

한국 마이크로소프트

# Microsoft Technical Trainer

Enterprise Skills Initiative

**AZ-104. LAB09C**

## **Azure Kubernetes Service 구현**

이 문서는 Microsoft Technical Trainer팀에서 ESI 교육 참석자분들에게 제공해 드리는 문서입니다.

**요약**

이 내용들은 표시된 날짜에 Microsoft에서 검토된 내용을 바탕으로 하고 있습니다. 따라서, 표기된 날짜 이후에 시장의 요구사항에 따라 달라질 수 있습니다. 이 문서는 고객에 대한 표기된 날짜 이후에 변화가 없다는 것을 보증하지 않습니다.

이 문서는 정보 제공을 목적으로 하며 어떠한 보증을 하지는 않습니다.

저작권에 관련된 법률을 준수하는 것은 고객의 역할이며, 이 문서를 마이크로소프트의 사전 동의 없이 어떤 형태(전자 문서, 물리적인 형태 막론하고) 어떠한 목적으로 재 생산, 저장 및 다시 전달하는 것은 허용되지 않습니다.

마이크로소프트는 이 문서에 들어있는 특허권, 상표, 저작권, 지적 재산권을 가집니다. 문서를 통해 명시적으로 허가된 경우가 아니면, 어떠한 경우에도 특허권, 상표, 저작권 및 지적 재산권은 다른 사용자에게 허여되지 않습니다.

© 2023 Microsoft Corporation All right reserved.

Microsoft®는 미합중국 및 여러 나라에 등록된 상표입니다.

이 문서에 기재된 실제 회사 이름 및 제품 이름은 각 소유자의 상표일 수 있습니다.

## 문서 작성 연혁

날짜	버전	작성자	변경 내용
2021.11.22	1.0.0	우진환	LAB09C 작성
2022.10.08	1.1.0	우진환	Azure 포털 변경 사항 적용
2023.02.09	1.2.0	우진환	Cloudslice 변경 사항 적용
2023.06.03	1.3.0	우진환	Cloudslice 변경 사항 적용

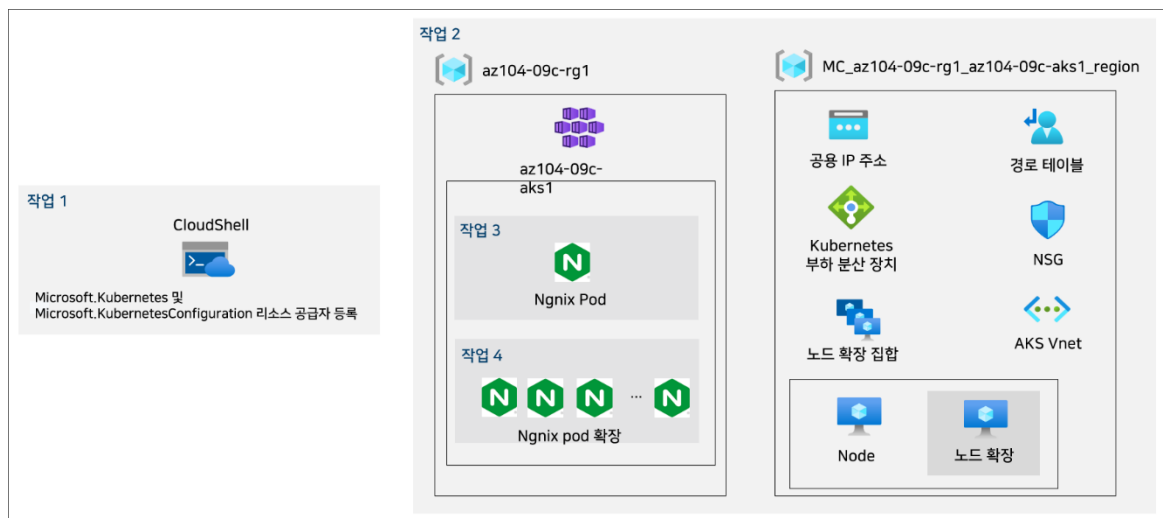
## 목차

실습 시나리오 .....	5
아키텍처 다이어그램 .....	5
TASK 01. CLOUD SHELL 준비.....	5
TASK 02. MICROSOFT.KUBERNETES 및 MICROSOFT.KUBERNETESCONFIGURATION 리소스 공급자 등록.....	6
TASK 03. AZURE KUBERNETES SERVICE 클러스터 배포.....	7
TASK 04. AZURE KUBERNETES SERVICE 클러스터에 POD 배포.....	10
TASK 05. AZURE KUBERNETES SERVICE 클러스터에서 컨테이너화된 워크로드 확장.....	13
TASK 06. 리소스 정리.....	14

