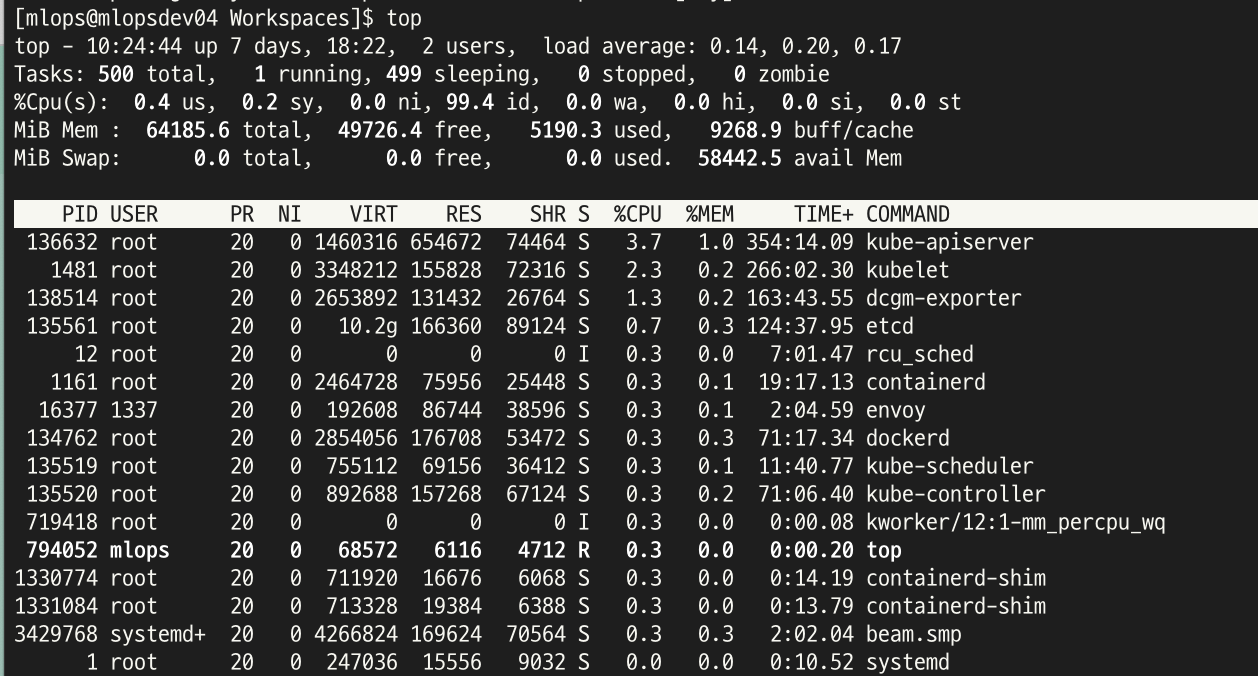
[그림 x - 18] API Service에서 할당해야 하는 리소스 확인

필요한 시스템 리소스가 확인되었다면, 부족한 리소스를 확인하고 조치를 취해야 한다.

.

