

MetaMedical

하이브리드 기반 보건 행정 서비스 클라우드 구축

 메타5형제 | 조은새 박상원 김지우 이은지 육지현

클라우드 인프라 엔지니어 양성과정



프로젝트 목차

Chapter.01 개요

Chapter.02 아키텍처

Chapter.03 기술스택

Chapter.04 프로젝트일정

Chapter.05 인프라구축

Chapter.06 개선사항

팀원 소개



조은새

Public Cloud

Public Cloud Architecture 구성
Terraform Infra 유지보수
App 개발 및 테스트



박상원

Private Cloud

Private Cloud Architecture 구성
OpenVPN Site to Site 연결
DB 이중화



김지우

CI Pipeline

CI Pipeline 구축
Jenkins 환경 구축
이미지 취약점 분석



이은지

CD Pipeline

CD Pipeline 구축
ArgoCD 환경 구축
코드 취약점 분석



육지현

Monitoring

ArgoCD, Web 서버,
Node, MySQL 모니터링
장애, 오류 인지와 예방
Slack 알람 모니터링

개요

MetaMedical
메타5형제



백신 예약 이어 또 '먹통'...쿠브앱 전산장애, 질병청 "송구"

홍완기 기자 wangj0602@doctorsnews.co.kr | 승인 2021.12.14 10:15 | 댓글 1



"시행 첫날 방역패스 적용 안해...13일 위반, 과태료 부과 안 한다"
서버 증설 했지만 실시간 대량 인증처리 장애 등 과부하 대응 '미흡'

복증하는 의료 데이터..“클라우드 기반 IT 혁신 필요해”

2023.04.26 16:49:40

55~59세 접종예약 재개…시스템 불안 또 '접속대기'

송고시간 | 2021-07-14 21:09

신선미 기자
[기자 페이지](#)

(서울=연합뉴스) 신선미 기자 = '예상 대기시간 111시간 23분 52초'.

만 55~59세 대상 신종 코로나바이러스 감염증(코로나19) 백신 접종 사전예약이 이틀 만인 14일 오후 8시 재개된 가운데 초반부터 접속이 원활하지 않아 이용자들이 불편을 겪고 있다.

예약 신청자가 한꺼번에 몰리면서 코로나19 예방접종 사전예약 시스템에 또다시 접속 장애 현상이 발생한 것이다.

시나리오

MetaMedical
메타5형제



Meta NEWS



현재 대한민국은 펜데믹 상황으로 일정기간 동안 백신접종 예약을 신청 받는다.

기존 온프레미스로 운영중인 중앙방역 대책 본부는 일정기간동안 만 40세에서 59세 대상으로 백신 접종 예약을 받는다.

민감정보를 보호하면서 고가용성을 갖춘 백신접종 예약서비스를 제공할 수 있는 인프라가 필요한 상황.

Check List

민감정보 보호

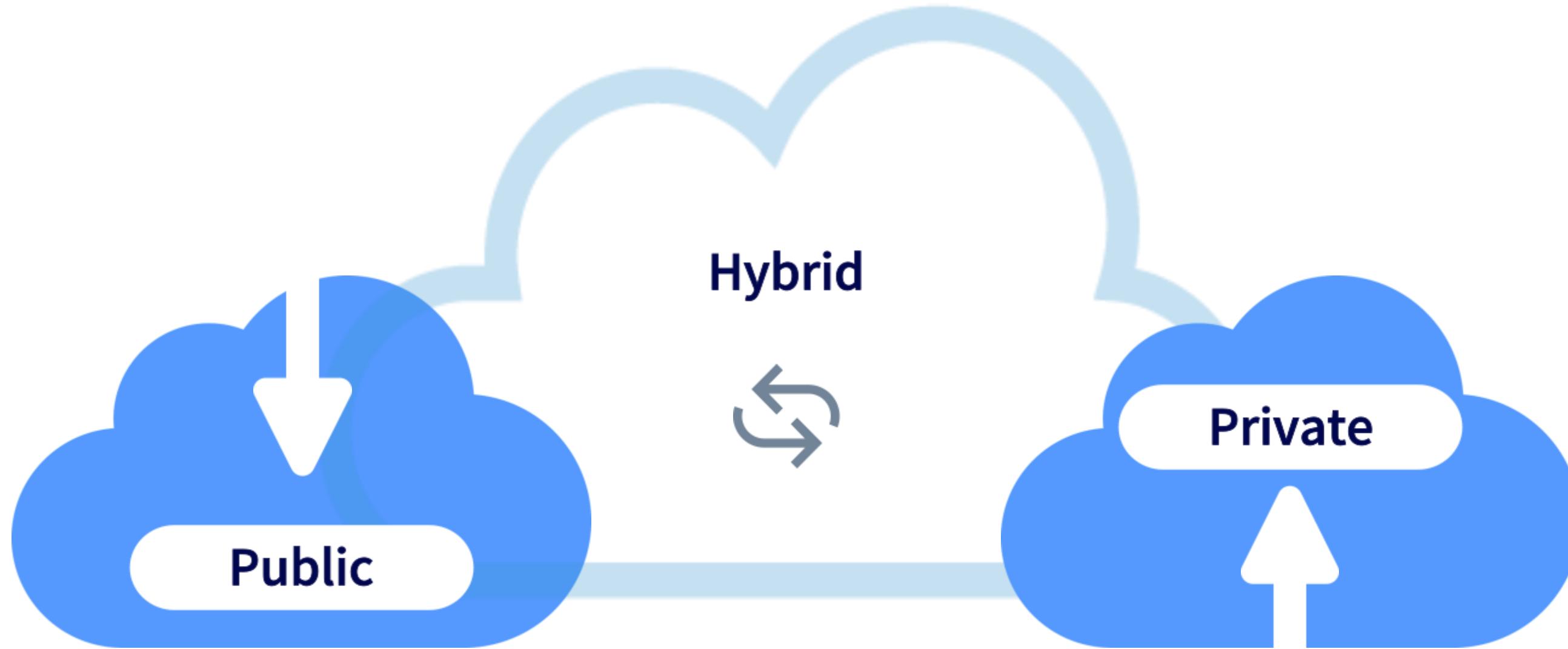
고가용성

안정적인 서비스

→ 하이브리드 클라우드 구축

프로젝트 목표

MetaMedical
메타5형제



대량의 워크로드 안정적이고 탄력적으로 처리

중요 데이터 분리 및 접근 제어
로 데이터 보안 향상

3-Tier 구성의 웹 어플리케이션 구축
신속하고 유연한 대응

프로젝트 목표

MetaMedical
메타5형제



인프라와 웹 서비스 배포 전반의

고가용성, 확장성, 안정성 확보

Public

Private

대량의 워크로드 안정적이고 탄력적으로 처리

중요 데이터 분리 및 접근 제어
로 데이터 보안 향상

3-Tier 구성의 웹 어플리케이션 구축
신속하고 유연한 대응

기술 스택

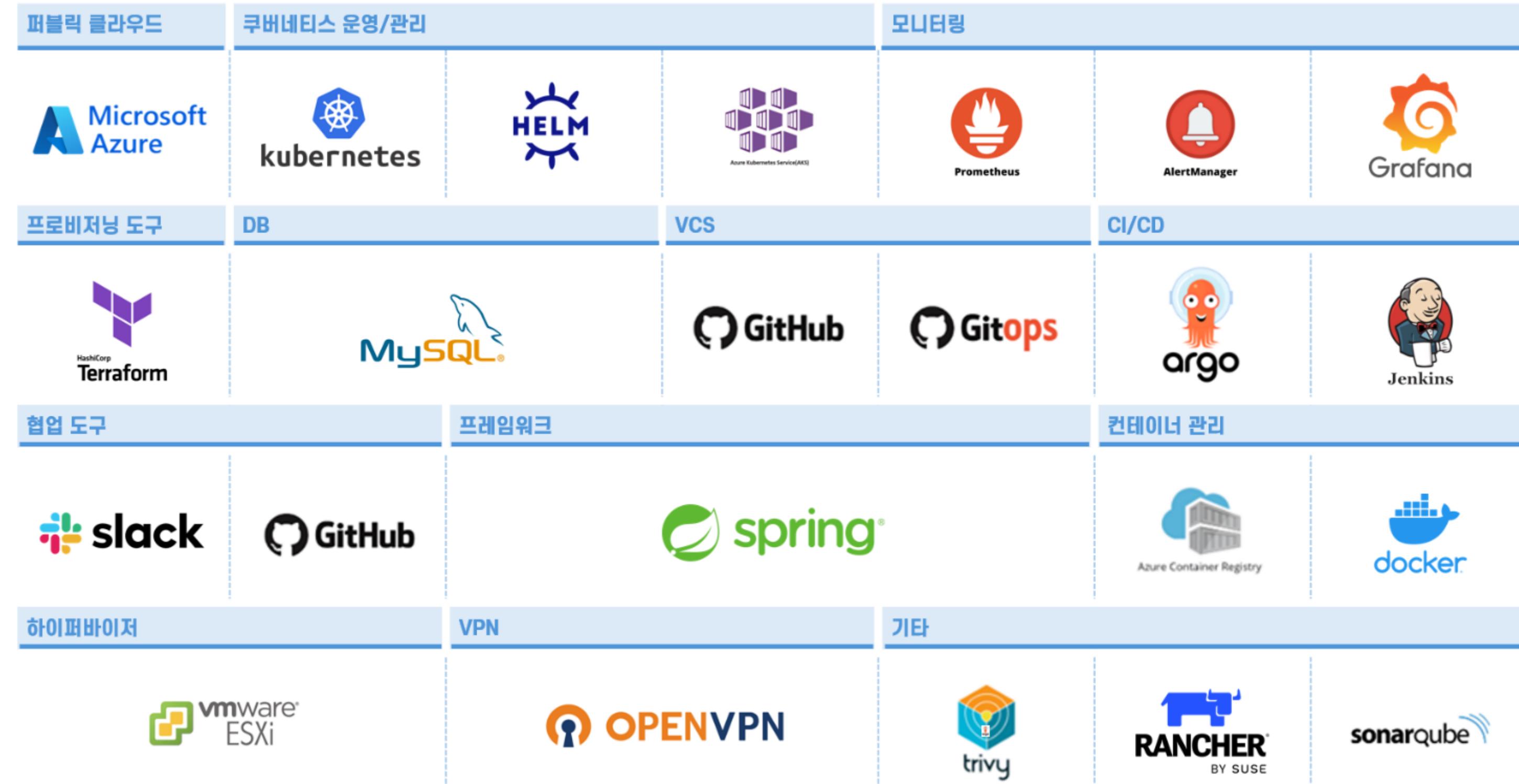
MetaMedical
메타5형제



●

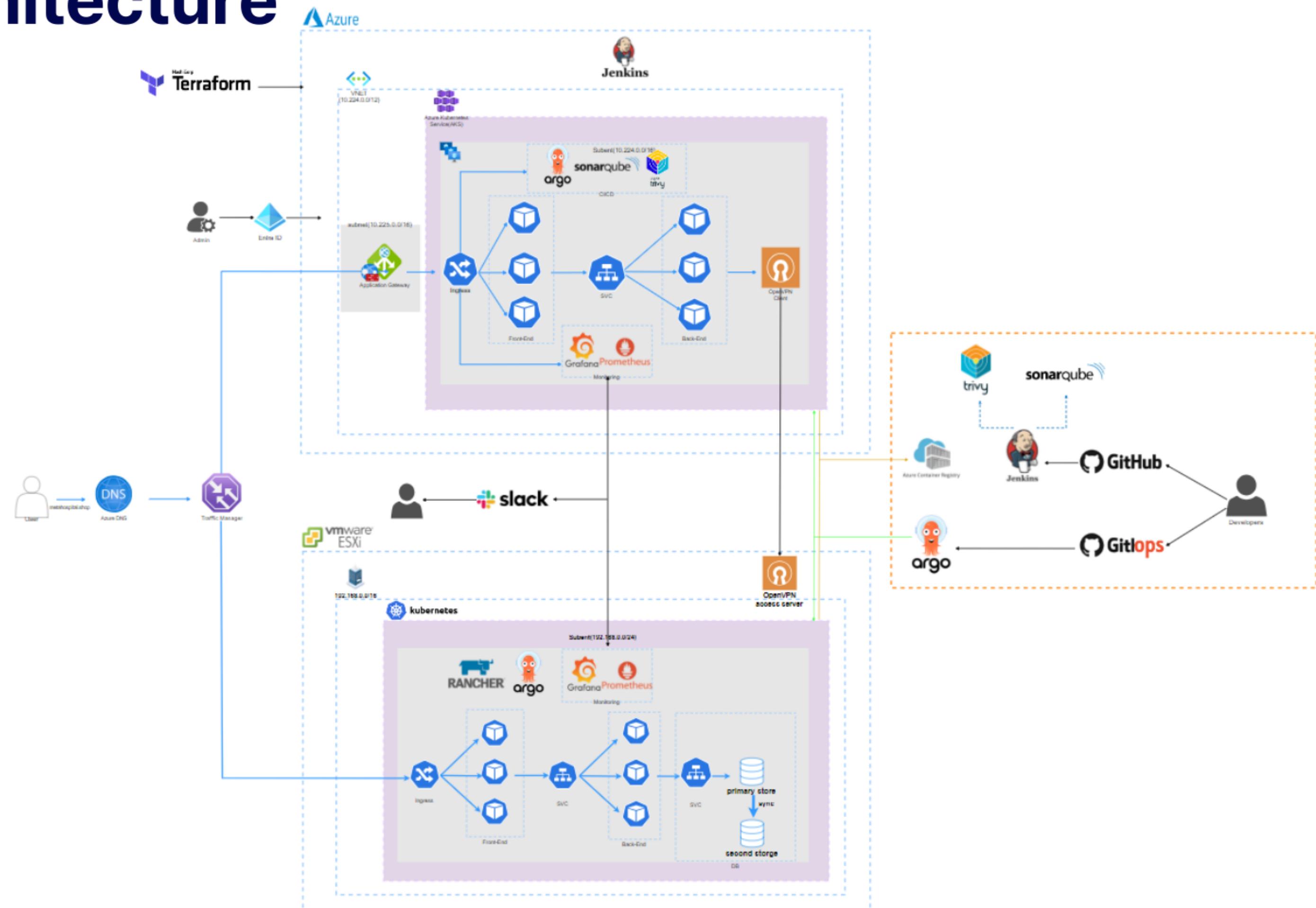
●

●



Architecture

MetaMedical
메타메디컬형제



프로젝트 일정

MetaMedical
메타5형제



협업

MetaMedical
메타5형제



slack & GitHub

일일보고 ~

+ 책갈피 추가

은지 오후 5:55
2024년 2월 8일
일일보고 - 이은지
금일추진현황
- kustomize 설계
: kustomize를 도입하여 kubernetes 매니페스트의 관리의 용이성을 높이는 설계를 진행
- jenkins-kustomize 실습
: Jenkins 내부에서 Kustomize를 통한 배포 프로세스를 테스트
- AKS - Azure Container Registry 배포
: AKS에 애플리케이션 배포 후 ACR에 이미지를 성공적으로 푸시
명일 추진 계획
- jenkins-kustomize 실습 완료
: Jenkins에서 Kustomize를 활용한 배포 자동화 실습을 마무리
미해결 문제
- kustomize 설계
: Kustomize의 상세 설계 미완성 (편집됨)

finalproject ~ Hybrid Cloud 기반 금융 상품 서비스

goodbird 오후 7:46
<https://github.com/JoEunSae/front-end>

GitHub
GitHub - JoEunSae/front-end
Contribute to JoEunSae/front-end development by creating an account on GitHub.
(34KB) ~

JoEunSae/front-end

https://github.com/JoEunSae/front-end

Ar: 3 Committed: 0 Issues: 0 Main: 2 Forks: 0

im-wujio-k 오후 9:10
2/8
일일보고 -김지우
금일추진현황
• Jenkins Pipeline 구축
• azure vm에 jenkins 구축 후 ACR에 이미지 배포
명일 추진 계획
• ACR 이미지 배포 및 aks 확인
미해결 문제
• Jenkinsfile 문제로 dockerhub에 우선 테스트 진행

im-wujio-k 오후 9:23
@goodbird
어제 CI 해결 못한 부분 때문에요!
pod 를 했을 때 error log 확인해봤더니 `no main manifest attribute, in /app.jar` 이렇게 떠서요.
gradle 사용할 때 jar 파일이 2개 생성된다는데... 확인 해주실 수 있을까요?
그리고 build.gradle에 아래 코드 한 번 추가 부탁드려요!

```
jar {
    enabled = false
}
```

<https://dongjuppp.tistory.com/87>

⚠️ 제목없음!
`no main manifest attribute in` 예외
no main manifest attribute in 예외는 spring 애플리케이션을 빌드한 결과물로 나온 jar파일에서 처음 호출할 Main 메소드를 찾지 못했다는 에러다. 주로 jar파일을 "java -jar"

front-end Public

1 Branch 0 Tags

Go to file Add file Code About

JoEunSae jwwoo-delete trivy code0217:1958 5833306 - 1 hour ago 137 Commits

- .idea front-end 5 days ago
- Dockerfile front-end 5 days ago
- Jenkinsfile jwwoo-delete trivy code0217:1958 1 hour ago
- README.md Initial commit 5 days ago
- default.conf front-end 4 days ago
- health-check.html aks클러스터에서 front와 back연결 테스트 yesterday
- index.html jwwoo-delete trivy code0217:1958 2 hours ago
- join.html aks클러스터에서 front와 back연결 테스트 yesterday
- login.css front-end 5 days ago
- login.js front-end 5 days ago
- main.html front-end 5 days ago
- nginx.conf front-end 5 days ago
- sonar-project.properties commit 0544 2 days ago
- trivy-image-scan.sh jwwoo-delete trivy code0217:1958 1 hour ago
- vaccine.html aks클러스터에서 front와 back연결 테스트 yesterday

README

back-end Public

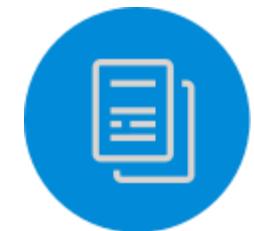
1 Branches 0 Tags

Go to file Add file Code About

JoEunSae 면전박 1806x32 - yesterday 192 Commits

- gradle/wrapper test commit 5 days ago
- sic 면전박 yesterday
- ignore test commit 5 days ago
- Dockerfile test commit yesterday
- Jenkinsfile test commit yesterday
- README.md init 2 weeks ago
- build.gradle test commit 4 days ago
- deployment.yaml test commit 5 days ago
- gradlew plz 5 days ago
- gradlew.bat init 2 weeks ago
- settings.gradle init 2 weeks ago
- sonar-project.properties plz 1017 3 days ago
- trivy-image-scan.sh test commit yesterday