## 실습 시나리오

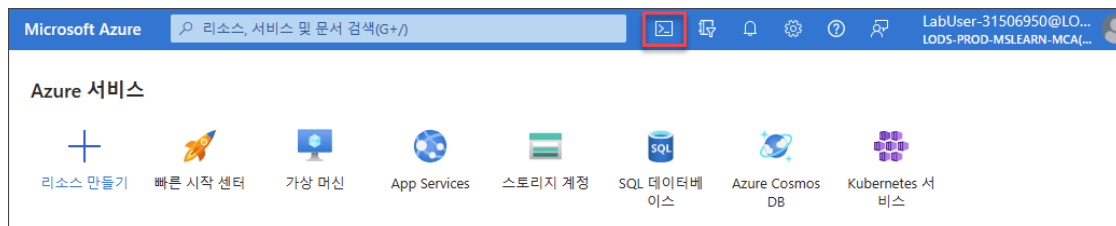
Contoso에는 Azure Container Instances를 사용하여 실행하기에 적합하지 않은 여러 다중 계층 애플리케이션이 있습니다. 컨테이너화된 워크로드로 실행할 수 있는지 여부를 확인하기 위해 Kubernetes를 컨테이너 오케스트레이터로 사용하여 평가하려고 합니다. 관리 오버헤드를 최소화하기 위해 간소화된 배포 환경 및 크기 조정 기능을 포함하여 Azure Kubernetes Service를 테스트하려고 합니다.

## 아키텍처 다이어그램

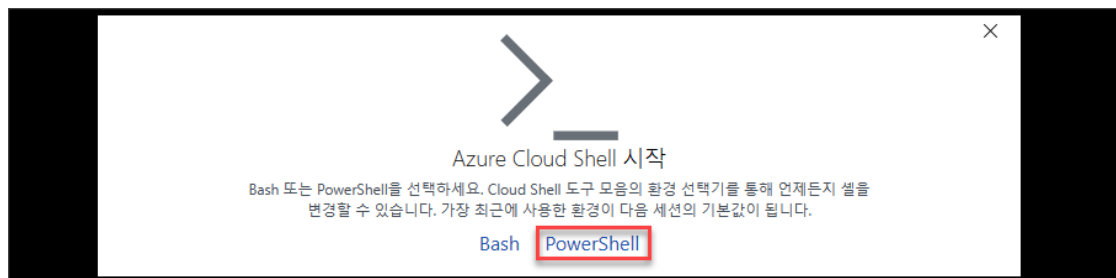


## TASK 01. Cloud Shell 준비

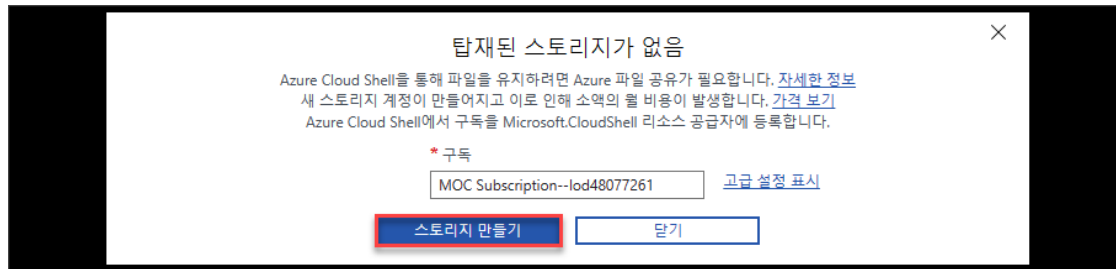
1. Azure 포털의 우측 상단에서 [Cloud Shell] 아이콘을 클릭합니다.



2. [Azure Cloud Shell 시작] 창에서 [PowerShell]을 클릭합니다.



3. [탐재된 스토리지가 없음] 페이지에서 [스토리지 만들기]를 클릭합니다.



## TASK 02. Microsoft.Kubernetes 및 Microsoft.KubernetesConfiguration 리소스 공급자 등록

이 작업에서는 Azure Kubernetes Service 클러스터 배포에 필요한 리소스 공급자를 등록합니다.

1. Azure 포털에서 [Cloud Shell]을 실행합니다. [Cloud Shell]의 PowerShell 세션에서 다음 명령을 실행하여 Kubernetes와 관련된 리소스 공급자를 등록합니다.

```
# Kubernetes 리소스 공급자 등록
Register-AzResourceProvider -ProviderNamespace Microsoft.Kubernetes
Register-AzResourceProvider -ProviderNamespace Microsoft.KubernetesConfiguration

PowerShell
PS /home/labuser-31506950> # Kubernetes 리소스 공급자 등록
PS /home/labuser-31506950> Register-AzResourceProvider -ProviderNamespace Microsoft.Kubernetes

ProviderNamespace : Microsoft.Kubernetes
RegistrationState  : Registering
ResourceTypes     : {connectedClusters, locations, locations/operationStatuses, registeredSubscriptions_}
Locations         : {West Europe, East US, West Central US, South Central US_}

PS /home/labuser-31506950> Register-AzResourceProvider -ProviderNamespace Microsoft.KubernetesConfiguration

ProviderNamespace : Microsoft.KubernetesConfiguration
RegistrationState  : Registering
ResourceTypes     : {sourceControlConfigurations, extensions, fluxConfigurations, operations_}
Locations         : {East US, West Europe, West Central US, West US 2_}

PS /home/labuser-31506950> 
```

2. [Cloud Shell]에서 다음 명령을 실행하여 리소스 공급자가 등록되었는지 확인합니다.

