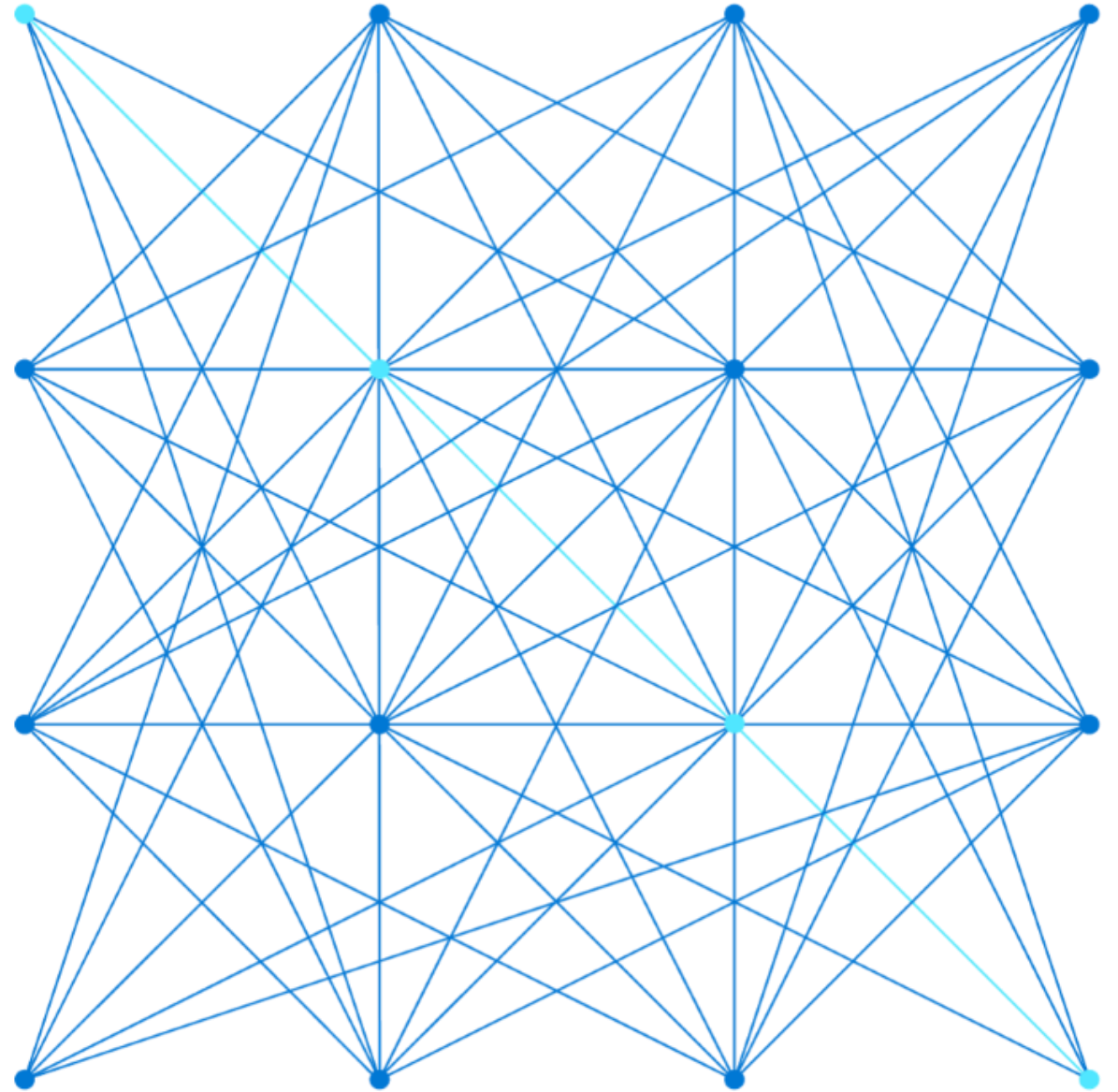


AZ-400.00

Module 16: Creating and Managing Kubernetes Service Infrastructure



(#mw) Create an AKS Resource now

Lesson 01: Module overview



Module overview



Lesson 1: Module overview



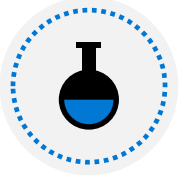
Lesson 2: Azure Kubernetes Service (AKS)