README



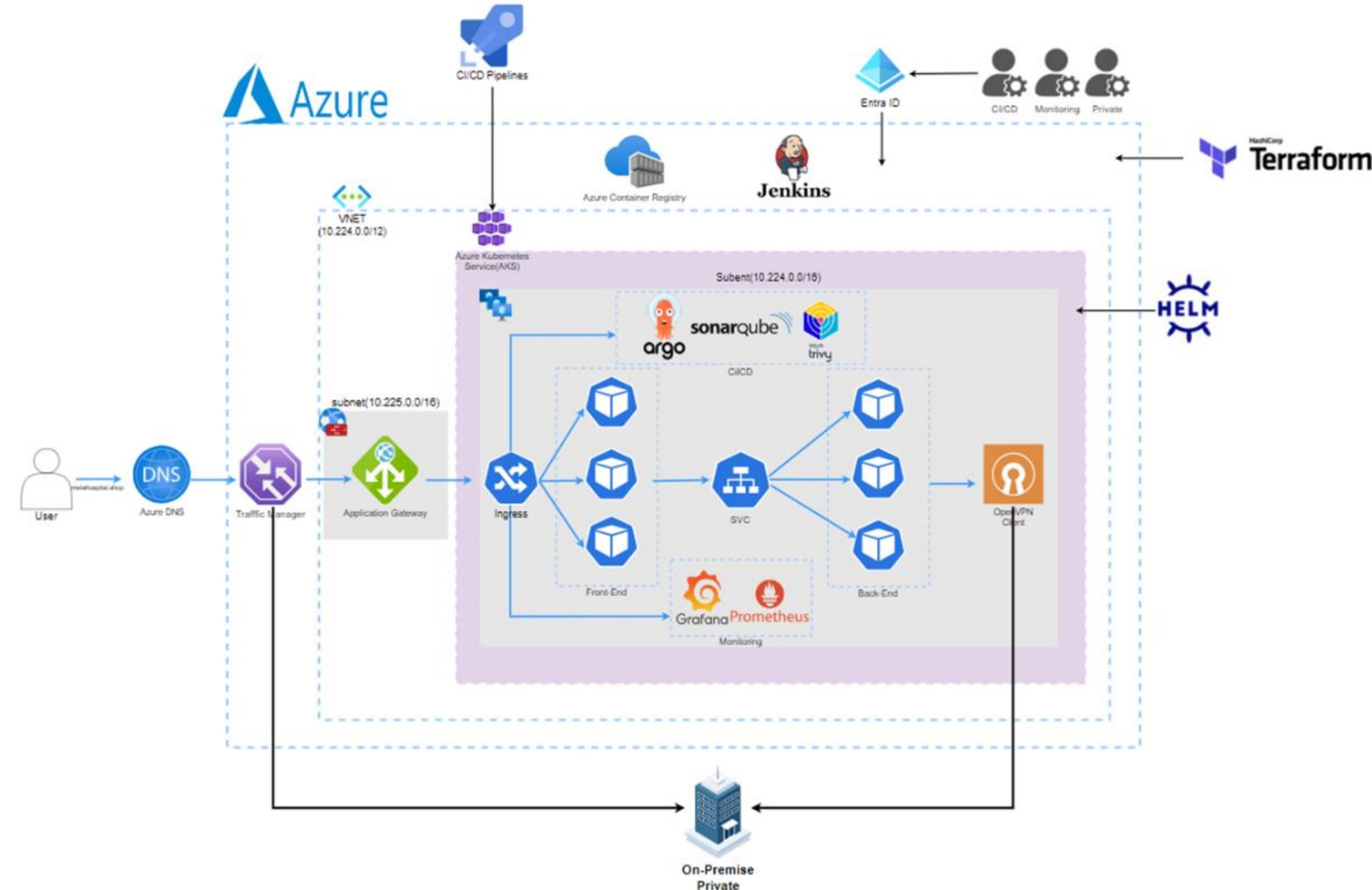
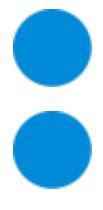
Public cloud

Azure AKS Cluster 기반 탄력적인 인프라 구축

발표자: 조은새

Public Architecture

MetaMedical
메타5형제



앱 개발 및 테스트

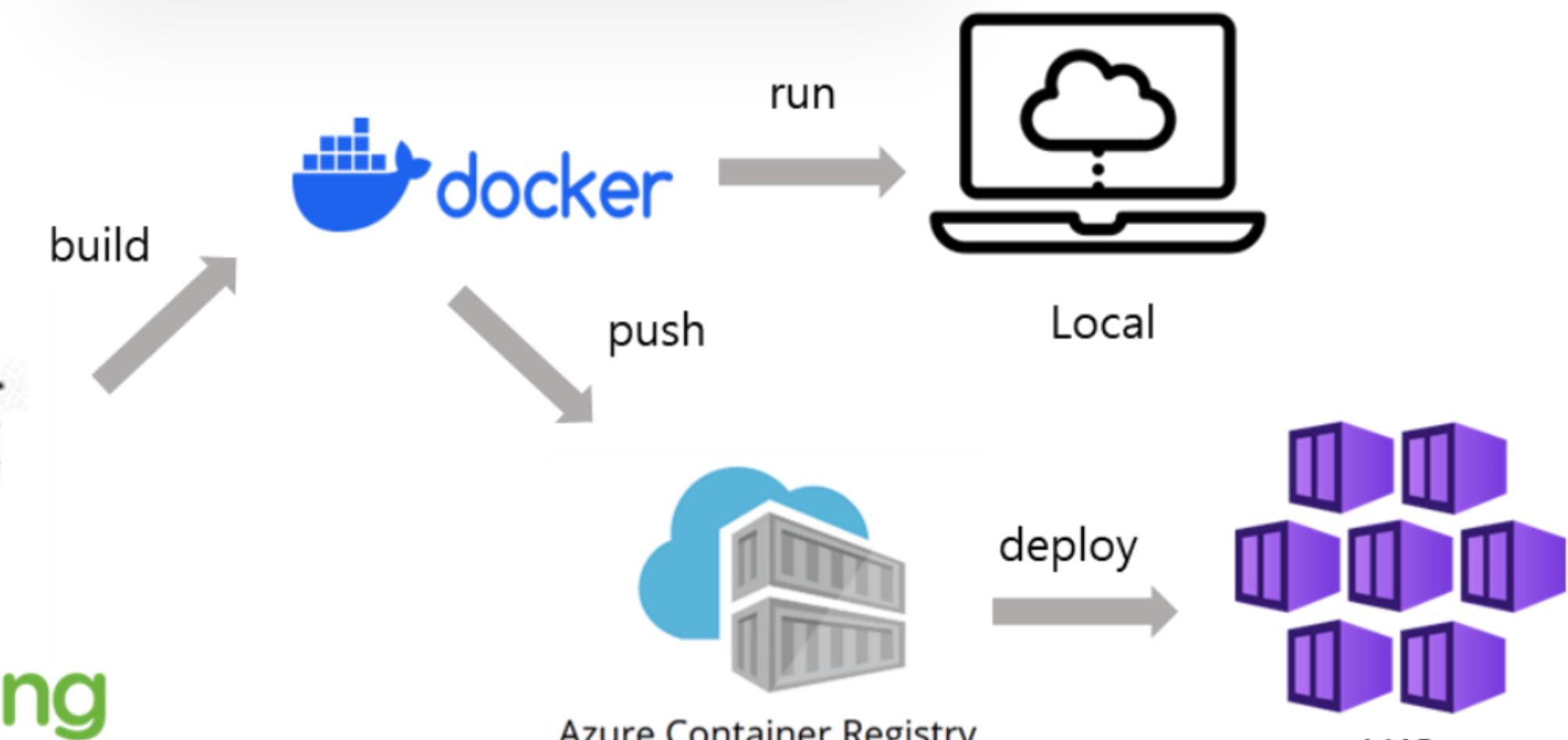
MetaMedical
메타5형제



●
●
●
●
●



로컬 환경에서 개발한 후 DevOps엔지니어가
CI/CD파이프 라인을 구축할 수 있도록
로컬/ AKS클러스터 환경에서 테스트



앱 개발 및 테스트

MetaMedical
메타5형제



회원가입!

ID
user
Password
...
이름
조은재
나이
20
주소
서울
핸드폰 번호
010-1111-2222
Register

login

user
...
 Remember

회원이 아니신가요? 회원가입

메타 의료 시스템

백신 신청
백신을 신청하세요!
신청하기

건강검진 신청
건강검진 서비스를 신청하세요!
신청하기

백신 신청

이름
조은재
나이
20
주소
서울
신청하기

건강검진 신청

이름
나이
주소
신청하기



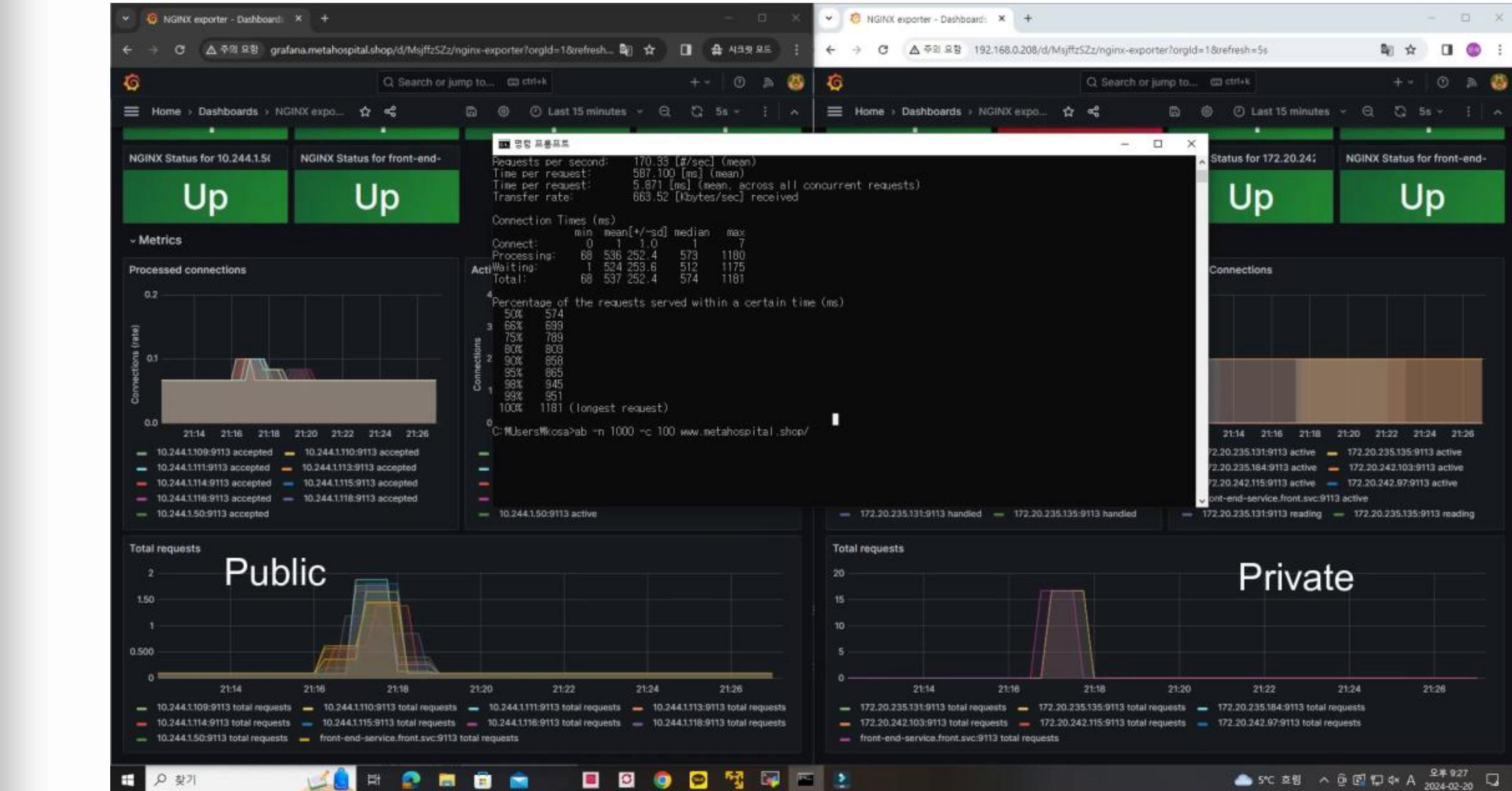
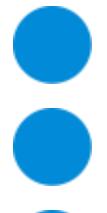
TrafficManager를 통한 트래픽분산

MetaMedical
메타5형제



TrafficManager를 통한 트래픽분산 시연 영상

MetaMedical
메타5형제



Role Based Access Control



CI
CD
Monitoring
On-premise

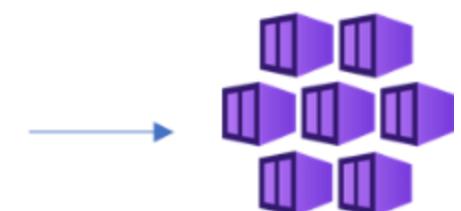


여러 관리자들을 사용자로 추가하여 각자의 역할에 맞게
액세스를 제어하도록 설정

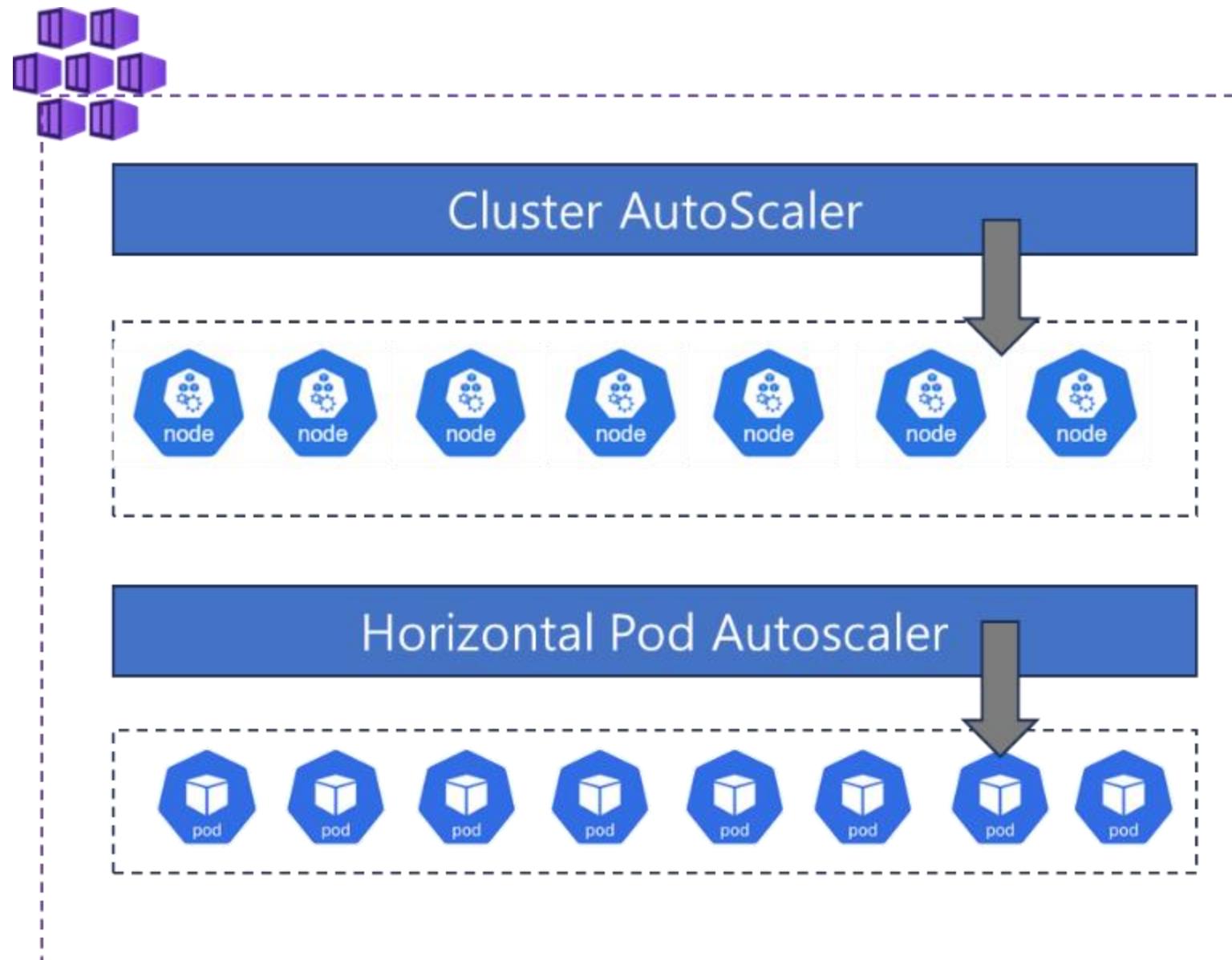
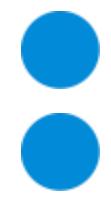


이름	형식
6개 항목 (사용자 6명)	
소유자 (1)	
<input type="checkbox"/> 조은 gjsl1945_naver.com#EXT#@gjsl1945...	사용자
기여자 (2)	
<input type="checkbox"/> 박상원(Private Cloud) (게스트) ppsw4488@naver.com	사용자
<input type="checkbox"/> 이온지(CD) (게스트) rlozimeta_2@outlook.com	사용자
Azure Kubernetes Fleet Manager Contributor Role (1)	
<input type="checkbox"/> 윤 지원(Monitoring) (게스트) yukgh8@gmail.com	사용자
가상 머신 관리자 로그인 (1)	
<input type="checkbox"/> 김지우(CI) (게스트) jiwwoo@student.hyunwoman.ac.kr	사용자
사용자 역세스 관리자 (1)	
<input type="checkbox"/> 이온지(CD) (게스트) rlozimeta_2@outlook.com	사용자

역할	범위	조건
소유자 ⓘ	이 리소스	추가
기여자 ⓘ	이 리소스	없음
기여자 ⓘ	이 리소스	없음
Azure Kubernetes Fleet Manager Contributor Role ⓘ	이 리소스	없음
가상 머신 관리자 로그인 ⓘ	이 리소스	없음
사용자 역세스 관리자 ⓘ	이 리소스	추가



Azure AKS Horizontal Pod Autoscaler



HPA.yaml

```
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: frontend-hpa
  namespace: front
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: front-end-deployment
  minReplicas: 2
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: cpu
    target:
      type: Utilization
      averageUtilization: 10
```



CPU 사용률이
10%가 넘으면
초대 10개까지
pod 확장

Auto scaling



노드 이름으로 필터링
전체 노드 이름 일렬

노드	상태	CPU	메모리	디스크	Pod	Kubernetes 버전
aks-agentpool-36424117-vmss000001	준비됨	54%	104%	31%	-	1.27.7

노드 풀 스케일링
agentpool

노드 풀은 애플리케이션을 실행할 수 있는 공간을 제공합니다. 다양한 유형의 노드 풀을 클러스터에 추가하여 다양한 워크로드를 처리할 수 있고, 기존 노드 풀을 스케일링 및 업그레이드 할 필요하지 않은 노드 풀을 삭제할 수 있습니다. 각 노드 풀에는 가상 머신이 지원하는 노드가 포함됩니다. 노드 풀에 대해 자세히 알아보기