RegistrationState에서 "Registered"로 표시되어야 합니다.

```
# Kubernetes 리소스 공급자 등록 확인
Get-AzResourceProvider -ProviderNamespace Microsoft.Kubernetes

PowerShell
PS /home/labuser-31506950> # Kubernetes 리소스 공급자 등록 확인
PS /home/labuser-31506950> Get-AzResourceProvider -ProviderNamespace Microsoft.Kubernetes

ProviderNamespace : Microsoft.Kubernetes
RegistrationState  : Registered
ResourceTypes     : {connectedClusters}
Locations         : {West Europe, East US, West Central US, South Central US_}

ProviderNamespace : Microsoft.Kubernetes
RegistrationState  : Registered
ResourceTypes     : {locations}
Locations         : {}

ProviderNamespace : Microsoft.Kubernetes
RegistrationState  : Registered
ResourceTypes     : {locations/operationStatuses}
Locations         : {East US 2 EUAP, West Europe, East US, West Central US_}

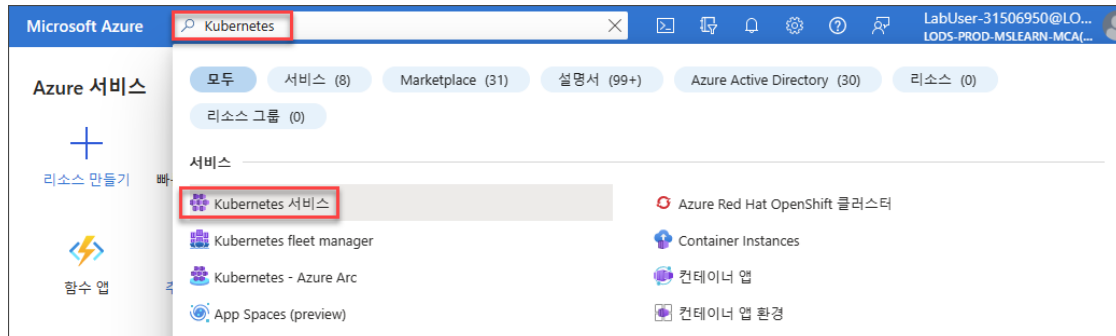
ProviderNamespace : Microsoft.Kubernetes
RegistrationState  : Registered
ResourceTypes     : {registeredSubscriptions}
Locations         : {}

ProviderNamespace : Microsoft.Kubernetes
RegistrationState  : Registered
ResourceTypes     : {operations}
Locations         : {}
```

### TASK 03. Azure Kubernetes Service 클러스터 배포

이 작업에서는 Azure 포털을 사용하여 Azure Kubernetes Service 클러스터를 배포합니다.

1. Azure 포털의 검색창에서 "Kubernetes"를 검색한 후 [Kubernetes 서비스]를 클릭합니다.



2. [Kubernetes 서비스] 블레이드의 메뉴에서 [만들기 - Kubernetes 클러스터 만들기]를 클릭합니다.



3. [Kubernetes 클러스터 만들기] 블레이드의 [기본 사항] 탭에서 아래와 같이 구성하고 [다음]을 클릭합니다.

클러스터 이름을 포함한 모든 설정은 CloudSlice의 Azure Policy로 지정되어 있기 때문에 반드시 아래 내용으로 설정을 구성해야 합니다.

- [프로젝트 정보 - 리소스 그룹]: "새로 만들기"를 클릭한 후 "az104-09c-rg1" 이름을 입력합니다.
- [클러스터 세부 정보 - 클러스터 사전 설정 구성]: 개발/테스트
- [클러스터 세부 정보 - Kubernetes 클러스터 이름]: az104-9c-aks1
- [클러스터 세부 정보 - 지역]: (US) East US
- [클러스터 세부 정보 - 가용성 영역]: 없음
- [클러스터 세부 정보 - AKS 가격 책정 계층]: 무료
- [클러스터 세부 정보 - Kubernetes 버전]: 기본값 사용
- [클러스터 세부 정보 - 자동 업그레이드]: 패치와 함께 활성화됨
- [주 노드 풀 - 노드 크기]: 기본값을 사용합니다.
- [주 노드 풀 - 크기 조정 방법]: 수동
- [주 노드 풀 - 노드 개수]: 1



### Kubernetes 클러스터 만들기

기본 사항 노드 풀 액세스 네트워킹 통합 고급 태그 검토 + 만들기

AKS(Azure Kubernetes Service)는 호스트된 Kubernetes 환경을 관리하여 컨테이너 오케스트레이션 전문 기술을 사용하지 않고도 컨테이너화된 애플리케이션을 쉽고 빠르게 배포하고 관리할 수 있도록 합니다. 또한 애플리케이션을 오프라인으로 전환하지 않고도 요청 시 리소스를 프로비전, 업그레이드 및 확장할 수 있어 지속적인 작업 및 유지 관리에 대한 부담이 없어집니다. [자세한 정보](#)

**프로젝트 정보**  
 배포된 리소스와 비용을 관리할 구독을 선택합니다. 폴더 같은 리소스 그룹을 사용하여 모든 리소스를 정리 및 관리합니다.

구독 \* ① MOC Subscription--lod48077261

리소스 그룹 \* ① (신규) az104-09c-rg1 [새로 만들기](#)