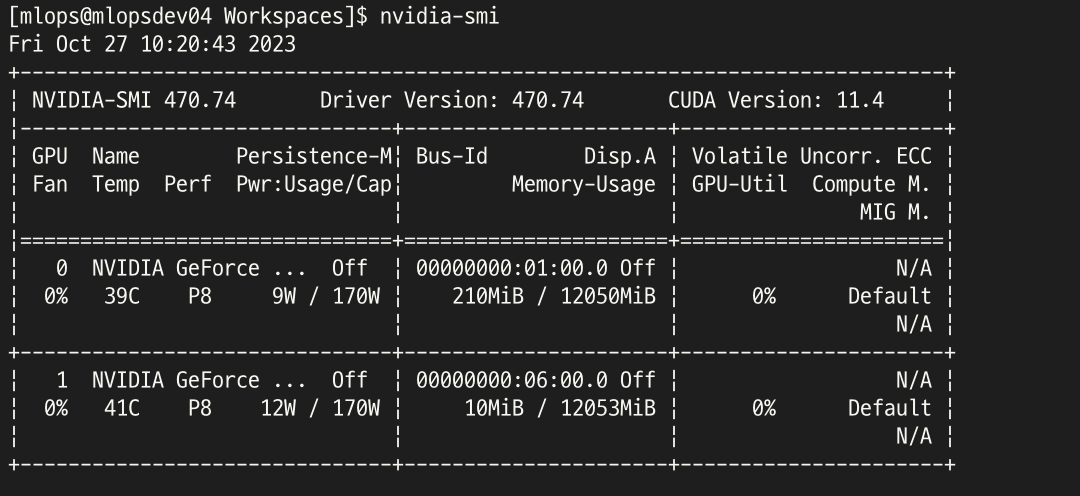
* 프로세스 별 CPU, 메모리 사용량

| $ top |
| --- |

[그림 x - 19] top을 통해 확인한 프로세스 별 CPU, Memory 사용 현황

* 시스템 내 GPU 및 GPU 메모리 사용량 확인

| $ nvidia-smi |
| --- |

[그림 x - 20] nvidia-smi를 통해 확인한 GPU 및 GPU 메모리 사용 현황

* 서비스 도커 이미지 미존재

시스템 내 로드되어 있는 도커 이미지는 원래 사라지지 않는다. 하지만, 디스크 사용량이 90% 이상일 경우, Kubernetes는 해당 Node 내의 도커 이미지들을 삭제하여 용량을 확보한다. 때문에 따로 도커 이미지를 삭제하지 않았더라도 도커 이미지가 사라질 수 있다.

Dev 환경, API / 배치 서비스에서 사용 중인 도커 이미지 이름은 아래 화면에서 확인할 수 있다.



[그림 x - 21] Dev Jupyter에서 사용 중인 도커 이미지 이름 확인

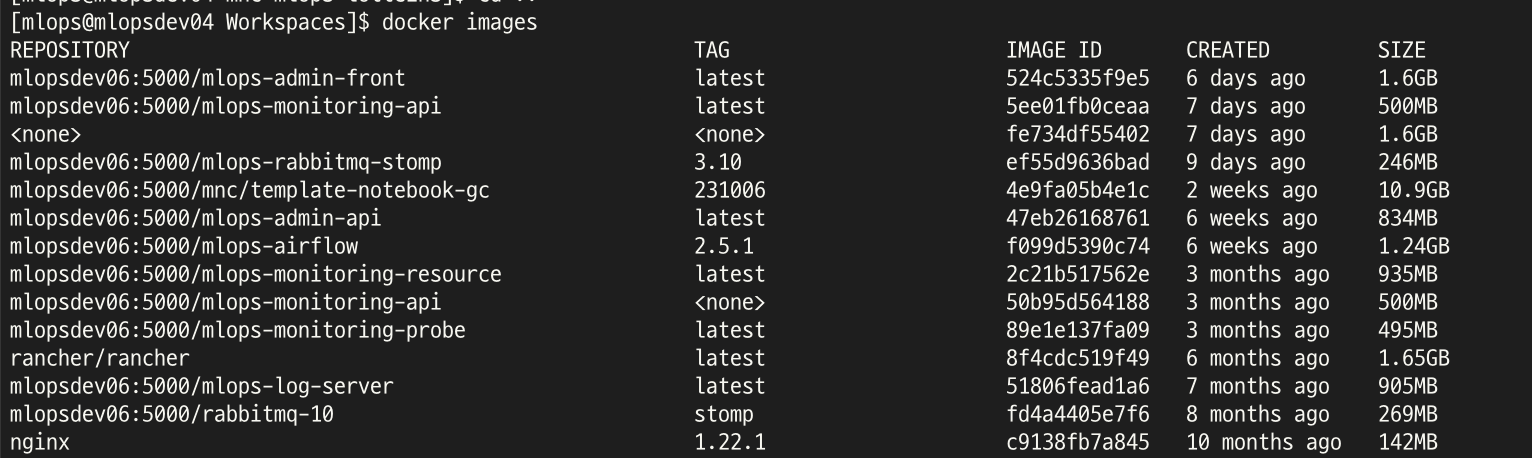


[그림 x - 22] API Service에서 사용 중인 도커 이미지 이름 확인

해당 이미지 이름을 확인했다면, 이제 시스템 내에 로드되어 있는 도커 이미지를 확인해야 한다. 아래 명령어로 확인 가능하며, 만약 도커 이미지가 정말로 존재하지 않는다면 MNC에 문의해야 한다.