자동 크기 조정 이벤트 0
자동 크기 조정 경고 0
강화가 트리거되지 않음 0

노드 풀 상태
agentpool 성공

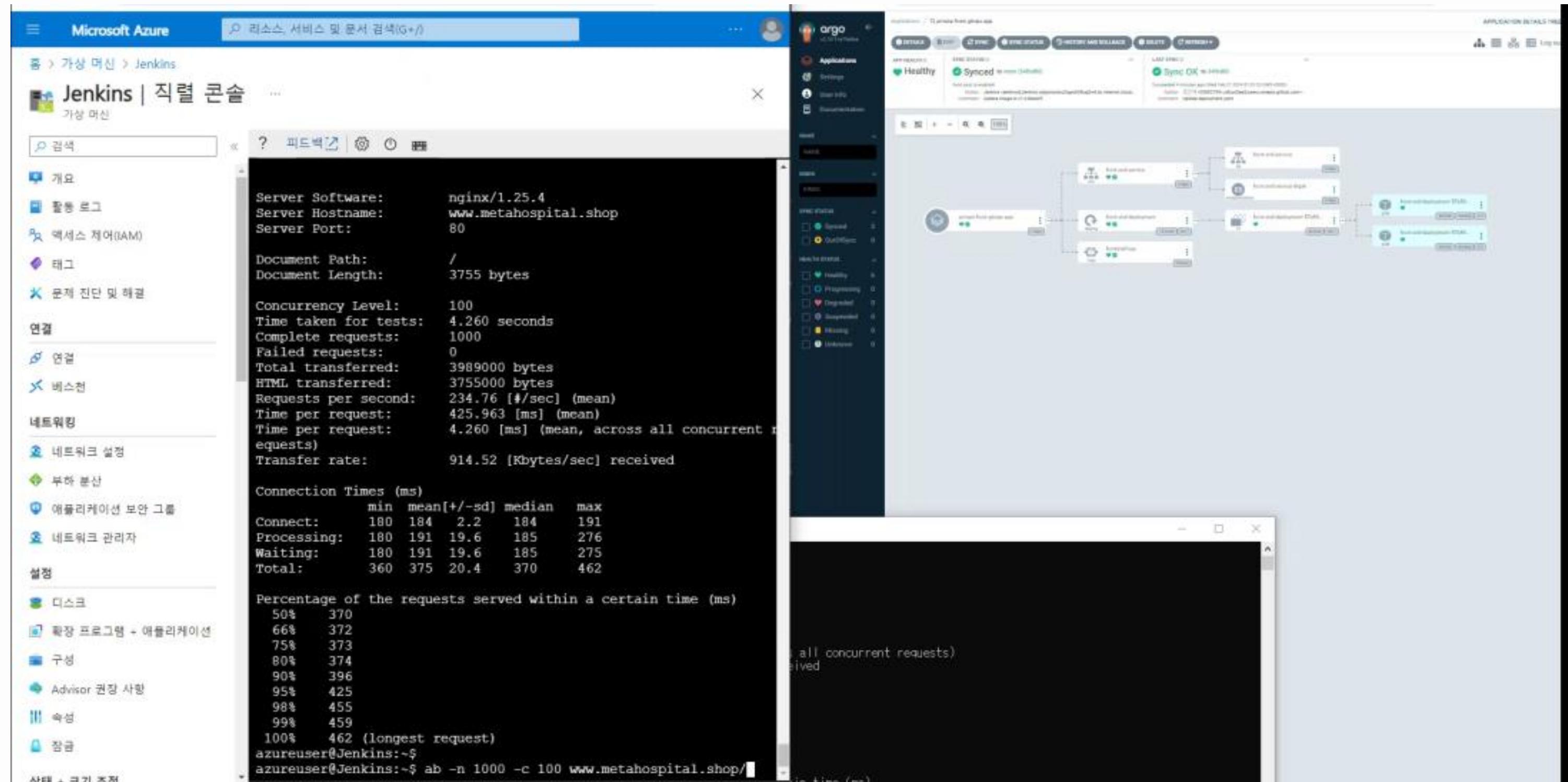
D:\#ingress>kubectl top nodes
NAME CPU(cores) CPU% MEMORY(bytes) MEMORY%
aks-agentpool-36424117-vmss000001 314m 8% 5592Mi 48%

메모리 사용량이 100%가 넘어
Pod Scaling 실패 이슈 발생

노드 풀 Scale-Up 후 노드 풀 Scaling 구성

메모리 공간 확보 후 Pod 확장 확인

Auto scaling 시연 영상



Terraform을 이용한 Azure Infra 관리



```
main.tf | import.tf | provider.tf | terraform.tf

143 }
144
145 resource "azurerm_kubernetes_cluster_node_pool" "res-11" {
146   enable_auto_scaling = true
147   kubernetes_cluster_id = "/subscriptions/c18ec0d3-5492-4dc9-b1ee-9b278abfdd7f/resourceGroups/AKS/providers/Microsoft.ContainerService/managedClusters/aks-cluster"
148   max_count          = 1
149   min_count          = 1
150   mode               = "System"
151   name               = "agentpool"
152   vm_size             = "Standard_DS2_v2"
153   depends_on = [
154     azurerm_kubernetes_cluster.res-10,
155   ]
156 }
157 resource "azurerm_web_application_firewall_policy" "res-14" {
158   location        = "koreacentral"
159   name            = "aksfw"
160   resource_group_name = "AKS"
161   managed_rules {
162     managed_rule_set {
163       version = "3.2"
164     }
165   }
166   policy_settings {
167   }
168 }
169 resource "azurerm_dns_zone" "res-15" {
170   name            = "metahospital.shop"
171   resource_group_name = "AKS"
172 }
173 resource "azurerm_network_interface" "res-16" {
174   location        = "koreacentral"
175   name            = "aks-pnic-a164d1b6-e496-4cc1-b2f8-dc3630c29036"
176   resource_group_name = "AKS"
177   ip_configuration {
178     name            = "aks-subnet-1"
179     private_ip_address_allocation = "Dynamic"
180     subnet_id      = "/subscriptions/c18ec0d3-5492-4dc9-b1ee-9b278abfdd7f/resourceGroups/MS_AKS_aks-cluster_koreacentral/providers/Microsoft.Network/virtualNetworks/aks-vnet/subnets/aks-subnet-1"
181   }
182 }
183 import {
184   id = "/subscriptions/c18ec0d3-5492-4dc9-b1ee-9b278abfdd7f/resourceGroups/AKS/providers/Microsoft.ContainerService/managedClusters/aks-cluster"
185   to = azurerm_ssh_public_key.res-1
186 }
187 import {
188   id = "/subscriptions/c18ec0d3-5492-4dc9-b1ee-9b278abfdd7f/resourceGroups/AKS/providers/Microsoft.ContainerService/managedClusters/aks-cluster"
189   to = azurerm_linux_virtual_machine.res-2
190 }
191 import {
192   id = "/subscriptions/c18ec0d3-5492-4dc9-b1ee-9b278abfdd7f/resourceGroups/AKS/providers/Microsoft.ContainerService/managedClusters/aks-cluster"
193   to = azurerm_virtual_machine_extension.res-3
194 }
195 import {
196   id = "/subscriptions/c18ec0d3-5492-4dc9-b1ee-9b278abfdd7f/resourceGroups/AKS/providers/Microsoft.ContainerService/managedClusters/aks-cluster"
197   to = azurerm_container_registry.res-4
198 }
199 import {
200   id = "/subscriptions/c18ec0d3-5492-4dc9-b1ee-9b278abfdd7f/resourceGroups/AKS/providers/Microsoft.ContainerService/managedClusters/aks-cluster"
201   to = azurerm_container_registry_scope_map.res-5
202 }
203 import {
204   id = "/subscriptions/c18ec0d3-5492-4dc9-b1ee-9b278abfdd7f/resourceGroups/AKS/providers/Microsoft.ContainerService/managedClusters/aks-cluster"
205   to = azurerm_container_registry_scope_map.res-6
206 }
```

Private Cloud

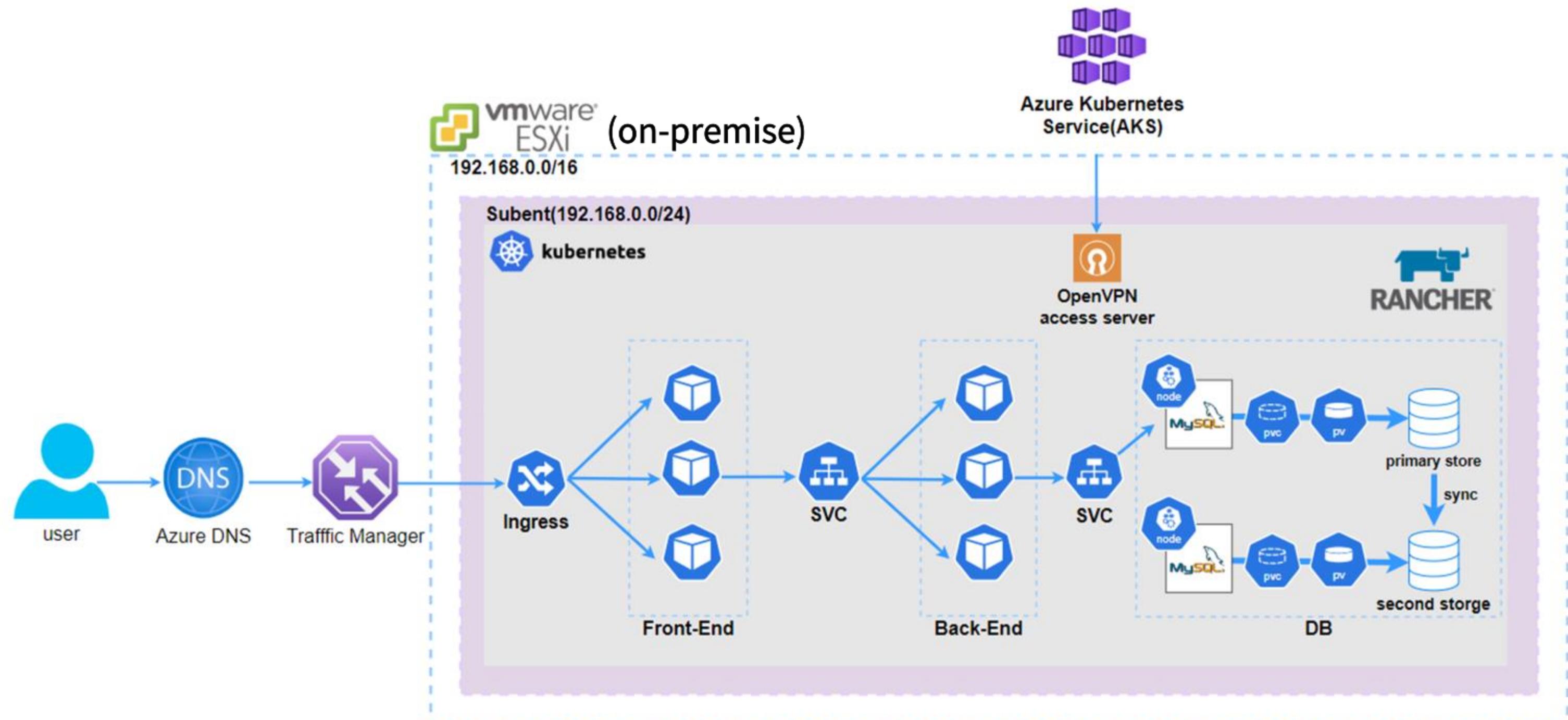
VMware ESXi 기반 K8S 서비스 인프라 환경 구축



발표자: 박상원

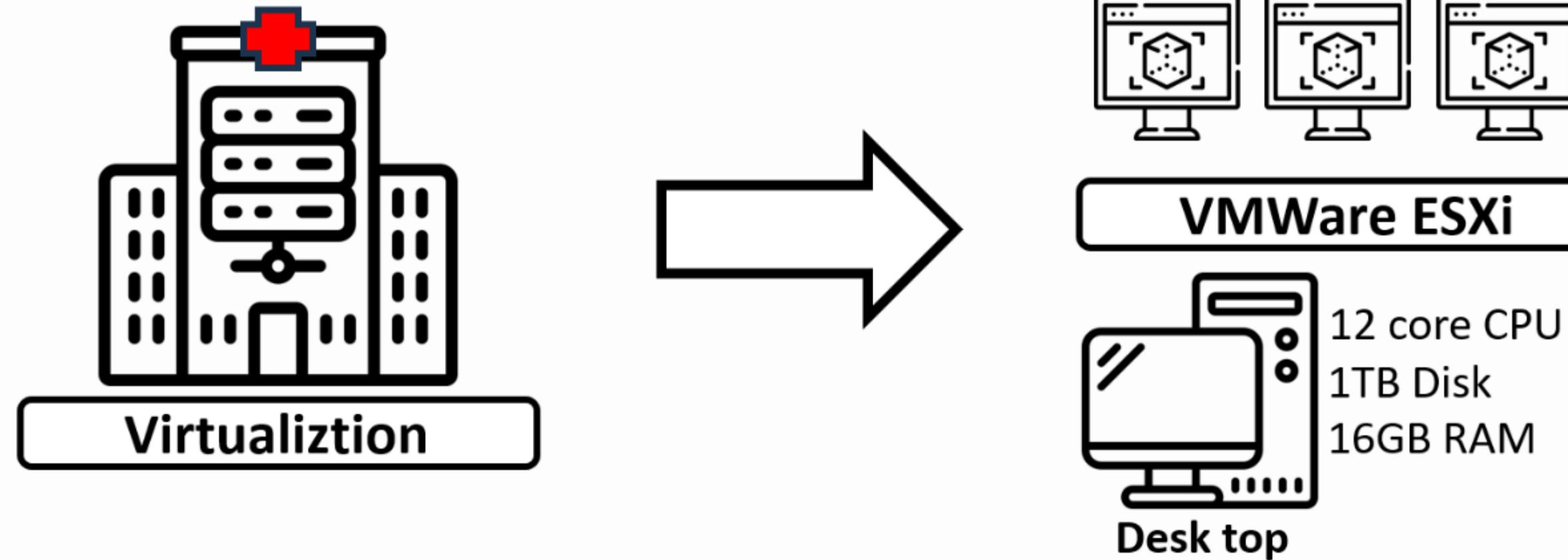
Private Cloud Architecture

MetaMedical
메타5형제



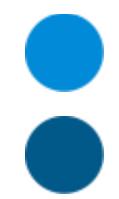
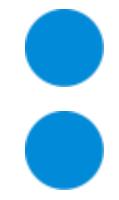
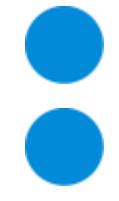
Virtualization by VMware ESXi

Virtualizatin by VMware ESXi



병원의 데이터 센터의 가상화 환경을 가정하기위해 Desktop을 Vmware ESXi 8 을 통해 가상화

VMware ESXi를 통한 가상화



The screenshot shows the VMware ESXi Host Client interface. On the left, there's a navigation sidebar with sections like '함께기', '호스트', '가상 시스템', '스토리지', '네트워킹', and '최근 작업'. The main pane displays a running virtual machine named 'master'. The 'master' VM details are as follows:

- Guest OS: Ubuntu Linux(64비트)
- Host OS: ESXi 8.0 U2 가상 시스템
- CPU: 4
- Memory: 6 GB
- Host Name: cp

On the right side, there are performance graphs for CPU, Memory, and Storage. Below the VM details, there are sections for '일반 정보', '하드웨어 구성', '리소스 사용', and '최근 1시간 성능 요약'. The '하드웨어 구성' section lists the following resources:

- CPU: 4 vCPUs
- Memory: 6 GB
- Hard Disk 1: 90 GB
- USB Controller: USB 2.0
- Network Adapter: VM Network (연결됨)
- Video Card: 16 MB
- CD/DVD Drive: ISO [datastore1] 05/ubuntu-22.04.3-live-server-amd64.iso
- Disk: 디스크 이미지 선택