**클러스터 세부 정보**  
 클러스터 사전 설정 구성 개발/테스트  
 Kubernetes 클러스터를 빠르게 사용자 지정하려면 위의 사전 설정 구성 중 하나를 선택하세요. 이러한 구성은 언제든지 수정할 수 있습니다. [사전 설정에 대해 자세히 알아보고 비교](#)

Kubernetes 클러스터 이름 \* ① az104-9c-aks1 ✓

지역 \* ① (US) East US

가용성 영역 ① 없음

AKS 가격 책정 계층 ① 무료

Kubernetes 버전 \* ① 1.25.6(기본값)

자동 업그레이드 ① 패치와 함께 활성화됨(권장)

**주 노드 풀**  
 클러스터의 주 노드 풀에 있는 노드의 수와 크기입니다. 프로덕션 워크로드의 경우 복원력을 위해 3개 이상의 노드가 권장됩니다. 개발 또는 테스트 워크로드의 경우 하나의 노드만 필요합니다. 노드 풀을 추가하거나 이 노드 풀의 추가 구성 옵션을 보려면 위의 '노드 풀' 탭으로 이동하세요. 클러스터를 만든 후 다른 노드 풀을 추가할 수 있습니다. [Azure Kubernetes Service의 노드 풀에 대한 자세한 정보](#)

노드 크기 \* ① Standard B4ms  
 4 vcpu, 16 GiB 메모리  
 개발/테스트 구성에는 표준 B4ms를 사용하는 것이 좋습니다. [크기 변경](#)

크기 조정 방법 \* ① ☒ 수동  
☐ 자동 크기 조정  
 개발/테스트 구성에는 자동 스케일링이 권장됩니다.

노드 개수 \* ① 1

4. [노드 풀] 탭에서 "가상 노드 사용" 옵션을 선택하지 않고 [다음]을 클릭합니다.

### Kubernetes 클러스터 만들기

기본 사항 노드 풀 액세스 네트워킹 통합 고급 태그 검토 + 만들기

**노드 풀**  
 기본 탭에 구성된 필수 주 노드 풀 외에도 선택적 노드 풀을 추가하여 다양한 워크로드를 처리할 수 있습니다. [노드 풀에 대해 자세히 알아보기](#)

+ 노드 풀 추가 - 삭제

이름	모드	OS 유형	노드 개수	노드 크기	가용성 영역	최대 Pod/노드
<input type="checkbox"/> agentpool	시스템	Linux	1	Standard_B4ms	없음	110

**가상 노드 사용**  
 가상 노드는 서비스 Azure Container Instances에서 지원하는 버스트 가능 크기 조정을 허용합니다. [가상 노드에 대한 자세한 정보](#)

가상 노드 사용 ① ☐

**노드 풀 OS 디스크 암호화**  
 기본적으로 AKS의 모든 디스크는 Microsoft 관리형 키를 사용하여 미사용 암호화됩니다. 암호화에 대한 추가 제약이 필요한 경우 Azure Key Vault에서 지원하는 디스크 암호화 집합을 사용하여 고유한 키를 제공할 수 있습니다. 디스크 암호화 집합은 클러스터의 모든 노드 풀에 대해 OS 디스크를 암호화하는 데 사용됩니다. [자세한 정보](#)

암호화 형식 (기본값) 플랫폼 관리형 키로 미사용 데이터 암호화

5. [액세스] 탭에서 아래와 같이 구성한 후 [다음]을 클릭합니다.

- 리소스 ID: 시스템에서 할당한 관리 ID
- 인증 및 권한 부여: Kubernetes RBAC가 있는 로컬 계정

**Kubernetes 클러스터 만들기** ...

기본 사항 노드 풀 **액세스** 네트워킹 통합 고급 태그 검토 + 만들기

리소스 ID ① **시스템에서 할당한 관리 ID**  
기본적으로 Azure는 관리 ID를 사용합니다. 서비스 주체를 사용하려면 CLI를 사용하세요. [자세한 정보](#)

인증 및 권한 부여 ① **Kubernetes RBAC가 있는 로컬 계정**

① 클러스터가 배포되면 Kubernetes CLI를 사용하여 RBAC 구성을 관리합니다. [자세한 정보](#)

6. [네트워킹] 탭에서 아래와 같이 구성하고 [다음]을 클릭합니다.

- 네트워크 구성: kubernetes
- DNS 이름 접두사: 기본값 사용

**Kubernetes 클러스터 만들기** ...

기본 사항 노드 풀 액세스 **네트워킹** 통합 고급 태그 검토 + 만들기

'kubernetes' 또는 'Azure CNI'의 두 가지 네트워킹 옵션 중에서 선택할 수 있습니다.

- **kubernetes** 네트워킹 플러그인은 기본값을 사용하여 클러스터의 새 VNet을 만듭니다.
- **Azure CNI** 네트워킹 플러그인을 통해 클러스터에서 사용자 지정 가능한 주소로 새 VNet 또는 기존 VNet을 사용할 수 있습니다. 애플리케이션 Pod가 VNet에 직접 연결되어 VNet 기능과 네이티브 통합이 허용됩니다.