| $ docker images |
| --- |

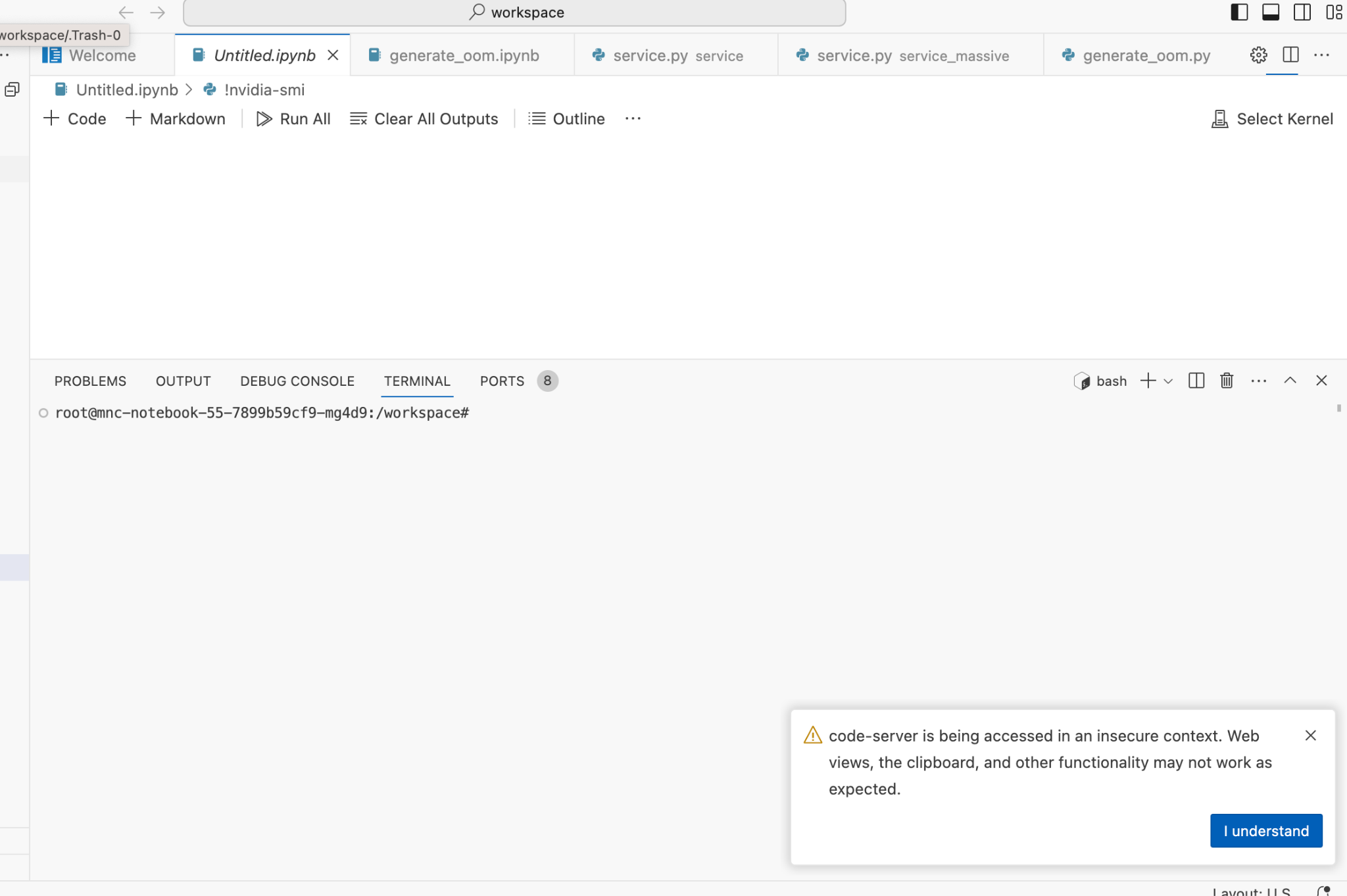
\*\* 현재 user가 docker 그룹 내에 없다면 sudo 권한이 필요할 수 있다.



[그림 x - 23] 시스템 내 도커 이미지 목록 (예시)

## VS Code Jupyter Extension 동작 안 하는 경우

MLOps 상에서 VSCode 실행 시 파일 조회, 터미널 등 모두 정상 동작하고 ipynb Notebook 파일의 cell만 표시가 안 되는 경우가 있다. 이는 Code Server가 HTTPS가 아닌 HTTP로 접속했기 때문에 발생하는 문제인데, jupyter extension이 정상 동작하지 않는 것이다.

[그림 x - 25] VSCode Jupyter Extension 동작 오류

물론 HTTPS로 접속하는 방법도 있지만, 당장 작업이 급하거나 시스템 작업을 하기에 여의치 않은 경우, 브라우저 설정을 통해 해결할 수 있다.

[그림 x - 26] Chrome Insecure Origin 설정 창

위 그림과 같이 chrome://flags 로 크롬 플래그 설정 창에 진입할 수 있다. 그 중 <Insecure origins treated as secure> 플래그를 Enabled로 설정 변경 하고, 현재 mlops가 실행 중인 시스템 접근에 사용하는 호스트명, DNS, IP주소 등을 입력하고 저장한 뒤 크롬을 재실행하면 된다.

이는 같은 브라우저 엔진을 사용하는 MS Edge에서도 동일하다. (edge의 경우 edge://flags로 진입할 수 있다.)