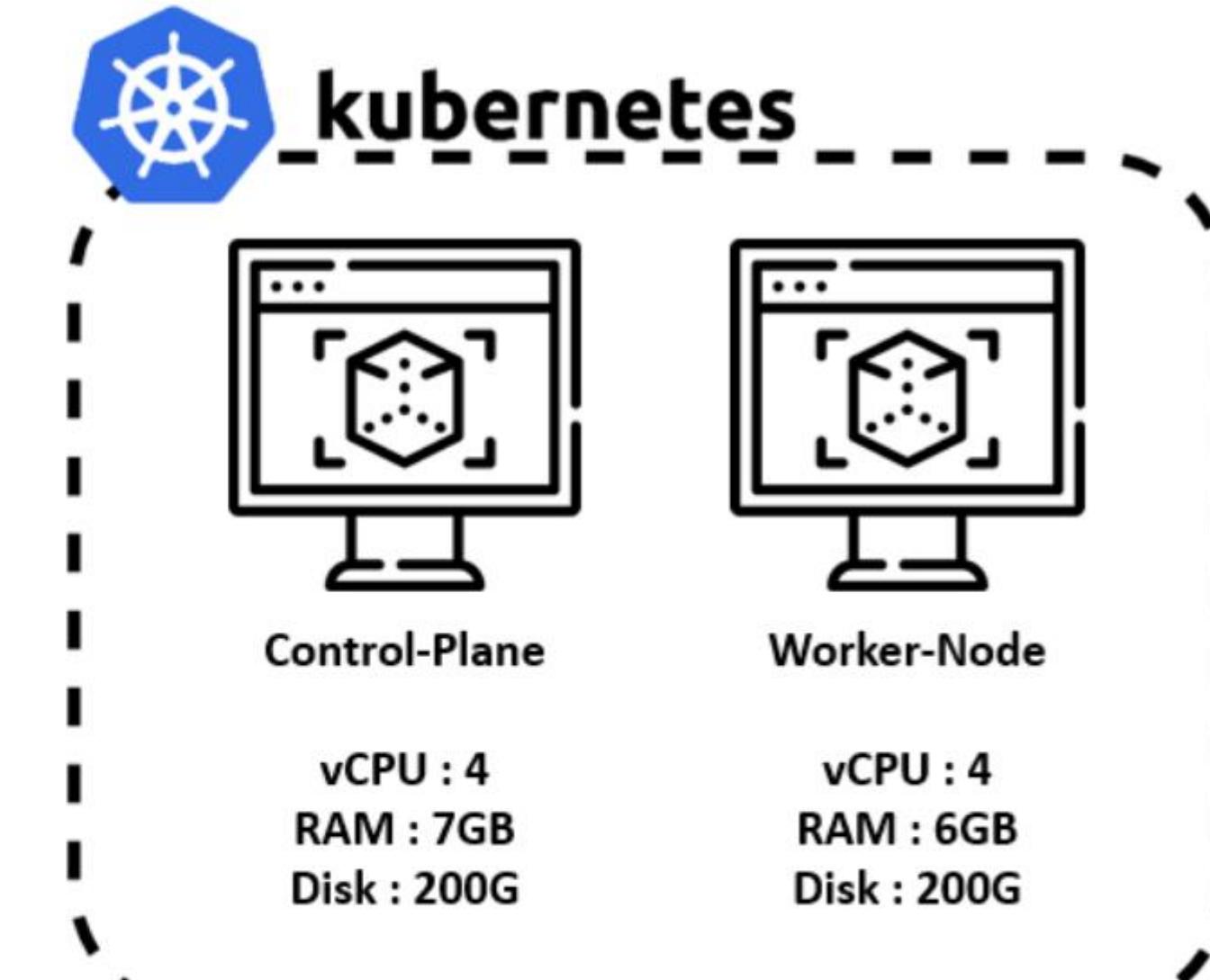
ESXi 호스트 클라이언트를 통해 가상머신 생성

Kubernetes 클러스터 생성

MetaMedical
메타5형제



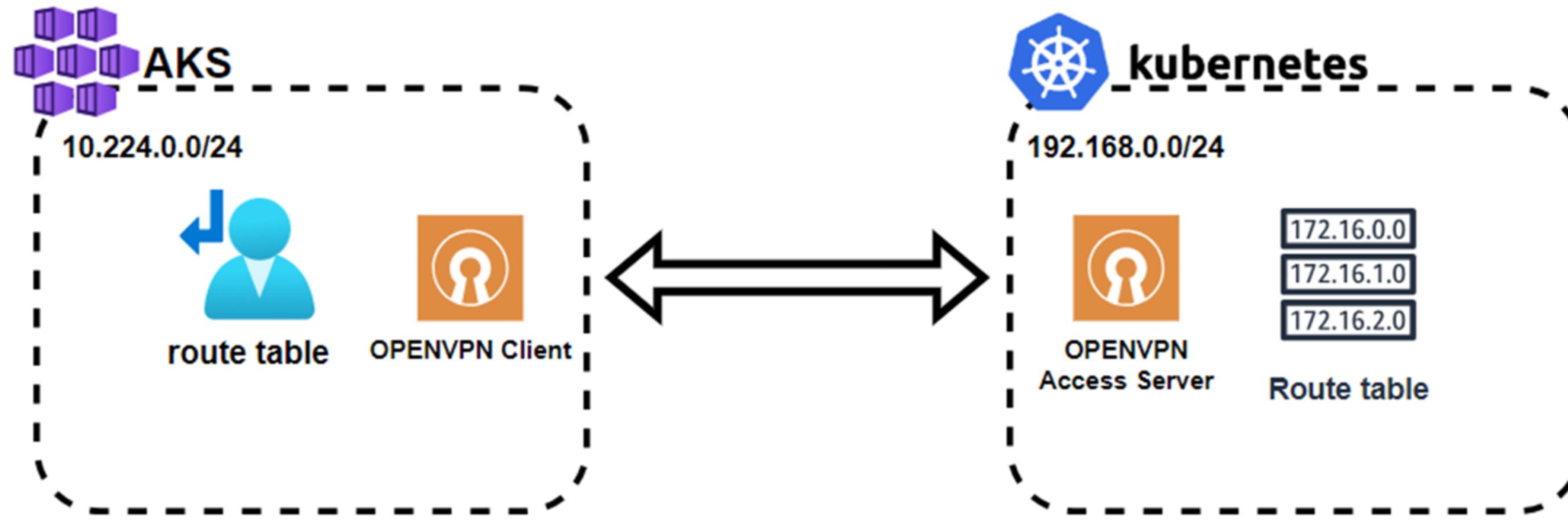
containerd



```
root@cp:~# kubectl get node -o wide
NAME     STATUS   ROLES      AGE    VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE           KERNEL-VERSION   CONTAINER-RUNTIME
cp       Ready    control-plane   47h   v1.27.1   192.168.0.243  <none>        Ubuntu 22.04.4 LTS  5.15.0-94-generic  containerd://1.6.28
worker1  Ready    worker      47h   v1.27.1   192.168.0.245  <none>        Ubuntu 22.04.4 LTS  5.15.0-94-generic  containerd://1.6.28
root@cp:~#
```

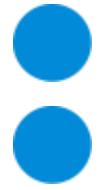
2개의 VM으로 K8s 클러스터 구축

OpenVPN을 통한 Site to Site 연결



- OpenVPN Access Server Esxi VM 생성
- OpenVPN Client Azure VM 생성
- Route Table 설정을 통한 게이트 웨이 설정
- 공유기 포트포워딩
- PING 테스트를 통한 연결 확인

OpenVPN 을 통한 Site to Site 연결



```
psw@cp:~$ ip route
default via 192.168.0.1 dev ens34 proto static
10.224.0.0/16 via 192.168.0.242 dev ens34
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1 linkdown
172.20.235.128/26 via 192.168.0.245 dev ens34 proto 80 onlink
blackhole 172.20.242.64/26 proto 80
172.20.242.65 dev calid889f760681 scope link
172.20.242.66 dev cali0ae39a1cb16 scope link
172.20.242.67 dev cali2702fb1dc86 scope link
172.20.242.68 dev cali66a78ee5f19 scope link
172.20.242.69 dev calie7e0ce1213d scope link
172.20.242.70 dev cali548dc992b18 scope link
172.20.242.71 dev cali37b09b1cd06 scope link
172.20.242.73 dev cali69336619c4d scope link
172.20.242.75 dev calie3a276562ae scope link
172.20.242.76 dev cali7d08b112e80 scope link
172.20.242.77 dev cali31ff59d11e scope link
172.20.242.83 dev caliadcd5e70516 scope link
172.20.242.88 dev cali1e99e54ab41 scope link
172.20.242.89 dev cali08d25850ac1 scope link
172.20.242.113 dev cali6579c2cb4e0 scope link
172.20.242.120 dev calif4b1f4923a3 scope link
172.20.242.122 dev calid4ea85c23d2 scope link
172.27.224.0/20 via 192.168.0.242 dev ens34
192.168.0.0/24 dev ens34 proto kernel scope link src 192.168.0.243
```

10.244.0.0/16 이 목적지인 트래픽의 라우팅 설정

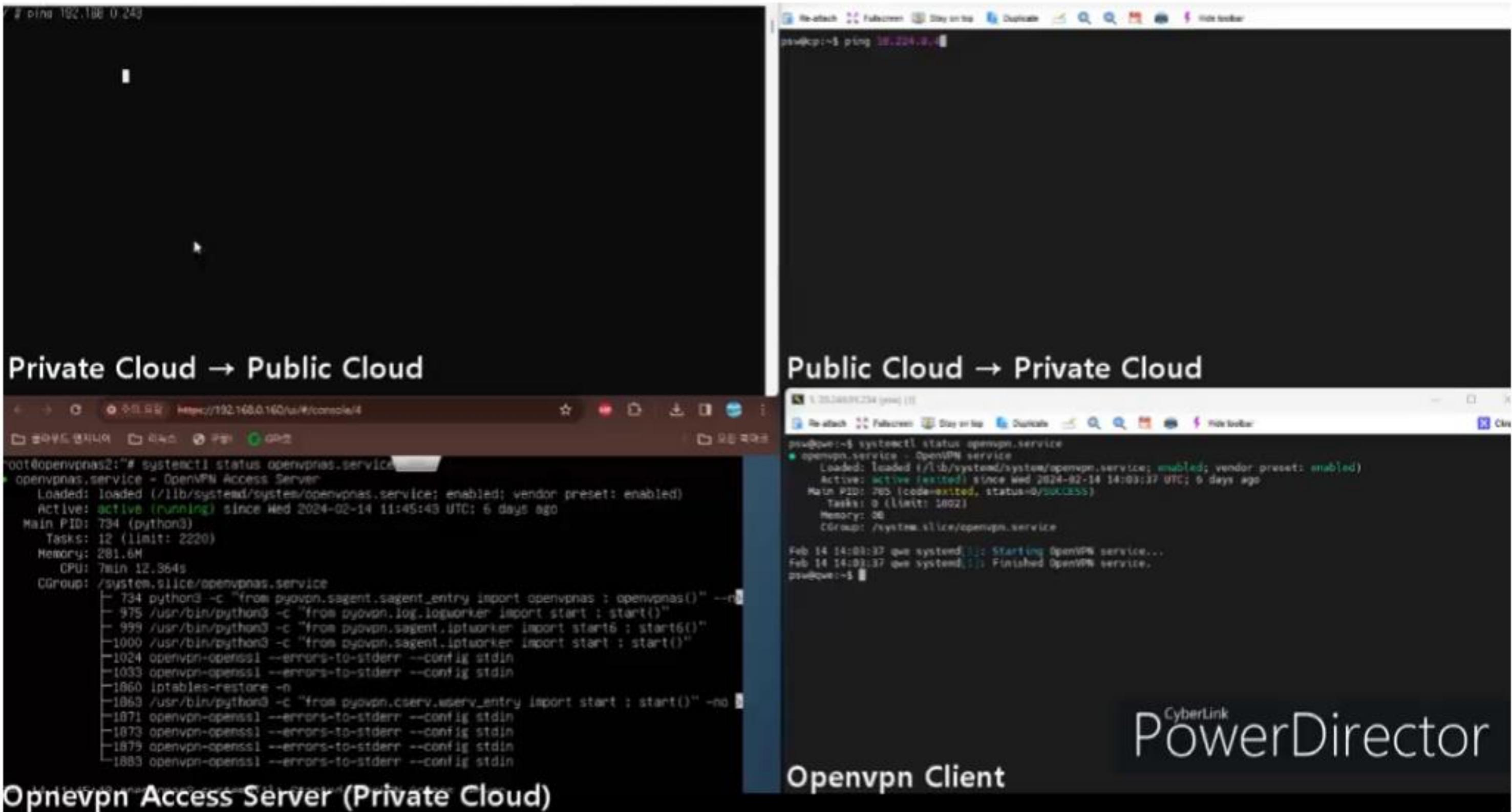
The screenshot shows the Azure portal interface for managing a route table named 'aks-agentpool-20474252-routetable'. The left sidebar includes options like 검색, 태그, 문제 진단 및 해결, 설정, 구성, 경로, 서브넷, 속성, 잠금, 모니터링, 경고, 자동화, CLI / PS, 작업(미리 보기), 템플릿 내보내기, and 도움말. The main content area displays the route table details, including its location (Korea Central), resource group (mc_aks_aks-cluster_koreacentral), and subnet (aks-subnet). It also lists three routes:

이름	주소 접두사	다음 흐 험식	다음 총 IP 주소
aks-agentpool-36424117-vmss000001__10...	10.244.1.0/24	가상 어플라이언스	10.224.0.4
vpntoprivate	192.168.0.0/16	가상 어플라이언스	10.224.0.6
vpntoturnel	172.27.224.0/20	가상 어플라이언스	10.224.0.6

Below the routes, there is a section for subnets, listing 'aks-subnet' and 'ingress-appgateway-subnet'.

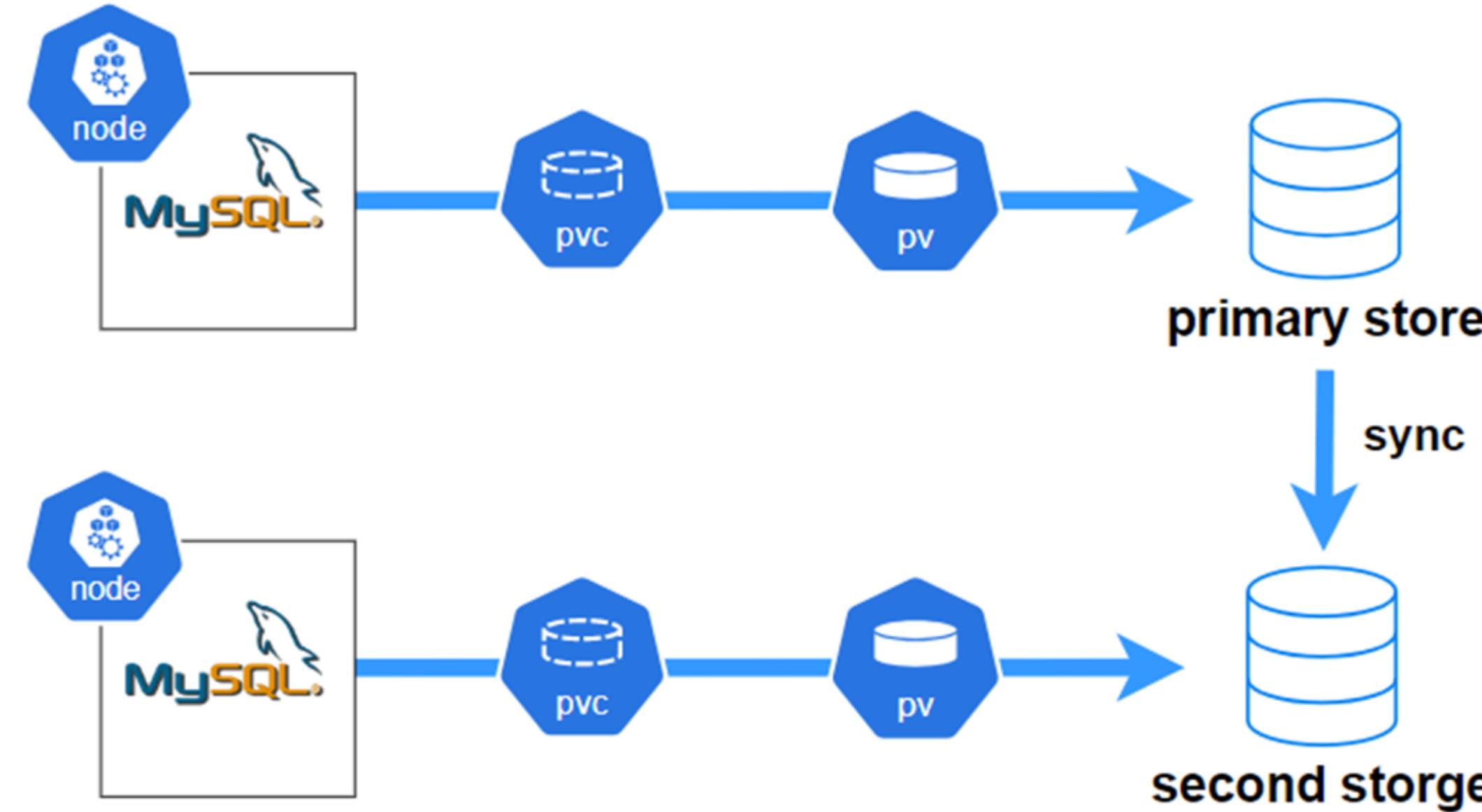
192.168.0.0/16 이 목적지인 트래픽의 라우팅 설정

OpenVPN 을 통한 Site to Site 연결 시연 영상



K8S MySQL DB 이중화

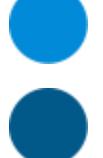
MetaMedical
메타5형제



Persistent Volumes 바운드를 통한 데이터 저장

Master/Slave 이중화 구성을 통한 단일실패 지점 극복

DB 이중화 시연 영상



```
root@mysql-0:/# mysql -u root -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 1926
Server version: 8.0.28 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

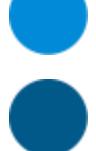
mysql> use test
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_test |
+-----+
| healthcheck   |
| users          |
| vaccine        |
+-----+
3 rows in set (0.01 sec)

mysql>
```

Rancher를 사용한 멀티클러스터 관리

MetaMedical
메타5형제



The screenshot shows the Rancher Cluster Dashboard for a cluster named "private". The dashboard includes the following information:

- Provider:** Other **Kubernetes Version:** v1.27.11 **Created:** 1 mins ago
- Total Resources:** 249
- Nodes:** 2
- Deployments:** 27
- Capacity:**
 - Pods:** Used 47 / 220 (21.36%)
 - CPU:** Reserved 1.79 / 7 cores (25.57%)
Used 0.48 / 7 cores (6.88%)
 - Memory:** Reserved 2.92 / 10 GiB (29.20%)
Used 7.39 / 11 GiB (67.18%)
- Status Indicators:** Etcd, Scheduler, Controller Manager (all green)
- Events:** (link)

The left sidebar shows navigation links for Cluster, Projects/Namespaces, Nodes (2), Cluster and Project Members, Events (8), Workloads, Apps, Service Discovery, Storage, Policy, Istio, More Resources, and Cluster Tools. The version is listed as v2.8.2.