[Azure Kubernetes Service의 네트워킹에 대한 자세한 정보](#)

네트워크 구성 ① **kubernetes**  
Azure CNI

DNS 이름 접두사 \* ① **az104-9c-aks1-dns**

트래픽 라우팅  
부하 분산 장치 ① Standard

보안  
프라이빗 클러스터 사용 ① ☐

권한 있는 IP 범위 설정 ① ☐

네트워크 정책 ① **없음**  
Calico  
Azure

① Azure 네트워크 정책이 kubernetes 네트워킹과 호환되지 않습니다.

7. [통합] 탭에서 아래와 같이 구성한 후 [다음]을 클릭합니다. 프로덕션 환경에서는 모니터링을 활성화하는 것이 권장됩니다.

- [클라우드용 Microsoft Defender - 기본 플랜을 무료로 사용]: 선택하지 않습니다.
- [Azure Container Registry - 컨테이너 레지스트리]: 없음
- [컨테이너 인사이트 - 컨테이너 로그 사용]: 선택하지 않습니다.
- [관리되는 Prometheus - Prometheus 메트릭 사용]: 선택하지 않습니다.
- [Grafana - Grafana 사용]: 선택하지 않습니다.
- [경고 중 - 권장 경고 규칙 사용]: 선택하지 않습니다.
- [Azure Policy - Azure Policy]: 사용 안 함

**Kubernetes 클러스터 만들기** ...

기본 사항   노드 풀   액세스   네트워킹   **통합**   고급   태그   검토 + 만들기

AKS 클러스터를 추가 서비스와 연결합니다.

**클라우드용 Microsoft Defender**  
클라우드용 Microsoft Defender는 하이브리드 클라우드 워크로드 전반에 걸쳐 통합 보안 관리 및 고급 위협 보호를 제공합니다.  
[자세한 정보](#)

기본 플랜을 무료로 사용 ☐

**Azure Container Registry**  
클러스터를 Azure Container Registry에 연결하여 개인 이미지 레지스트리에서 원활한 배포를 지원합니다.  
[Azure Container Registry에 대한 자세한 정보](#)

컨테이너 레지스트리

**Azure Monitor**  
AKS에 기본적으로 포함된 CPU 및 메모리 메트릭 외에도 컨테이너 인사이트를 사용하도록 설정하여 클러스터의 전체 성능 및 상태에 대한 보다 포괄적인 데이터를 살펴볼 수 있습니다. 청구는 데이터 수집 및 보존 설정을 기준으로 이루어집니다.  
[컨테이너 성능 및 상태 모니터링에 대한 자세한 정보](#)  
[가격에 대한 자세한 정보](#)

**컨테이너 인사이트**  
컨테이너 로그 사용 ☐

**관리되는 Prometheus(미리 보기)**  
Azure에서 모니터링되는 Prometheus 관리 서비스는 컨테이너화된 워크로드를 모니터링하기 위한 확장성 있는고가용성 보안 메트릭 플랫폼을 제공합니다. [자세한 정보](#)

**Prometheus 메트릭 사용(미리 보기)** ☐

**Grafana**  
Azure Monitor 작업 영역에 저장된 관리되는 Prometheus 데이터를 시각화하기 위해 Grafana의 완전 관리형 인스턴스를 선택합니다. [가격에 대한 자세한 정보](#)

**Grafana 사용** ☐

**경고 중**  
권장 경고 규칙 사용 ☐

**Azure Policy**  
Azure Policy를 통해 일관된 중앙 집중식 방식으로 AKS 클러스터의 대규모 적용 및 보호 기능을 적용하세요.  
[AKS용 Azure Policy에 대한 자세한 정보](#)

Azure Policy ☐ 사용 ☒ 사용 안 함

8. [고급] 탭에서 기본 설정을 유지하고 [검토 + 만들기]를 클릭합니다. [검토 + 만들기] 탭에서 [만들기]를 클릭합니다.

**Kubernetes 클러스터 만들기** ...

기본 사항   노드 풀   액세스   네트워킹   통합   **고급**   태그   검토 + 만들기

비밀 저장소 CSI 드라이버 사용 ☐

인프라 리소스 그룹  ☒ [편집](#)

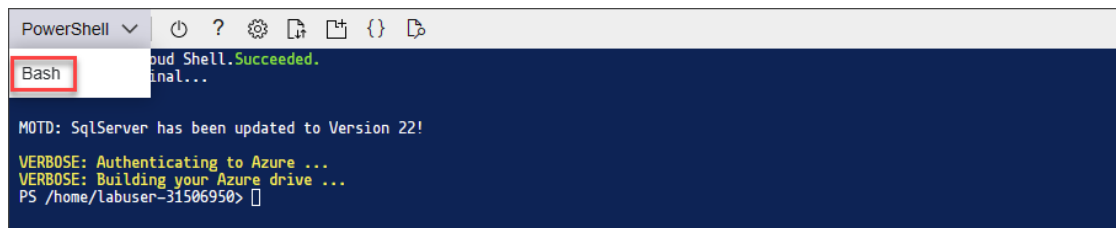
## TASK 04. Azure Kubernetes Service 클러스터에 pod 배포

이 작업에서는 Azure Kubernetes Service 클러스터에 pod를 배포합니다.

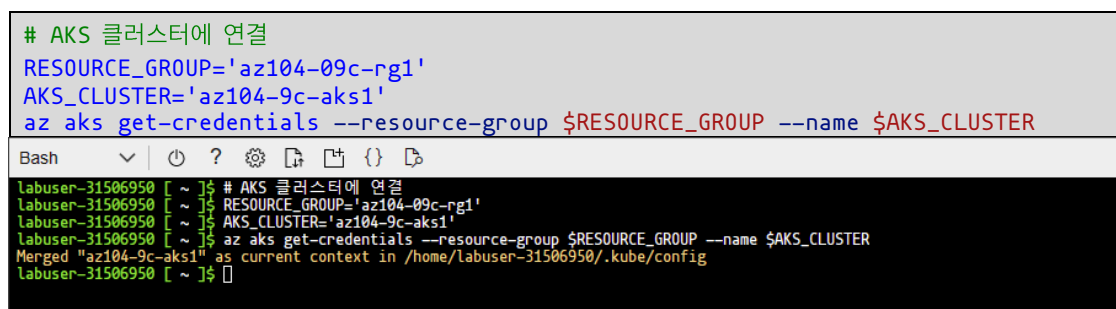
1. [az104-9c-aks1 Kubernetes 서비스] 블레이드로 이동한 후 [설정 - 노드 풀]로 이동합니다. 하나의 노드로 구성된 단일 풀의 클러스터가 표시되는 것을 확인합니다.



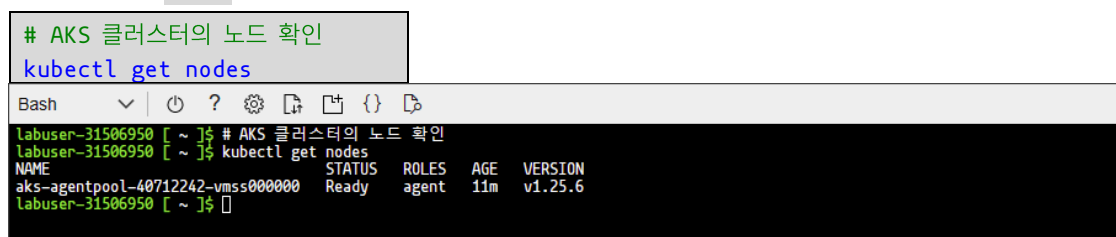
2. Azure 포털에서 [Cloud Shell]을 열고 Bash 세션을 실행합니다.



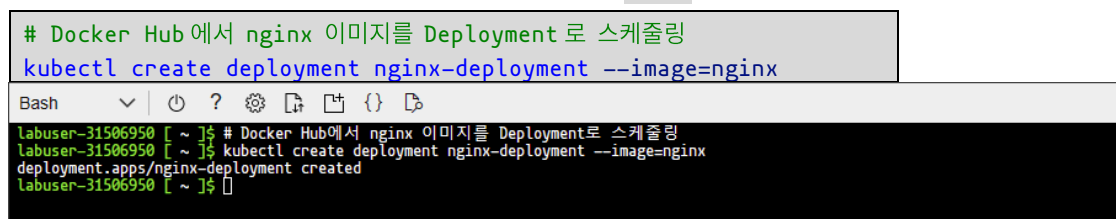
3. [Cloud Shell]의 Bash 세션에서 다음 명령을 실행하여 AKS 클러스터에 액세스하기 위한 자격 증명을 가져옵니다.



4. [Cloud Shell]에서 다음 명령을 실행하여 AKS 클러스터에 배포된 노드를 확인합니다. 출력 결과에서 노드의 상태가 **Ready**로 표시되는지 확인합니다.



5. [Cloud Shell]에서 다음 명령을 실행하여 Docker Hub에서 **nginx** 이미지를 배포합니다.



6. [Cloud Shell]에서 다음 명령을 실행하여 Kubernetes pod가 생성되었는지 확인합니다.



```
kubectl get pods
```

Bash

```
labuser-31506950 [ ~ ]$ # Pod 생성 확인
labuser-31506950 [ ~ ]$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-5fddf85c67-8xj45  1/1     Running   0           41s
labuser-31506950 [ ~ ]$
```

7. [Cloud Shell]에서 다음 명령을 실행하여 **Deployment** 상태를 확인합니다.

```
# Deployment 상태 확인
kubectl get deployment
```

Bash

```
labuser-31506950 [ ~ ]$ # Deployment 상태 확인
labuser-31506950 [ ~ ]$ kubectl get deployment
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment    1/1     1             1           64s
labuser-31506950 [ ~ ]$
```

8. [Cloud Shell]에서 다음 명령을 실행하여 인터넷에서 **pod**에 액세스할 수 있도록 구성합니다.

```
# Pod를 인터넷에서 액세스할 수 있도록 설정
kubectl expose deployment nginx-deployment --port=80 --type=LoadBalancer
```