CICD

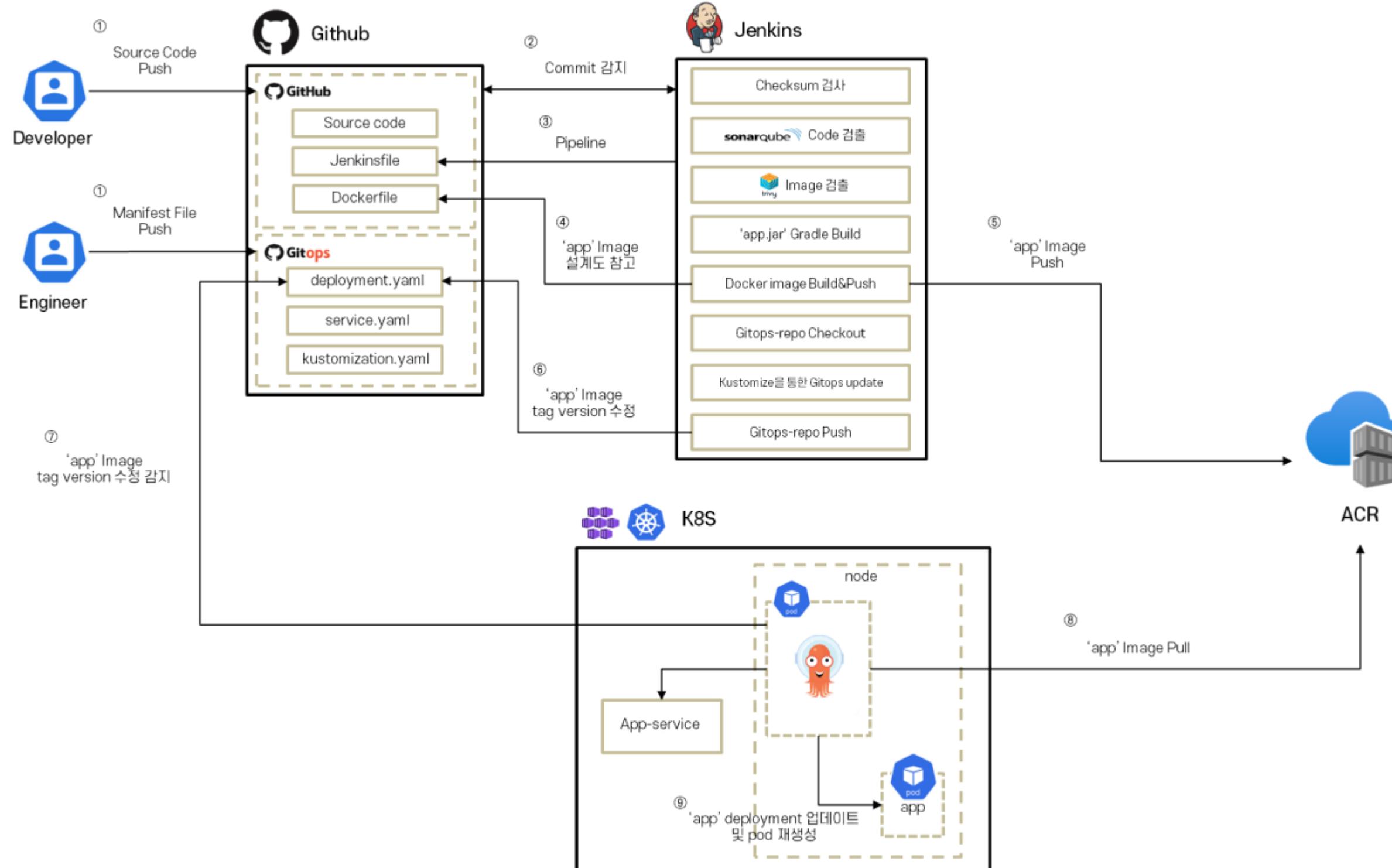
Jenkins 기반 CI 자동화 및 이미지 점검



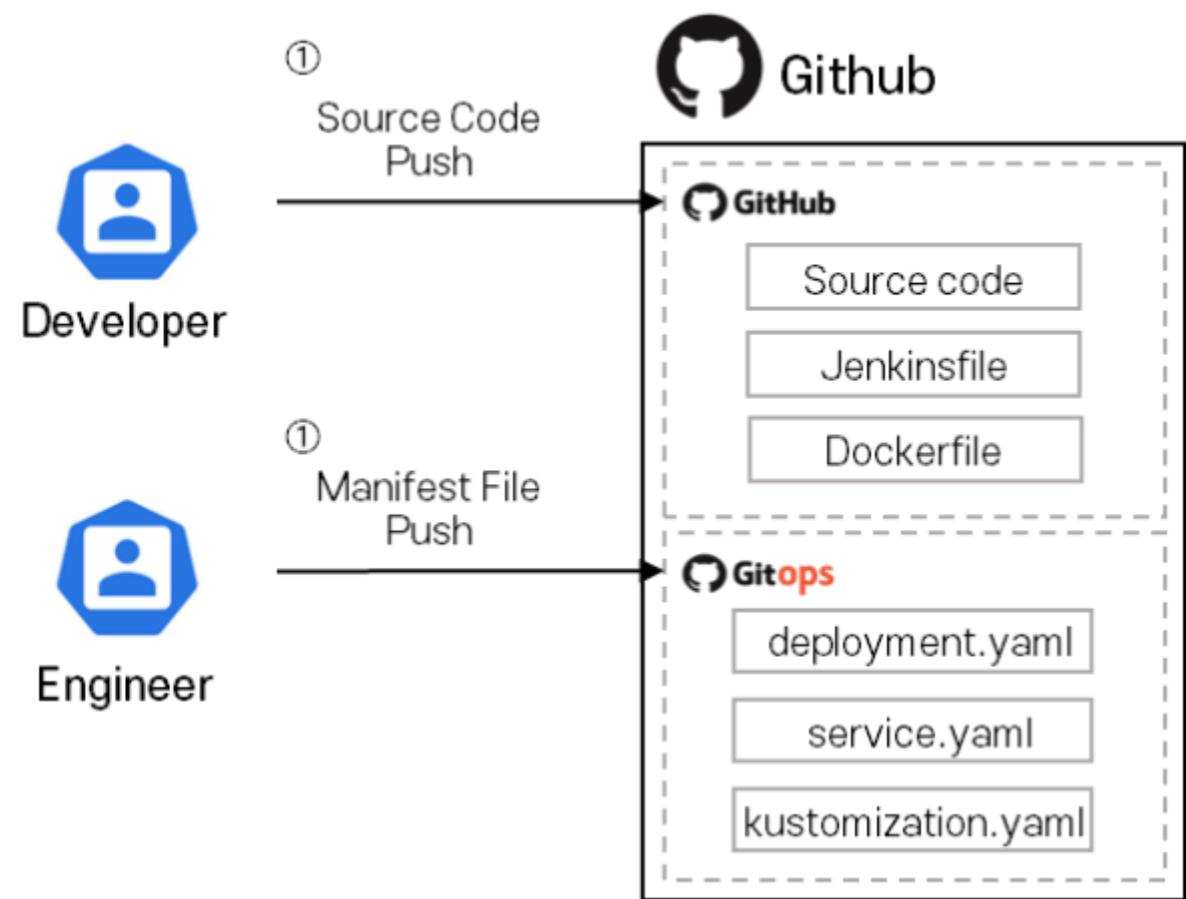
발표자: 김지우

CICD Architecture

MetaMedical
메타5형제

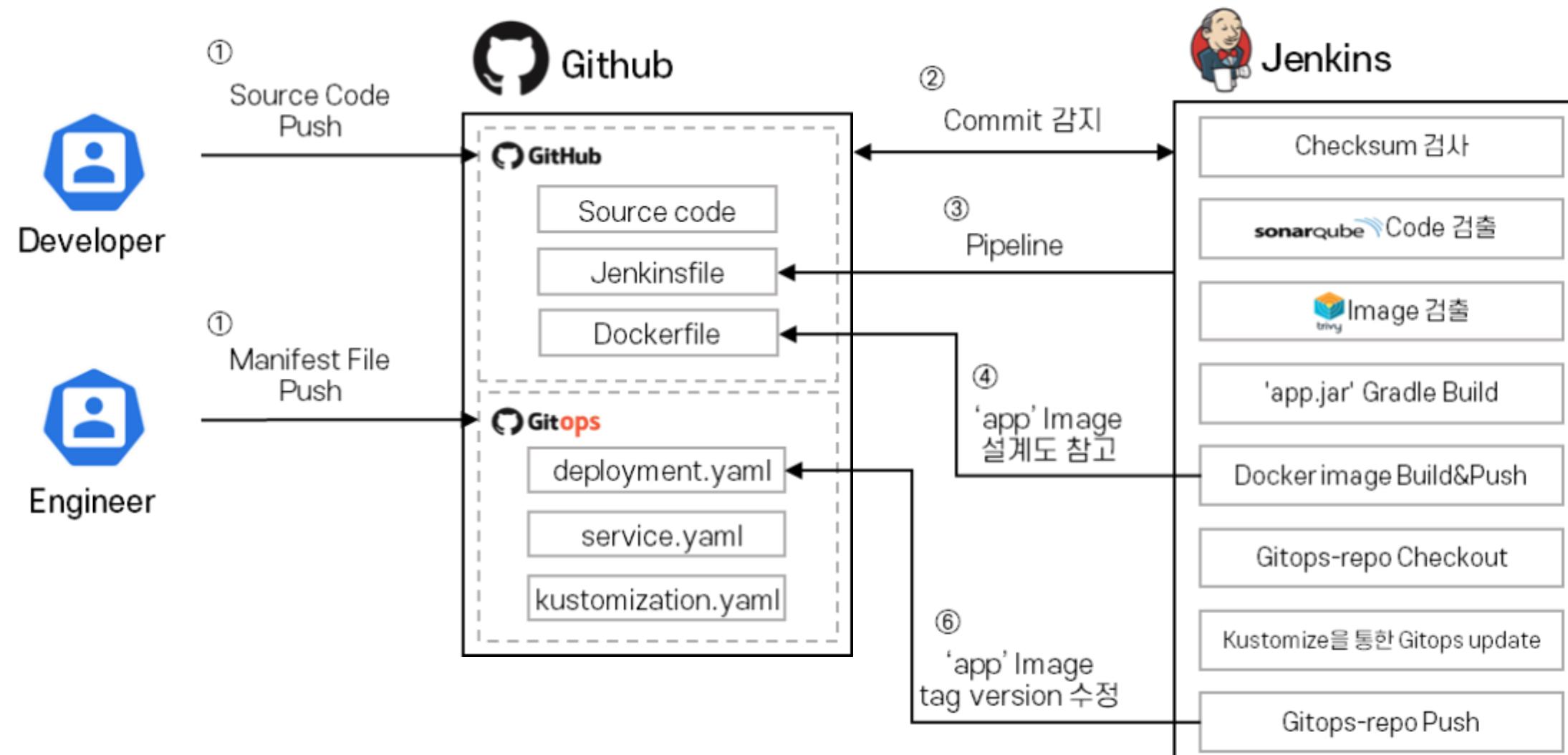


CI Architecture



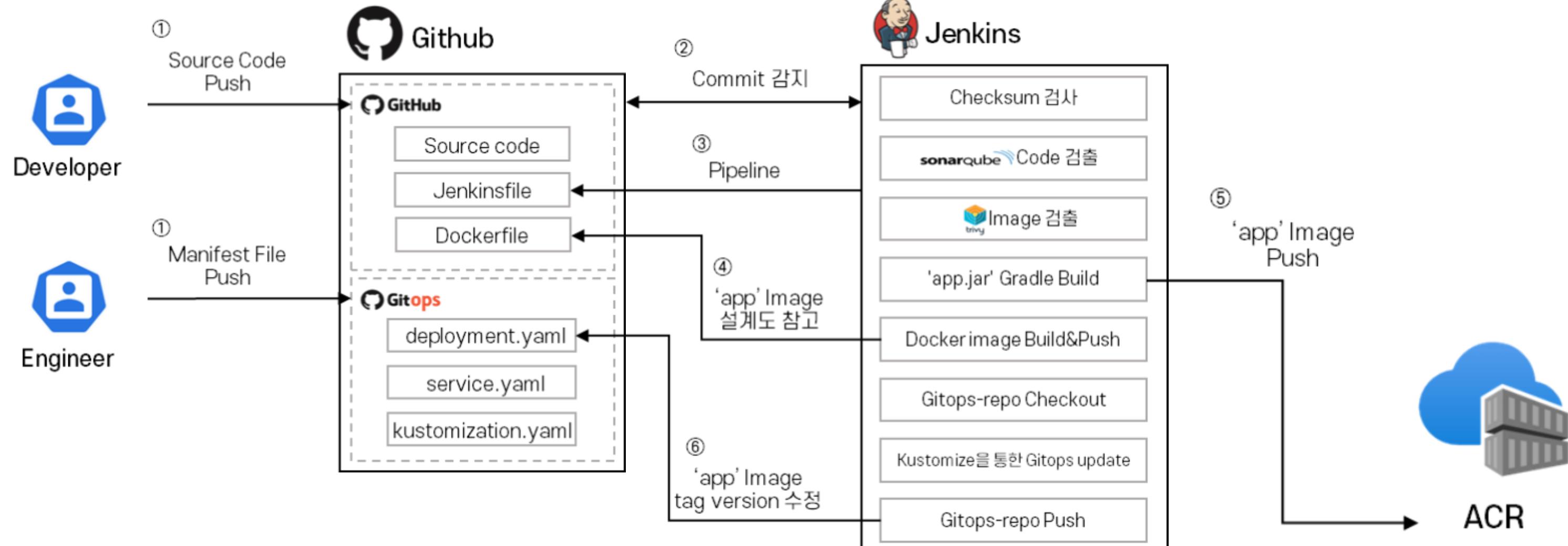
CI Architecture

MetaMedical
메타5형제



CI Architecture

MetaMedical
메타5형제



Github Webhook 설정



Payload URL *

http://172.206.241.135:8080/

Content type

application/json

Secret

http://172.206.241.135:8080/github-webhook\

▲ Github Webhook 설정

Build Triggers

Build after other projects are built ?

Build periodically ?

GitHub hook trigger for GITScm polling ?

Poll SCM ?

Quiet period ?

빌드를 원격으로 유발 (예: 스크립트 사용) ?

▲ Build Trigger 설정

Update credentials

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

Username ?

im-k-jiwoo

Treat username as secret ?

Password ?

Concealed

Change Password

ID ?

wujio

Description ?

jiwoo's credentials

▲ Github Credential 설정

Jenkins Pipeline 설정



im-k-jjwoo	add sl...	8084282 · 13 minutes ago	193 Commits
gradle/wrapper	test commit	5 days ago	
src	찐찐막	yesterday	
.gitignore	test commit	5 days ago	
Dockerfile	test commit	5 days ago	
Jenkinsfile	add slackback	13 minutes ago	
README.md	init	2 weeks ago	
build.gradle	test commit	4 days ago	
deployment.yaml	test commit	5 days ago	
gradlew	plz	5 days ago	
gradlew.bat	init	2 weeks ago	
settings.gradle	init	2 weeks ago	
sonar-project.pro...	plz 1017	3 days ago	
trivy-image-scan.sh	test commit	yesterday	

▲ Github - Jenkinsfile

Pipeline

Definition

Pipeline script from SCM

SCM

Git

Repositories

Repository URL

https://github.com/im-k-jjwoo/back-test.git

◀ Github Repo URL

Credentials

im-k-jjwoo/******** (jjwoo's credentials)

◀ Github Credentials

Add Branch

Branch Specifier (blank for 'any')

*/main

Additional Behaviours

Add

Script Path

Jenkinsfile

◀ Jenkins 파일명

▲ Jenkins Pipeline 설정

Jenkins Pipeline



```
environment {
    //Azure 계정 정보 설정
    AZURE_SUBSCRIPTION_ID = 'c8ce3edc-0522-48a3-b7e4-afe8e3d731d9'
    AZURE_TENANT_ID = '4ccd6048-181f-43a0-ba5a-7f48e8a4fa35'
    CONTAINER_REGISTRY = 'goodbirdacr.azurecr.io'
    RESOURCE_GROUP = 'AKS'
```

1

```
REPO = 'medicine/back'
IMAGE_NAME = 'medicine/back:latest'
//TAG = 'latest'
TAG_VERSION = "v1.0.Beta"
TAG = "${TAG_VERSION}${env.BUILD_ID}"
NAMESPACE = 'back'
```



Azure 및 Docker 이미지 정보 설정

```
// 체크섬 검사
stages {
    stage('Checkout') {
        steps {
            checkout scm
        }
    }
}

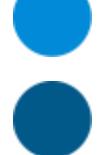
// gradle 실행 권한 부여
stage('Grant Execute Permission to Gradle Wrapper') {
    steps {
        sh 'chmod +x ./gradlew'
    }
}
```

2



소스 코드 체크아웃

Jenkins Pipeline



3

```
// JAR 파일 빌드 단계 추가
stage('Build JAR') {
    steps {
        script {
            withEnv(['JAVA_HOME=/usr/lib/jvm/jdk-21.0.2']) {
                // Gradle을 사용하여 JAR 파일 빌드
                sh './gradlew --version'
                sh './gradlew clean build --warning-mode=none
                    -x test --info'
            }
        }
    }
}
```



Gradle을 통한 JAR 파일 빌드

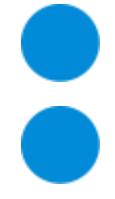
4

```
stage('Build and Push Docker Image to ACR') {
    steps {
        script {
            withCredentials([usernamePassword(
                credentialsId: 'acr-credential-id',
                passwordVariable: 'ACR_PASSWORD',
                usernameVariable: 'ACR_USERNAME')]) {
                // Log in to ACR
                sh "az acr login --name $CONTAINER_REGISTRY
                    --username $ACR_USERNAME --password $ACR_PASSWORD"
                // Dockerfile에 있는 JAR 파일을 사용하여 Docker 이미지 빌드
                sh "docker build -t $REPO:$TAG ."
                // 이미지 태그 지정 및 ACR로 푸시
                sh "docker tag $REPO:$TAG $CONTAINER_REGISTRY/$REPO:$TAG"
                sh "docker push $CONTAINER_REGISTRY/$REPO:$TAG"
            }
        }
    }
}
```



Docker 이미지 빌드 및 푸시

Jenkins+Slack 알림



The screenshot shows the Slack App Directory interface for the Jenkins CI integration. It includes sections for '통합 설정' (Integration Settings), '채널에 포스트' (Post to Channel), '토근' (Token), '설명 라벨' (Description Label), and '이름 사용자 지정' (Name User). A dashed blue box highlights the '토근' section, which displays a token value: XCUDzfcBKUrTcKQ2QvHIL90v. Below this, there's a '다시 생성' (Generate Again) button.

▲ Slack Token 생성

Update credentials

The screenshot shows the Jenkins 'Update credentials' configuration page for Slack. It includes fields for 'Scope' (set to 'Global'), 'Secret' (containing a redacted token), 'ID' (set to 'slack_token'), 'Workspace' (set to 'goodbirdhq'), 'Credential' (set to 'slack_token'), and 'Default channel / member id' (set to '#jenkins-build-alert'). There is also a '+ Add' button for adding more credentials.

▲ Jenkins Slack Token 설정

Jenkins+Slack 알림



```
post {  
    success {  
        slackSend (  
            channel: SLACK_CHANNEL,  
            color: SLACK SUCCESS COLOR,  
            message: "ACR Push에 성공하였습니다."  
        )  
    }  
    failure {  
        slackSend (  
            channel: SLACK_CHANNEL,  
            color: SLACK FAIL COLOR,  
            message: "ACR Push에 실패하였습니다."  
        )  
    }  
}  
  
-----  
  
post {  
    success {  
        slackSend (  
            channel: SLACK_CHANNEL,  
            color: SLACK SUCCESS COLOR,  
            message: "배포에 성공하였습니다."  
        )  
    }  
    failure {  
        slackSend (  
            channel: SLACK_CHANNEL,  
            color: SLACK FAIL COLOR,  
            message: "배포에 실패하였습니다."  
        )  
    }  
}
```



jenkins-build-alert

2 캔버스

+ 책갈피 추가

어제

오늘

jenkins 오전 1:33 ACR Push에 성공하였습니다.
ACR Push에 성공하였습니다.
배포에 성공하였습니다.
배포에 성공하였습니다.
ACR Push에 성공하였습니다.
배포에 성공하였습니다.

jenkins 오전 1:41 ACR Push에 성공하였습니다.
배포에 성공하였습니다.

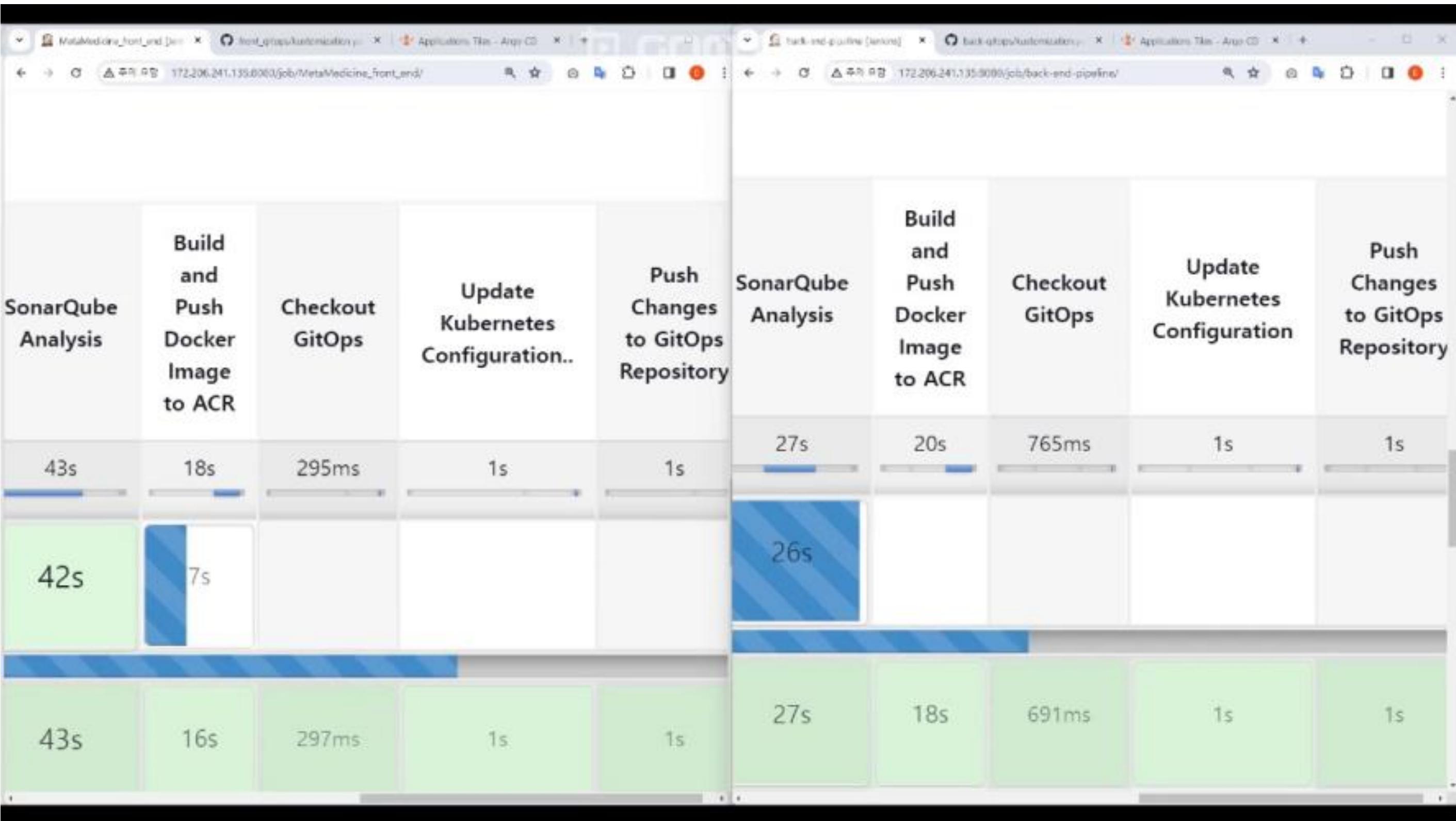
jenkins 오전 1:50 ACR Push에 성공하였습니다.
배포에 성공하였습니다.