Bash

```
labuser-31506950 [ ~ ]$ # Pod를 인터넷에서 액세스할 수 있도록 설정
labuser-31506950 [ ~ ]$ kubectl expose deployment nginx-deployment --port=80 --type=LoadBalancer
service/nginx-deployment exposed
labuser-31506950 [ ~ ]$
```

9. [Cloud Shell]에서 다음 명령을 실행하여 공용 IP 주소가 프로비저닝되었는지 확인합니다. **nginx-deployment** 항목의 **EXTERNALIP** 열에 공용 IP 주소가 표시되는지 확인합니다. 공용 IP 주소가 표시되지 않으면 조금 더 기다린 후 다시 명령을 실행하여 공용 IP 주소가 표시되는지 확인합니다. 표시되는 공용 IP 주소를 복사합니다.

```
# Service 개체 확인
kubectl get service
```

Bash

```
labuser-31506950 [ ~ ]$ # Service 개체 확인
labuser-31506950 [ ~ ]$ kubectl get service
NAME                TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
kubernetes          ClusterIP   10.0.0.1     <none>        443/TCP          15m
nginx-deployment    LoadBalancer 10.0.190.0   52.226.206.153 80:31838/TCP     27s
```

10. 브라우저에서 새 탭을 열고 위에서 복사한 공용 IP 주소에 액세스합니다. 아래와 같이 "Welcome to nginx!" 페이지가 표시되는 것을 확인합니다.



## TASK 05. Azure Kubernetes Service 클러스터에서 컨테이너화된 워크로드 확장

이 작업에서는 pod의 수를 수평적으로 증가시킨(horizontally scale) 다음 클러스터 노드의 수를 증가시킵니다.

1. [Cloud Shell]을 열고 다음 명령을 실행하여 pod의 수를 2개로 증가시켜 deployment를 확장합니다.

```
# 실행 중인 Pod 수 증가
kubectl scale --replicas=2 deployment/nginx-deployment
```

```
Bash
labuser-31506950 [ ~ ]$ # 실행 중인 Pod 수 증가
labuser-31506950 [ ~ ]$ kubectl scale --replicas=2 deployment/nginx-deployment
deployment.apps/nginx-deployment scaled
labuser-31506950 [ ~ ]$
```

2. [Cloud Shell]에서 다음 명령을 실행하여 deployment가 확장의 결과를 확인합니다. NGINX를 실행하는 pod수가 2개로 증가한 것을 확인합니다.

```
# Pod 확인
kubectl get pods
```

```
Bash
labuser-31506950 [ ~ ]$ # Pod 확인
labuser-31506950 [ ~ ]$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-5fbd85c67-8xj45    1/1     Running   0           4m2s
nginx-deployment-5fbd85c67-jfwxx    1/1     Running   0           23s
labuser-31506950 [ ~ ]$
```

3. [Cloud Shell]에서 다음 명령을 실행하여 노드의 수를 2대로 증가시켜 클러스터를 scale out 합니다. 추가 노드가 프로비저닝될 때까지 기다립니다. 작업은 대략 3분 정도가 소요됩니다.

```
# AKS의 노드 수 증가
RESOURCE_GROUP='az104-09c-rg1'
AKS_CLUSTER='az104-9c-aks1'
az aks scale --resource-group $RESOURCE_GROUP --name $AKS_CLUSTER --node-count 2
```

```
Bash
labuser-31506950 [ ~ ]$ # AKS의 노드 수 증가
labuser-31506950 [ ~ ]$ RESOURCE_GROUP='az104-09c-rg1'
labuser-31506950 [ ~ ]$ AKS_CLUSTER='az104-9c-aks1'
labuser-31506950 [ ~ ]$ az aks scale --resource-group $RESOURCE_GROUP --name $AKS_CLUSTER --node-count 2
{
  "aadProfile": null,
  "addonProfiles": {
    "azureKeyvaultSecretsProvider": {
      "config": null,
      "enabled": false,
      "identity": null
    },
    "azurepolicy": {
      "config": null,
      "enabled": false,
      "identity": null
    }
  }
}
```

4. [Cloud Shell]에서 다음 명령을 실행하여 클러스터 확장의 결과를 확인합니다. 노드의 수가 2대로 증가된 것을 확인할 수 있습니다.

```
# 클러스터 노드 확인
kubectl get nodes
```

```
Bash
labuser-31506950 [ ~ ]$ # 클러스터 노드 확인
labuser-31506950 [ ~ ]$ kubectl get nodes
NAME                                STATUS    ROLES    AGE   VERSION
aks-agentpool-40712242-vmss000000  Ready    agent    21m   v1.25.6
aks-agentpool-40712242-vmss000001  Ready    agent    95s   v1.25.6
labuser-31506950 [ ~ ]$
```

5. [Cloud Shell]에서 다음 명령을 실행하여 deployment를 다시 확장합니다.

```
# Deployment의 Pod 수를 10개로 증가
```

```
kubectl scale --replicas=10 deployment/nginx-deployment
```

Bash

```
labuser-31506950 [ ~ ]$ # Deployment의 Pod 수를 10개로 증가
labuser-31506950 [ ~ ]$ kubectl scale --replicas=10 deployment/nginx-deployment
deployment.apps/nginx-deployment scaled
labuser-31506950 [ ~ ]$
```