새 항목

jenkins 오전 1:59 ACR Push에 성공하였습니다.
배포에 성공하였습니다.

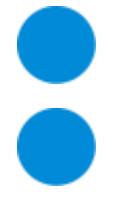
CI 시연 영상 (Build+Slack)

MetaMedical
메타5형제



CI 시연 영상(Build+Slack)

MetaMedical
메타5형제



KOSA 한국소프트웨어산업협회

Metanet

Microsoft Azure

리소스, 서비스 및 문서 검색(G+/)

jiwoo@student.hywom...
기본 디렉토리(GJS1945NAVE...)

홈 > goodbirdacr | 리포지토리 >

goodbirdacr | 리포지토리

컨테이너 레지스트리

검색 새고침 삭제된 리포지토리 관리

개요 활동 로그 액세스 제어(IAM) 태그 빠른 시작 이벤트

설정 액세스 키 암호화 ID 네트워킹 클라우드용 Microsoft Defender 속성 잠금

서비스 리포지토리 Webhooks 지역에서 복제

ACR의 새로운 기능인 아티팩트 스트리밍은 AKS 클러스터에서 이미지를 더 빠르게 가져올 수 있도록 도와줍니다. '아티팩트 스트리밍 상태' 열에는 이 기능을 사용하고 있는 리포지토리가 표시됩니다. 자세히

검색하여 리포지토리 필터링...

리포지토리 ↑↓

- back
- eunjitest_image/front
- front-end
- medical/back
- medical/front
- medicine/back
- medicine/front
- metahospital/front
- trivy/front

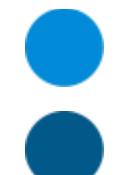
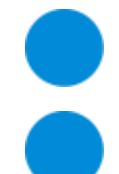
medical/back

리포지토리 삭제

버전	Commit Hash	작성일	...
v1.0.Beta111	sha256:4be771097a8740939c47ef72b4506f968...	2024. 2. 20. 오후 10:36 GMT+9	...
v1.0.Beta110	sha256:1ea7a4b086551bb60388e537e62905ee...	2024. 2. 20. 오후 10:35 GMT+9	...
v1.0.Beta109	sha256:0011b07e64107991d368094f68ebf128...	2024. 2. 20. 오후 10:32 GMT+9	...
v1.0.Beta26	sha256:90740c57283b3b1316558ca11b84a067...	2024. 2. 20. 오후 10:28 GMT+9	...
v1.0.Beta25	sha256:2ba3d9f74aa9015cf49f96bcb97870d3c...	2024. 2. 20. 오후 10:23 GMT+9	...
v1.0.Beta23	sha256:44c4608cb9dc5662a25438c352ade60b...	2024. 2. 20. 오후 9:13 GMT+9	...
v1.0.Beta22	sha256:67e52d97926b2289863a1899c2d5fde1...	2024. 2. 20. 오후 9:02 GMT+9	...
v1.0.Beta21	sha256:0347b8efca8d8883ccc53da3fde70d2b8...	2024. 2. 20. 오후 9:00 GMT+9	...
v1.0.Beta18	sha256:b6675c80b7c8df0445b4dc733550427f...	2024. 2. 20. 오후 8:14 GMT+9	...
v1.0.Beta19	sha256:a2fc3026dcc94c8cbc404ad5883b355...	2024. 2. 20. 오후 7:21 GMT+9	...
v1.0.Beta17	sha256:cc6130da677f99361ae43c4bb79a2235...	2024. 2. 20. 오후 7:15 GMT+9	...
v1.0.Beta15	sha256:fa24a2d310d94abdc5663e1e7fb4fcfc6...	2024. 2. 20. 오후 5:57 GMT+9	...
v1.0.Beta14	sha256:f25387088aeaabb81a40ac0ad3e088a7...	2024. 2. 20. 오후 4:37 GMT+9	...
v1.0.Beta13	sha256:b38a095295e88899beaa2ee4ceeffa2ff6...	2024. 2. 20. 오후 4:26 GMT+9	...
dev-9	sha256:1b86cde404abd7c3aefca28f8a6199fa3...	2024. 2. 20. 오전 11:34 GMT+9	...
dev-8	sha256:0e72567e281bd8349c2a511dc1070faf4...	2024. 2. 20. 오전 10:59 GMT+9	...
prod-2	sha256:0eaa6bd9a1ca1d3ebf196621e9b815c9...	2024. 2. 20. 오전 10:52 GMT+9	...
stg-4	sha256:751a0fffcfb1871ae69299d39e1094833...	2024. 2. 20. 오전 10:52 GMT+9	...
dev-7	sha256:f9cea067eb750fac890b94e0b0c87e014...	2024. 2. 20. 오전 10:51 GMT+9	...

Trivy

MetaMedical
메타5형제



이미지 빌드 시 스캔 및 스캔 결과에 따른 빌드 Fail/Success

Scanner	OS Packages	Application Dependencies	Easy to use	Accuracy	Suitable for CI
Trivy	✓	✓ (8 languages)	★ ★ ★	★ ★ ★	★ ★ ★
Clair	✓	x	★	★ ★	★ ★
Anchore Engine	✓	✓ (4 languages)	★ ★	★ ★	★ ★ ★
Quay	✓	x	★ ★ ★	★ ★	x
Docker Hub	✓	x	★ ★ ★	★	x
GCR	✓	x	★ ★ ★	★ ★	x

▲ <https://aquasecurity.github.io/trivy/v0.17.2/comparison/>



Trivy

```

#!/bin/bash
## Docker 이미지 이름
dockerImageName=$(awk 'NR==1 {print $2}' Dockerfile)
echo $dockerImageName

## Trivy 이미지 스캔 실행
# -v $WORKSPACE:/root/.cache/: Trivy 캐시를 Jenkins 워크스페이스로 캐시 공유
# -q: Trivy의 출력 간소화
# --exit-code 1: 취약점이 발견되면 'exit_code' 1 반환
# --severity CRITICAL: 심각도가 CRITICAL인 취약점만 보고
# --light: 가볍게 스캔, 패키지 버전만 확인
docker run --rm -v $WORKSPACE:/root/.cache/ -e
TRIVY_GITHUB_TOKEN="ghp_vduwdvYQwDqs91Umsr2sVyD7ik1TF3ldNoo"
aquasec/trivy:0.17.2 -q image --exit-code 1
--severity CRITICAL --light $dockerImageName

## Trivy 스캔 결과 처리
exit_code=$?
echo "종료 코드: $exit_code"
#
## 스캔 결과 확인
if [[ "$exit_code" == 1 ]]; then
    echo "이미지 스캔에 실패했습니다. CRITICAL 취약점이 발견되었습니다."
    exit 1
else
    echo "이미지 검사가 통과되었습니다. CRITICAL 취약점이 발견되지 않았습니다."
fi

```

O X 종료 코드: 1

./trivy-image-scan.sh: 12: [: not found

이미지 검사가 통과되었습니다. 심각한 취약점이 발견되지 않았습니다.

Total: 5 (CRITICAL: 5)

LIBRARY FIXED VERSION	VULNERABILITY ID	SEVERITY	INSTALLED VERSION
glibc	CVE-2019-9169	CRITICAL	2.28-225.0.3.el8 2:2.28-151.0.1.kssplice2.el8
glibc-common			
glibc-minimal-langpack			
gnutls	CVE-2021-20231	3.6.16-6.el8_7 10:3.6.16-4.0.1.el8_fips	
	CVE-2021-20232		

종료 코드: 1

이미지 스캔에 실패했습니다. 심각한 취약점이 발견되었습니다.

Average stage times:	Declarative: Checkout SCM	Checkout	Grant Execute Permission to Gradle Wrapper	Build JAR	Trivy Security	Build and Push Docker Image to ACR	Push Changes to GitOps Repository
3 commits	299ms	201ms	341ms	15s	1s	78ms	74ms
	299ms	201ms	341ms	15s	1s	78ms	74ms