6. [Cloud Shell]에서 다음 명령을 실행하여 **deployment** 확장 결과를 확인합니다. **pod**의 수가 10개로 증가된 것을 확인할 수 있습니다.

```
# 확장된 Pod 수 확인
kubectl get pods
```

Bash

```
labuser-31506950 [ ~ ]$ # 확장된 Pod 수 확인
labuser-31506950 [ ~ ]$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-5fddf85c67-8rqhn   1/1     Running   0           25s
nginx-deployment-5fddf85c67-8xj45   1/1     Running   0           9m34s
nginx-deployment-5fddf85c67-97q6j   1/1     Running   0           25s
nginx-deployment-5fddf85c67-bknhd   1/1     Running   0           25s
nginx-deployment-5fddf85c67-d58rn   1/1     Running   0           25s
nginx-deployment-5fddf85c67-hzwkk   1/1     Running   0           25s
nginx-deployment-5fddf85c67-jfwxx   1/1     Running   0           5m55s
nginx-deployment-5fddf85c67-lfwlm   1/1     Running   0           25s
nginx-deployment-5fddf85c67-mc9kl   1/1     Running   0           25s
nginx-deployment-5fddf85c67-qdbk2   1/1     Running   0           25s
labuser-31506950 [ ~ ]$
```

7. [Cloud Shell]에서 다음 명령을 실행하여 클러스터 노드간 **pod**가 어떻게 배포되어 있는지 확인합니다. **pod**가 노드간에 분산되어 있는 것을 확인할 수 있습니다.

```
# 클러스터 노드에 Pod가 분산되었는지 확인
kubectl get pod -o=custom-columns=NODE:.spec.nodeName,POD:.metadata.name
```

Bash

```
labuser-31506950 [ ~ ]$ # 클러스터 노드에 Pod가 분산되었는지 확인
labuser-31506950 [ ~ ]$ kubectl get pod -o=custom-columns=NODE:.spec.nodeName,POD:.metadata.name
NODE                                POD
aks-agentpool-40712242-vmss000001  nginx-deployment-5fddf85c67-8rqhn
aks-agentpool-40712242-vmss000000  nginx-deployment-5fddf85c67-8xj45
aks-agentpool-40712242-vmss000001  nginx-deployment-5fddf85c67-97q6j
aks-agentpool-40712242-vmss000000  nginx-deployment-5fddf85c67-bknhd
aks-agentpool-40712242-vmss000001  nginx-deployment-5fddf85c67-d58rn
aks-agentpool-40712242-vmss000000  nginx-deployment-5fddf85c67-hzwkk
aks-agentpool-40712242-vmss000001  nginx-deployment-5fddf85c67-jfwxx
aks-agentpool-40712242-vmss000000  nginx-deployment-5fddf85c67-lfwlm
aks-agentpool-40712242-vmss000001  nginx-deployment-5fddf85c67-mc9kl
aks-agentpool-40712242-vmss000000  nginx-deployment-5fddf85c67-qdbk2
labuser-31506950 [ ~ ]$
```

8. [Cloud Shell]에서 다음 명령을 실행하여 배포를 삭제합니다.

```
# Deployment 삭제
kubectl delete deployment nginx-deployment
```

Bash

```
labuser-31506950 [ ~ ]$ # Deployment 삭제
labuser-31506950 [ ~ ]$ kubectl delete deployment nginx-deployment
deployment.apps "nginx-deployment" deleted
labuser-31506950 [ ~ ]$
```

## TASK 06. 리소스 정리

1. [Cloud Shell]의 Bash 세션에서 다음 명령을 실행하여 이 실습에서 만든 리소스 그룹을 확인합니다.

```
# 실습에서 배포한 리소스 그룹 확인
az group list --query "[?starts_with(name,'az104-09c')].name" --output tsv
```

```
Bash
labuser-31506950 [ ~ ]$ # 실습에서 배포한 리소스 그룹 확인
labuser-31506950 [ ~ ]$ az group list --query "[?starts_with(name,'az104-09c')].name" --output tsv
az104-09c-rg1
labuser-31506950 [ ~ ]$
```

2. [Cloud Shell]에서 다음 명령을 실행하여 실습에서 만들었던 모든 리소스 그룹을 삭제합니다. 이 명령은 비동기식으로 실행되기 때문에 동일한 Bash 세션에서 다른 Azure CLI 명령을 실행할 수 있습니다. 하지만 리소스 그룹이 실제 삭제될 때까지는 몇 분 정도가 소요됩니다.

```
# 실습에서 사용한 리소스 그룹 삭제
az group list --query "[?starts_with(name,'az104-09c')].name" --output tsv | \
xargs -L1 bash -c 'az group delete --name $0 --no-wait --yes'
```

```
Bash
labuser-31506950 [ ~ ]$ # 실습에서 사용한 리소스 그룹 삭제
labuser-31506950 [ ~ ]$ az group list --query "[?starts_with(name,'az104-09c')].name" --output tsv | \
xargs -L1 bash -c 'az group delete --name $0 --no-wait --yes'
labuser-31506950 [ ~ ]$
```