CICD

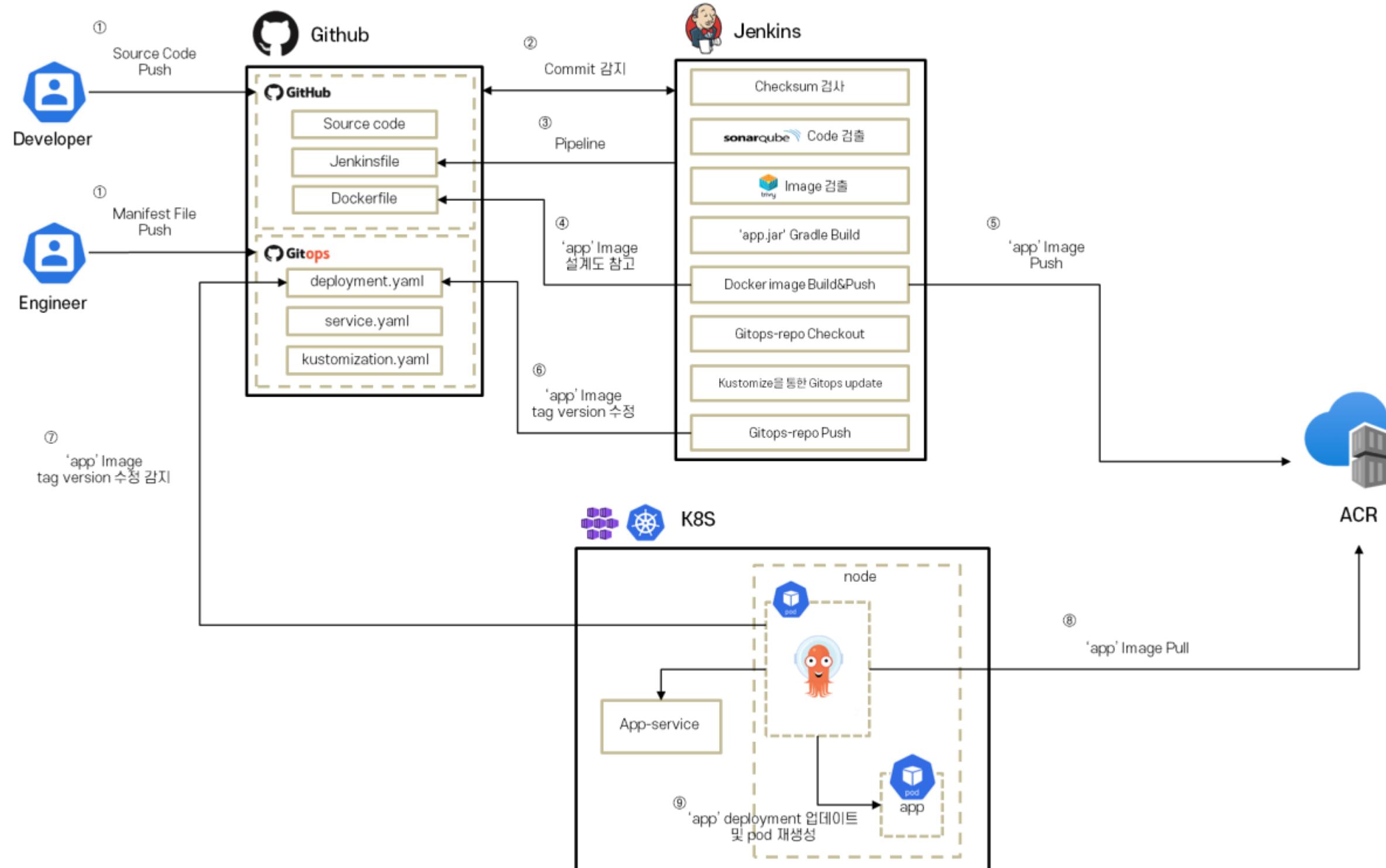
ArgoCD 기반 CD 자동화 및 코드 점검



발표자: 이은지

CICD Architecture

MetaMedical
메타5형제

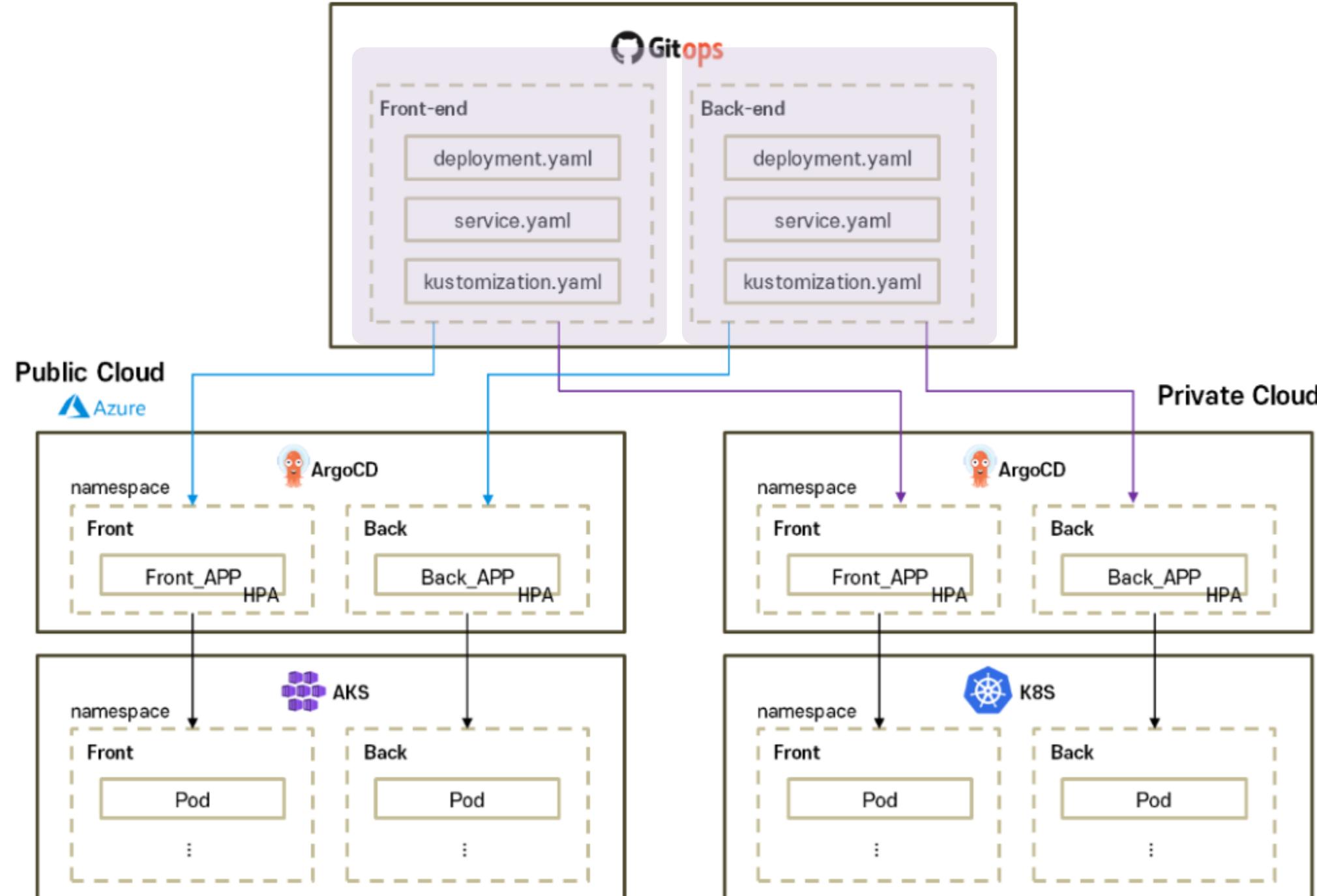


Front – Back 배포 과정

MetaMedical
메타5형제

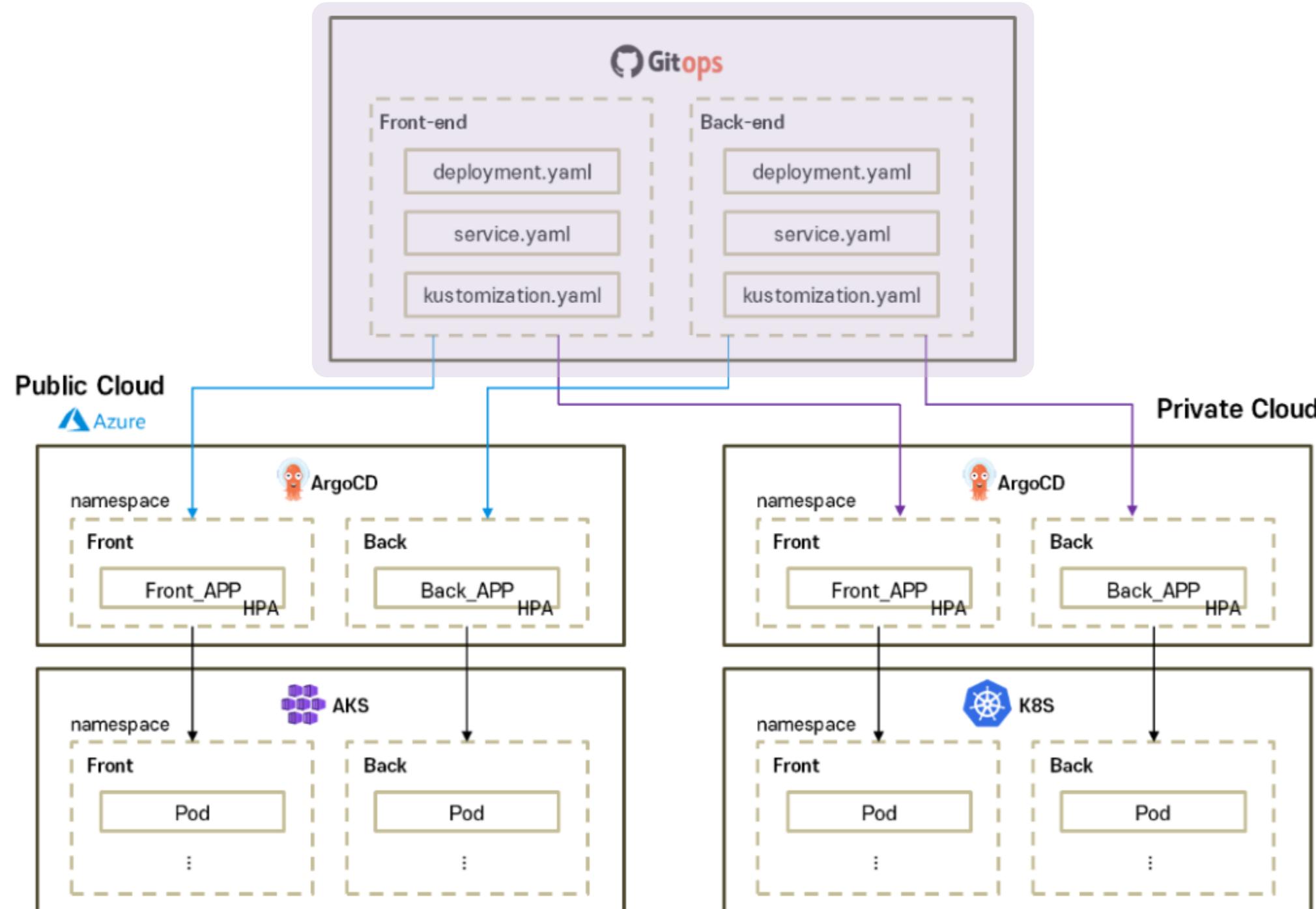


●
●
●
●
●



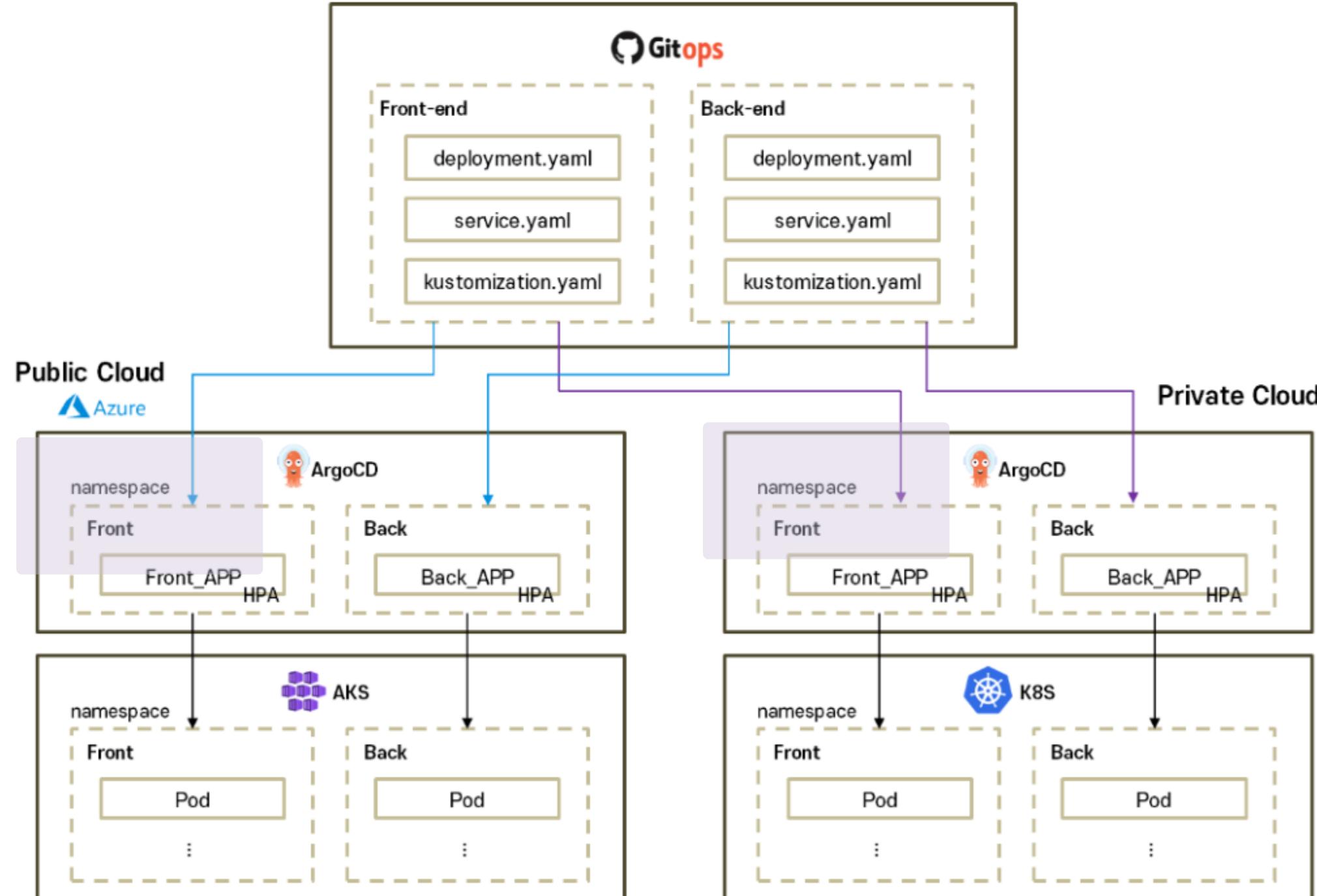
1. Front repo – Back repo로 분리
 2. Code repo – K8S Cluster repo로 분리
- 유지보수와 확장성을 향상
- Namespace로 argocd application 구분
- AKS에 Namespace로 구분하여 배포
- ★ Hybrid Cloud를 트래픽에 따라
탄력적으로 확장하기 위한 인프라 구축

Front – Back 배포 과정



1. Front repo – Back repo로 분리
 2. Code repo – K8S Cluster repo로 분리
- 유지보수와 확장성을 향상
- Namespace로 argocd application 구분
- AKS에 Namespace로 구분하여 배포
- ★ Hybrid Cloud를 트래픽에 따라
탄력적으로 확장하기 위한 인프라 구축

Front – Back 배포 과정



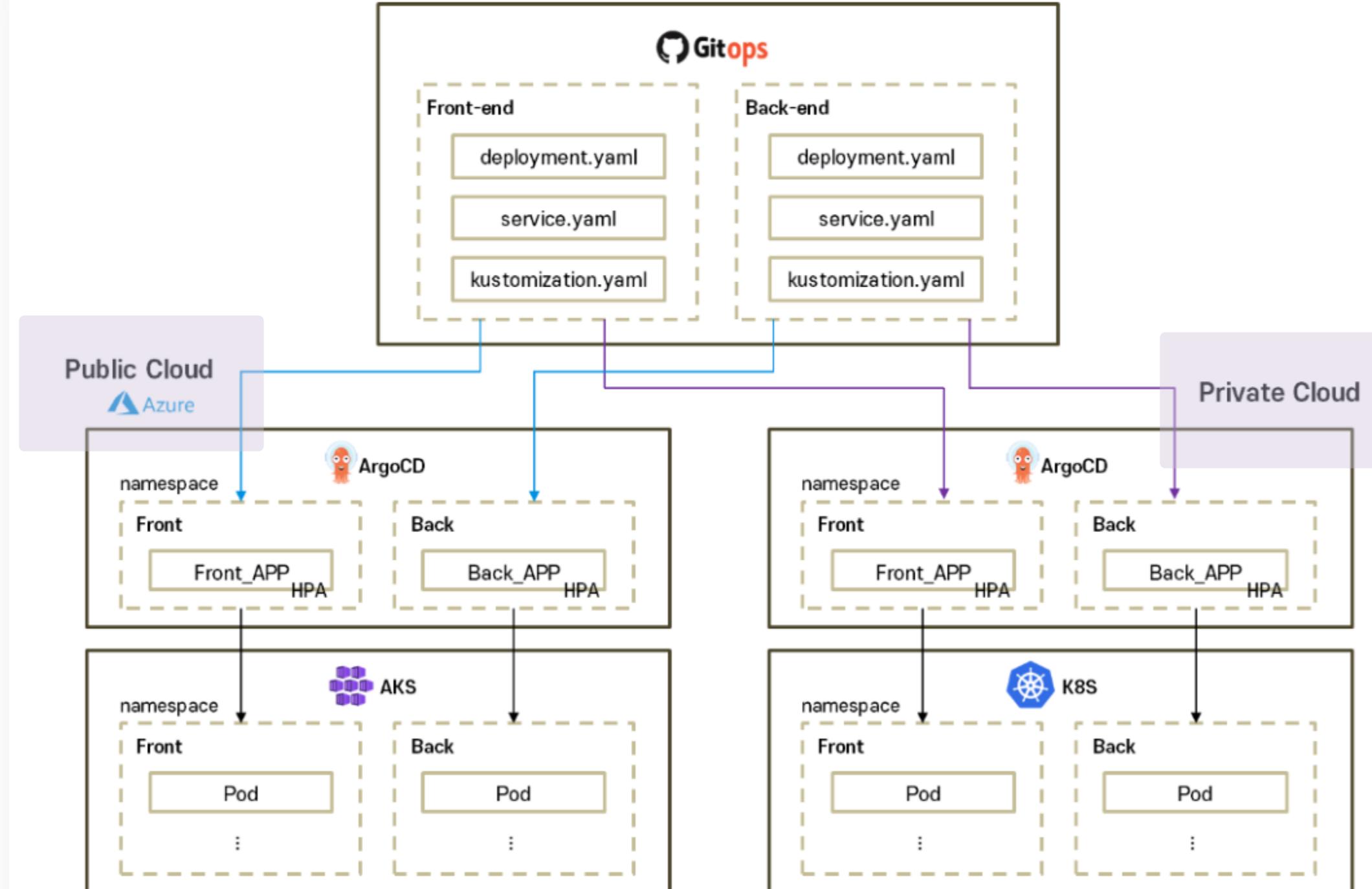
1. Front repo – Back repo로 분리
 2. Code repo – K8S Cluster repo로 분리
- 유지보수와 확장성을 향상

Namespace로 argocd application 구분

→ AKS에 Namespace로 구분하여 배포

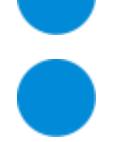
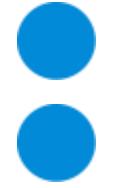
★ Hybrid Cloud를 트래픽에 따라
탄력적으로 확장하기 위한 인프라 구축

Front – Back 배포 과정



SonarQube - 코드 검증

MetaMedical
메타5형제



Front

The screenshot shows the SonarQube interface for the 'main' branch of the 'metahospital_project'. The top navigation bar includes links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search bar. The main content area displays the following information:

- Quality Gate Status:** Passed (green checkmark icon).
- Measures:** New Code (0), Overall Code (927 Lines of Code). Last analysis: 18 hours ago.
- New Issues:** 0 (Required: 0). Accepted Issues: 0 (Valid issues that were not fixed).
- Coverage:** 50.0% (Required: 3.0%). On 0 new lines to cover.
- Duplications:** 50.0% (Required: 3.0%). On 4 new lines.
- Security Hotspots:** 0.
- ACTIVITY:** A chart showing activity over the last 24 hours with no data points.
- History:** A table of analysis results from February 15, 2024:
 - 9:28 PM: NOT PROVIDED, 0 issues, Quality Gate: Passed.
 - 8:44 PM: NOT PROVIDED, 0 issues, Quality Gate: Passed.
 - 7:07 PM: NOT PROVIDED, 0 issues, Quality Gate: Passed.
 - 6:51 PM: First analysis, 0 issues, 0.0% Coverage, 15.0% Duplications, Quality Gate: Passed.

Back

The screenshot shows the SonarQube interface for the 'main' branch of the 'metahospital_back_project'. The top navigation bar includes links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search bar. The main content area displays the following information:

- Quality Gate Status:** Passed (green checkmark icon).
- Measures:** New Code (491 Lines of Code). Last analysis: 17 hours ago.
- Security:** 3 Open Issues (Severity: C). Reliability: 0 Open Issues. Maintainability: 13 Open Issues (Severity: A).
- Accepted issues:** 0 (Valid issues, but not fixed).
- Coverage:** 0.0% (On 66 new lines to cover).
- Duplications:** 20.6% (On 491 new lines).
- Security Hotspots:** 2.
- ACTIVITY:** A chart showing activity over the last 24 hours with no data points.
- History:** A table of analysis results from February 15, 2024:
 - 10:17 PM: NOT PROVIDED, 0 issues, Quality Gate: Passed.

Kustomize



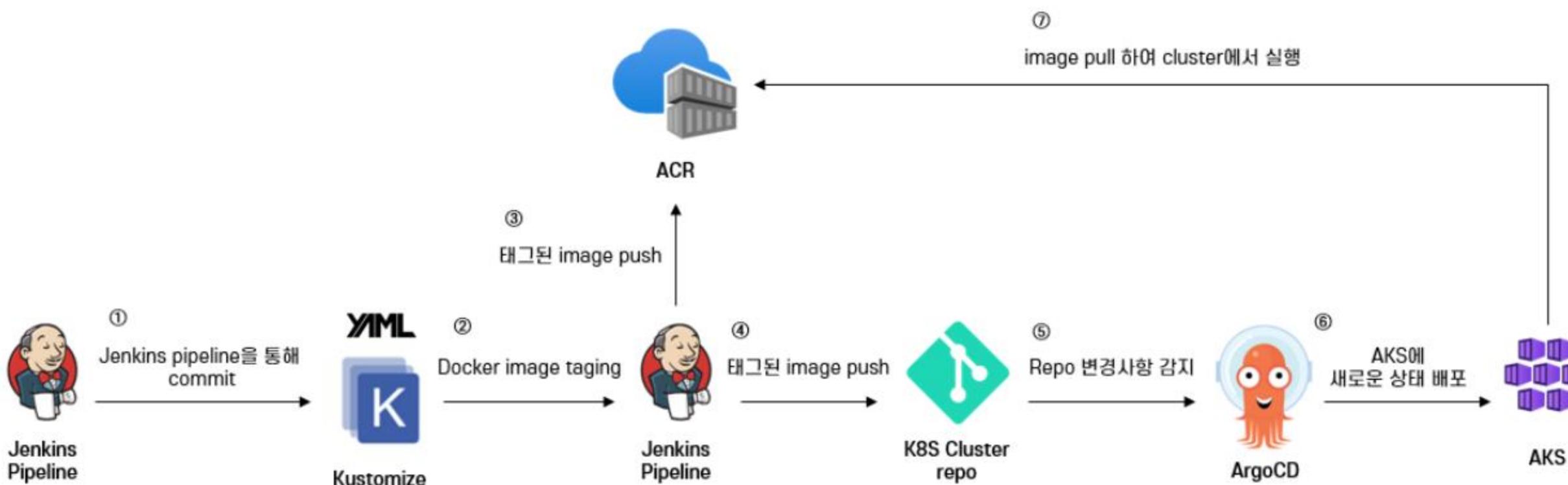
```
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization

resources:
- deployment.yaml
- service.yaml

images:
- name: goodbirdacr.azurecr.io/medicine/front
  newName: goodbirdacr.azurecr.io/medicine/front
  newTag: v1.0.Beta4
```

- Kustomization.yaml 파일
- Docker image의 tag 변경

→ ArgoCD가 변경사항 감지



Kustomize



Jenkins
Pipeline

Jenkinsfile 환경설정

```
environment {
    AZURE_SUBSCRIPTION_ID = 'c8ce3edc-0522-48a3-b7e4-afe8e3d731d9'
    AZURE_TENANT_ID = '4ccd6048-181f-43a0-ba5a-7f48e8a4fa35'
    CONTAINER_REGISTRY = 'goodbirdacr.azurecr.io'
    RESOURCE_GROUP = 'AKS'
    REPO = 'medicine/front'
    IMAGE_NAME = 'medicine/front:latest'

    TAG_VERSION = "v1.0.Beta"
    TAG = "${TAG_VERSION}${env.BUILD_ID}"
    NAMESPACE = 'front'
    GIT_CREDENTIALS_ID = 'jenkins-git-access'
}
```

Jenkinsfile Kustomize 실행

```
stage('Update Kubernetes Configuration..') {
    steps {
        script {
            sh "kustomize edit set image ${CONTAINER_REGISTRY}/${REPO}=${CONTAINER_REGISTRY}/${REPO}:${TAG}"
            sh "git add ."
            sh "git commit -m 'Update image to ${TAG}'"
        }
    }
}
```



Github
K8S Cluster repo

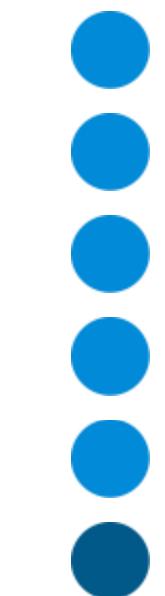
Kustomization.yaml 파일

```
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization

resources:
- deployment.yaml
- service.yaml

images:
- name: goodbirdacr.azurecr.io/medicine/front
  newName: goodbirdacr.azurecr.io/medicine/front
  newTag: latest
```

Kustomize



Jenkins
Pipeline

```
stage('Build and Push Docker Image to ACR') {  
    steps {  
        script {  
            withCredentials([usernamePassword(credentialsId: 'acr-credential-id', passwordVariable: 'ACR_PASSWORD', usernameVariable: 'ACR_USERNAME')]) {  
                sh "az acr login --name $CONTAINER_REGISTRY --username $ACR_USERNAME --password $ACR_PASSWORD"  
  
                sh "docker build -t $CONTAINER_REGISTRY/$REPO:$TAG ."  
                sh "docker push $CONTAINER_REGISTRY/$REPO:$TAG"  
            }  
        }  
    }  
}
```



ACR

medicine/front

리포지토리

새로 고침

삭제된 아티팩트 관리

리포지토리 삭제

▲ 기본 정보

리포지토리

: medicine/front

마지막으로 업데이트한 ...

: 2024. 2. 19. 오전 6:13 GMT+9

검색하여 태그 필터링...

태그 ↑↓

latest

v1.0.Beta4

- Jenkinsfile에서 ACR에 push

→ IMAGE 태그 변경

Kustomize

MetaMedical
메타5형제



Jenkins
Pipeline



```
stage('Push Changes to GitOps Repository') {
    steps {
        script {
            withCredentials([usernamePassword(credentialsId: 'jenkins-git-access', passwordVariable: 'GIT_PASSWORD', usernameVariable: 'GIT_USERNAME')]) {
                // 현재 브랜치 확인 및 main으로 체크아웃
                def currentBranch = sh(script: "git rev-parse --abbrev-ref HEAD", returnStdout: true).trim()
                if (currentBranch != "main") {
                    sh "git checkout main"
                }
                // 원격 저장소에서 최신 변경사항 가져오기
                sh "git pull --rebase origin main"
                def remote = "https://${GIT_USERNAME}:${GIT_PASSWORD}@github.com/JoEunSae/front-end.git"
                // 원격 저장소에 푸시
                sh "git push https://${GIT_USERNAME}:${GIT_PASSWORD}@github.com/rlozi99/front_gitops.git main"
            }
        }
    }
}
```



front_gitops / kustomization.yaml

Jenkins and Jenkins Update image to v1.0.Beta4

Code Blame 11 lines (9 loc) · 234 Bytes Code 55% faster with GitHub



Github
K8S Cluster repo

```
1  apiVersion: kustomize.config.k8s.io/v1beta1
2  kind: Kustomization
3
4  resources:
5    - deployment.yaml
6    - service.yaml
7
8  images:
9    - name: goodbirdacr.azurecr.io/medicine/front
10   newName: goodbirdacr.azurecr.io/medicine/front
11   newTag: v1.0.Beta4
```

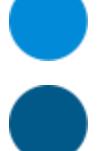
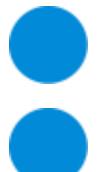
- Jenkinsfile에서 git에 push



Kustomization.yaml 파일 태그 변경

ArgoCD Slack 알림

MetaMedical
메타5형제



ArgoCD 웹 오후 11:28

✓ Application back-gitops-app has been successfully synced at 2024-02-18T14:28:25Z.

Sync operation details are available at: <https://localhost:4000/applications/back-gitops-app?operation=true>.

back-gitops-app

Sync Status

Synced

Repository

<https://github.com/rlozi99/back-gitops.git>

✓ Application back-gitops-app is now running new version of deployments manifests.

back-gitops-app

Sync Status

Synced

Repository

<https://github.com/rlozi99/back-gitops.git>

Revision

8803ff2aa5812f7fc7c3331368d3a69c2a8

c31a8



ArgoCD 웹 오전 5:15

! The sync operation of application front-gitops-app has failed at 2024-02-17T20:15:20Z with the following error: one or more synchronization tasks are not valid (retried 5 times).

Sync operation details are available at: <https://localhost:4000/applications/front-gitops-app?operation=true>.

front-gitops-app

Sync Status

OutOfSync

Repository

https://github.com/rlozi99/front_gitops.git



ArgoCD 웹 오전 5:20

! The sync operation of application front-gitops-app has failed at 2024-02-17T20:20:31Z with the following error: one or more synchronization tasks are not valid (retried 5 times).

Sync operation details are available at: <https://localhost:4000/applications/front-gitops-app?operation=true>.

front-gitops-app

Sync Status

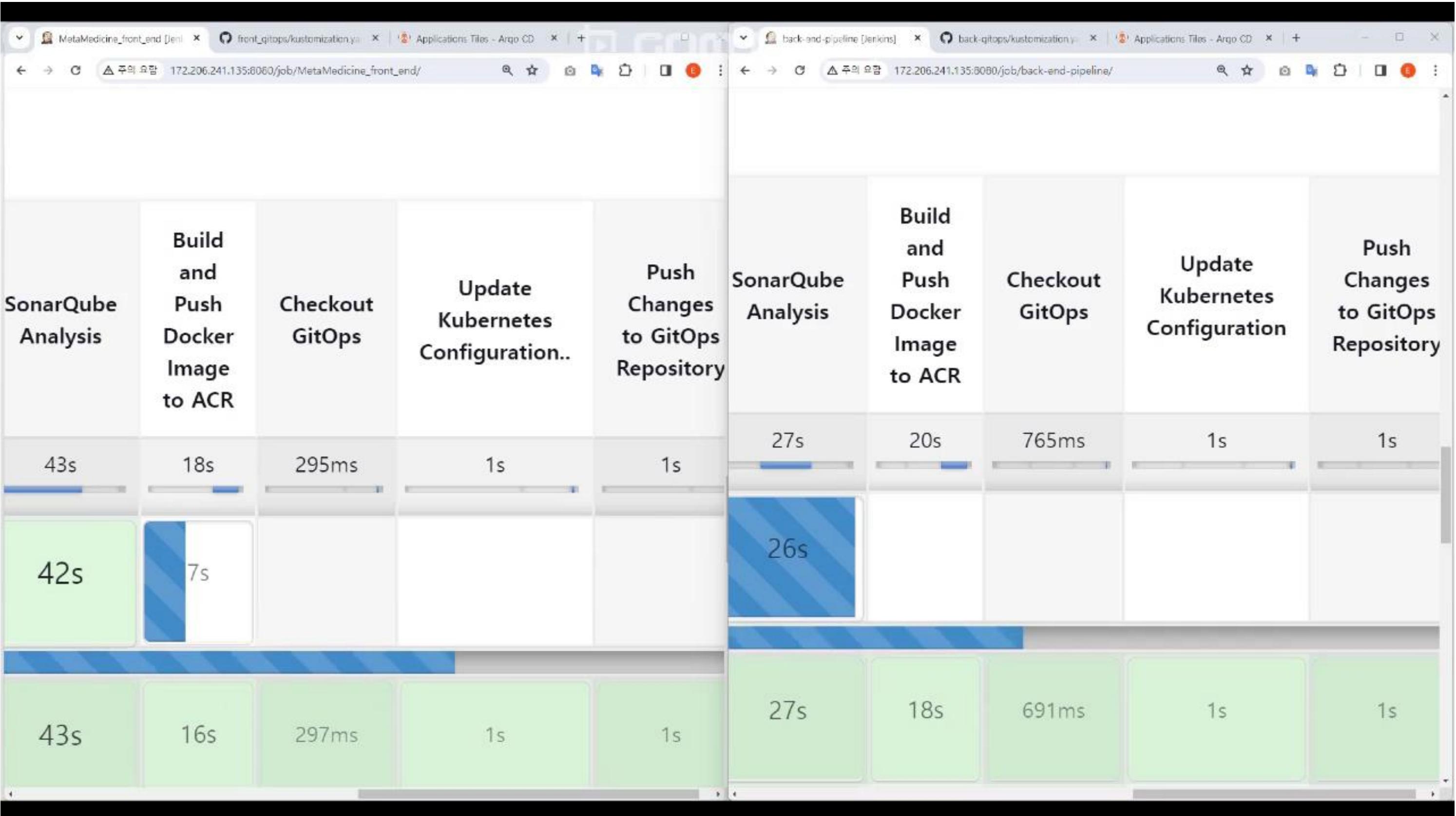
OutOfSync

Repository

https://github.com/rlozi99/front_gitops.git

시연 영상

MetaMedical
메타5형제





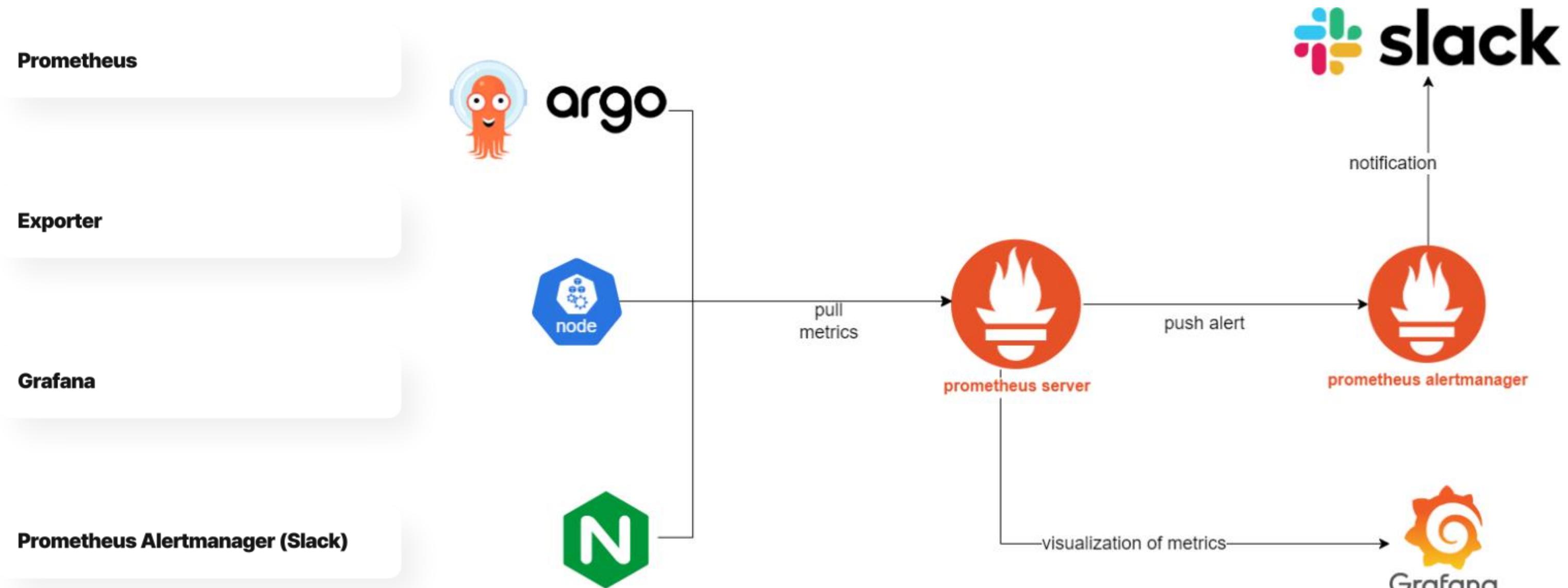
Monitoring

서비스 성능 및 장애 관리

발표자: 육지현

Monitoring Architecture

MetaMedical
메타5형제



Exporter 설치



Nginx exporter

```
- name: metrics
  image: nginx/nginx-prometheus-exporter
  resources:
    requests:
      cpu: 10m
      memory: 50Mi
    limits:
      cpu: 100m
      memory: 50Mi
  ports:
    - containerPort: 9113

- name: metrics
  protocol: TCP
  port: 9113
  targetPort: metrics
```

Nginx

```
location /metrics {
  stub_status on;
  access_log off;
  allow all;
}

- name: metrics
  args:
    - --nginx.scrape-uri
    - http://127.0.0.1:80/metrics
  image: nginx/nginx-prometheus-exporter
  resources:
    requests:
      cpu: 10m
      memory: 100Mi
    limits:
      cpu: 30m
      memory: 500Mi
  ports:
    - containerPort: 9113
  imagePullSecrets:
    - name: goodbirdacr
```

Exporter 설치



Mysql

```
spec:  
  containers:  
    - name: mysqld-exporter  
      image: prom/mysqld-exporter:v0.12.1  
      args:  
        - --collect.info_schema.tables  
        - --collect.info_schema.innodb_tablespaces  
        - --collect.info_schema.innodb_metrics  
        - --collect.global_status  
        - --collect.global_variables  
        - --collect.slave_status  
        - --collect.info_schema.processlist  
        - --collect.perf_schema.tablelocks  
        - --collect.perf_schema.eventsstatements  
        - --collect.perf_schema.eventsstatementssum  
        - --collect.perf_schema.eventswaits  
        - --collect.auto_increment.columns  
        - --collect.binlog_size  
        - --collect.perf_schema.tableiowaits  
        - --collect.perf_schema.indexiowaits  
        - --collect.info_schema.userstats  
        - --collect.info_schema.clientstats  
        - --collect.info_schema.tablestats  
        - --collect.info_schema.schemastats  
        - --collect.perf_schema.file_events  
        - --collect.perf_schema.file_instances  
        - --collect.perf_schema.replication_group_member_stats  
        - --collect.perf_schema.replication_applier_status_by_worker  
        - --collect.slave_hosts  
        - --collect.info_schema.innodb_cmp  
        - --collect.info_schema.innodb_cmpmem  
        - --collect.info_schema.query_response_time  
        - --collect.engine_tokudb_status  
        - --collect.engine_innodb_status  
      ports:  
        - containerPort: 9104  
          protocol: TCP  
      env:  
        - name: DATA_SOURCE_NAME  
          value: "mysqld-exporter:123456@(mysql-svc:3306)/*"
```

Node exporter

```
apiVersion: apps/v1  
kind: DaemonSet  
metadata:  
  name: node-exporter  
  namespace: monitoring  
  labels:  
    k8s-app: node-exporter  
spec:  
  selector:  
    matchLabels:  
      k8s-app: node-exporter  
  template:  
    metadata:  
      labels:  
        k8s-app: node-exporter  
    spec:  
      containers:  
        - name: node-exporter  
          image: quay.io/prometheus/node-exporter:v1.2.2  
          ports:  
            - name: http  
              containerPort: 9100
```

Prometheus



- 메트릭 수집 대상 지정

```
- job_name: 'nginx'
  static_configs:
    - targets: ['front-end-service.front.svc:9113']

- job_name: 'node-exporter'
  static_configs:
    - targets: ['node-exporter.exporter.svc:9100']

- job_name: 'argocd'
  static_configs:
    - targets: ['argocd-server-metrics.argocd.svc:8083', 'argocd-repo-server.argocd.svc:8084']
```

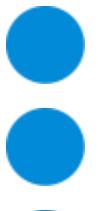
- alertmanager 설정

```
config:
  global:
    resolve_timeout: 5m
    slack_api_url: 'https://hooks.slack.com/services/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'
  receivers:
    - name: 'slack-notifications'
      slack_configs:
        - channel: '# project-alert'
          send_resolved: true
```

```
- name: node-exporter
  rules:
    - alert: HighCPUUsage
      expr: sum without(mode) (avg without (cpu) (rate(node_cpu_seconds_total{job="prometheus", mode!="idle"}[2m])) * 100 > 80
      for: 5m
      labels:
        severity: warning
      annotations:
        description: 'The Nginx server is experiencing a high connection rate.'
        summary: 'High Nginx Site Access'
```

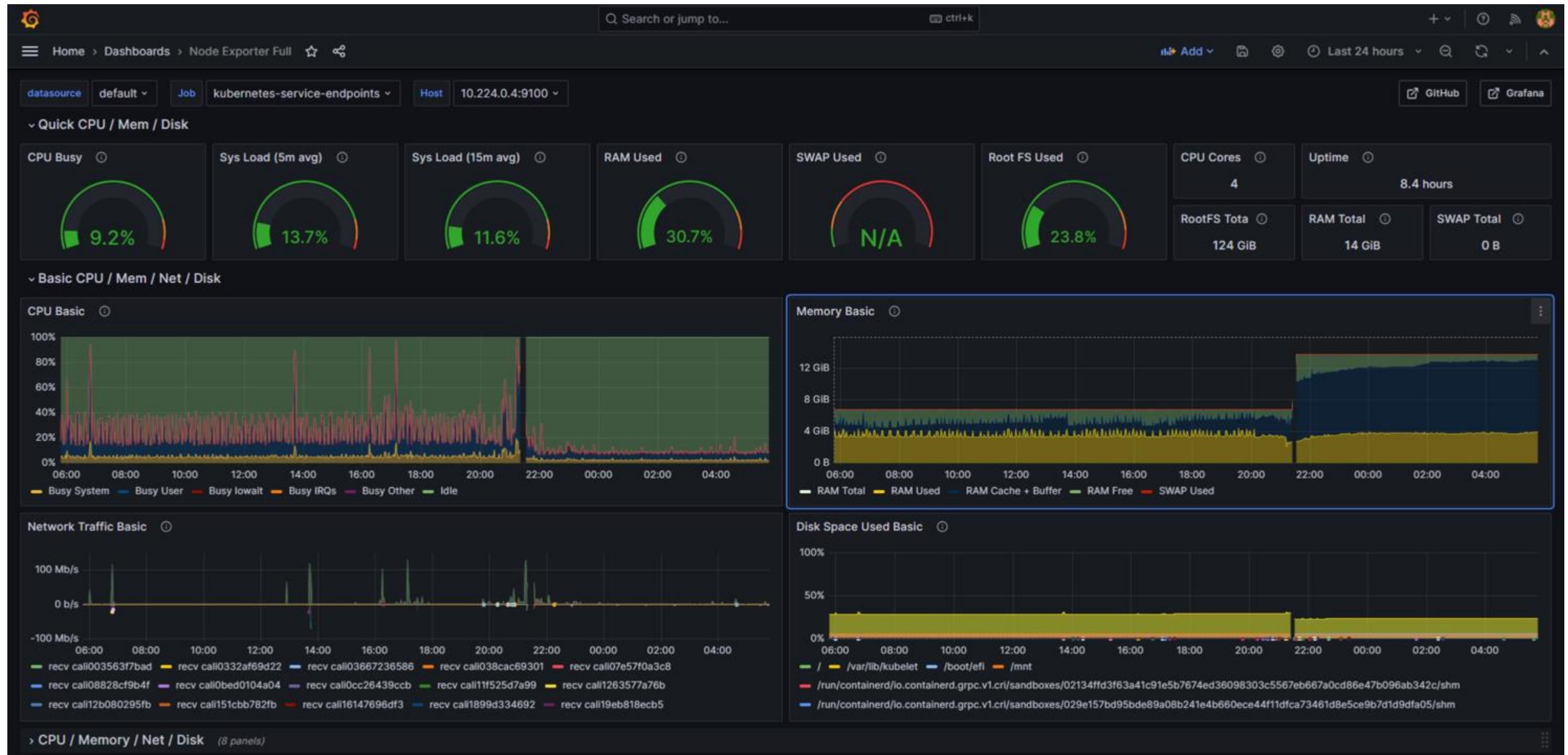
Prometheus Metrics&Alert status

MetaMedical
메타5형제



Grafana

MetaMedical
메타데이터



Alert manager – slack



The screenshot shows the Prometheus Alerting interface and a corresponding Slack channel.

Prometheus Alerting Interface:

- Header:** Prometheus, Alerts, Graph, Status, Help.
- Status Filter:** Inactive (0), Pending (0), Firing (1).
- Search:** Filter by name or labels.
- Alert Rule:** /etc/config/alerting_rules.yml > nginx
- Alert Details:** NginxAccessHigh (2 active)
 - Rule Definition:** name: NginxAccessHigh, expr: nginx_http_requests_total > 20000, for: 5m, labels: severity: warning, annotations: description: The Nginx server is experiencing a high connection rate., summary: High Nginx Site Access.

Slack Channel (# project-alert):

- Message Headers:** + 책갈피 추가, ↑ 20 새 메시지 ×.
- Messages:**
 - [FIRING:1] HighCPUUsage (node-exporter.exporter.svc:9100 prometheus warning) - Alertmanager (오전 2:10)
 - [RESOLVED] HighCPUUsage (node-exporter.exporter.svc:9100 prometheus warning) - Alertmanager (오전 2:15)
 - [FIRING:1] HighCPUUsage (node-exporter.exporter.svc:9100 prometheus warning) - Alertmanager (오전 2:41)
 - [RESOLVED] HighCPUUsage (node-exporter.exporter.svc:9100 prometheus warning) - Alertmanager (오전 2:46)
 - [FIRING:1] NginxAccessHigh (nginx.ngxex.svc:9113 nginx warning) - Alertmanager (오전 5:32)
- Date Filter:** 오늘.

모니터링 시연 영상(Nginx+node+Argo+DB)

MetaMedical
메타5형제



The screenshot shows a GitHub repository page for 'MetaMedicine_front_end'. The repository has 23 commits, 4 branches, and 0 tags. The 'About' section includes a description of the repo, a link to the Readme, activity stats (0 stars, 1 watching, 1 fork), and a link to report the repository. The 'Releases' section indicates no releases have been published, with a link to create a new release. The 'Packages' section shows no packages have been published, with a link to publish your first package. The 'Contributors' section lists two contributors: 'rl0z99' and 'jihyeon8822'. The 'Languages' section shows the code distribution: HTML (65.3%), CSS (22.0%), JavaScript (11.0%), and Dockerfile (1.7%).