Lesson 3: Kubernetes tooling



Lesson 4: Integrating AKS with pipelines



Lesson 5: Lab



Lesson 6: Module review and takeaways

Learning objectives

After completing this module, students will be able to:



Deploy and configure Managed Kubernetes Cluster

Lesson 02: Azure Kubernetes Service



Kubernetes overview

Container orchestration technology (originated from Google)

Open-source system for automating deployment, scaling, and management of containerized applications

Kubernetes can be thought of as:

- A container platform

- A microservices platform

- A portable cloud platform (and more)

Several other container orchestration technologies are available, including:

- Docker Swarm

- Mesosphere DC/OS

- AWS ECS

- Azure Service Fabric



Azure Kubernetes Service

AKS is Microsoft's implementation of Kubernetes:

Reduces the complexity of managing a Kubernetes Cluster by offloading it to Azure

Easier to deploy and manage container applications without container orchestration expertise

AKS manages the following aspects of a Kubernetes cluster:

Manages health monitoring and maintenance

Performs simple cluster scaling

Enables master nodes to be fully managed by Microsoft

You're responsible only for managing the agent nodes

Master nodes are free, and you pay only for running agent nodes



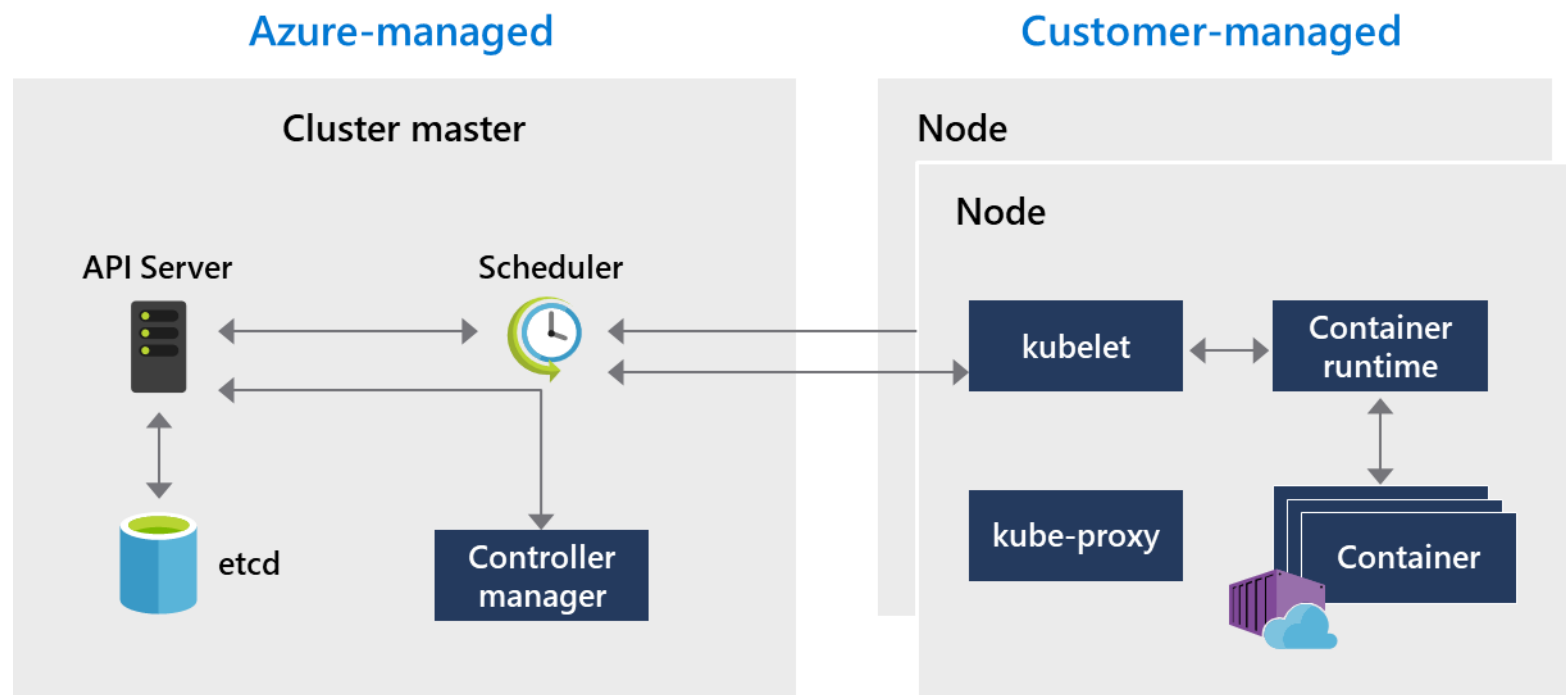
az-p-aks

AKS architectural components

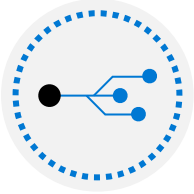
Kubernetes cluster is divided into two components:

Cluster master nodes – provide core Kubernetes services and orchestration of application workloads

Nodes that run your application workloads



Kubernetes networking



Kubernetes uses Services to logically group a set of pods together to provide network connectivity



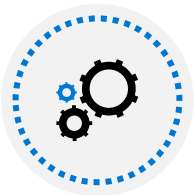
ClusterIP:

Creates an internal IP address for use **within** the AKS cluster



NodePort:

Creates a **port mapping on the underlying node**, so an application can be accessed directly with the node IP address and port



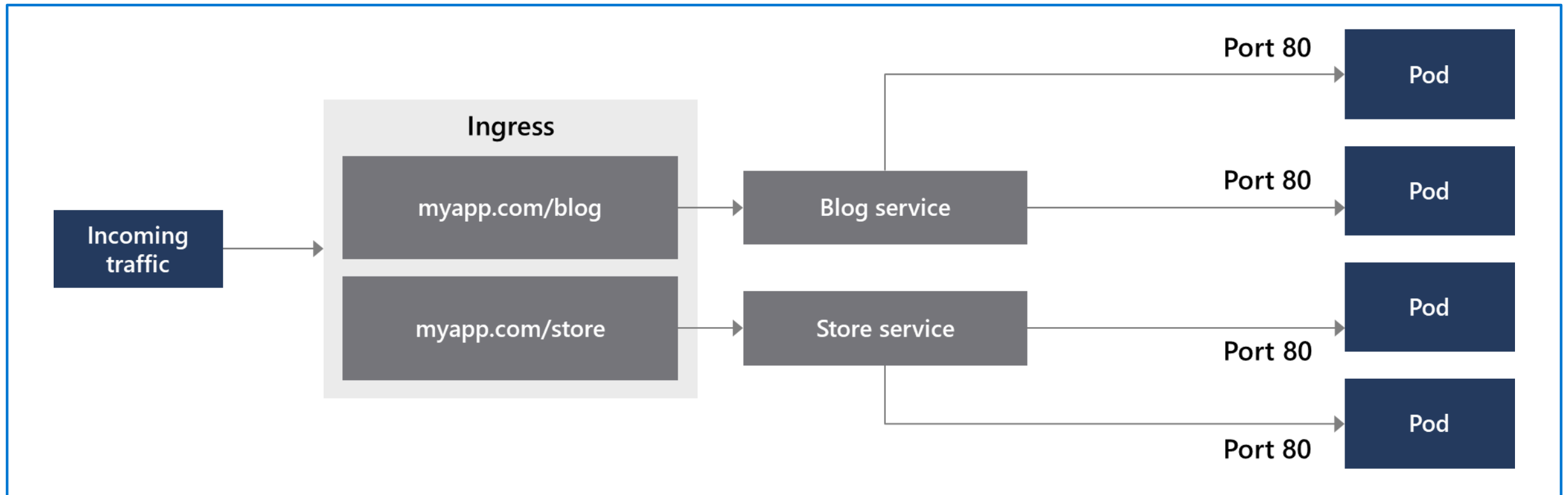
LoadBalancer:

Creates an Azure LB resource, **configures an external IP address, and connects the pods to the load balancer backend pool**

Customer traffic allowed through load balancing rules on the desired ports

Ingress controllers

Ingress controllers work at Layer 7 so can use more intelligent rules to distribute traffic
For example, route HTTP traffic to different applications based on the inbound URL



Deployment units



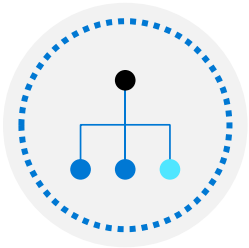
Pod– deployment unit representing a running process on the cluster:

Used by Kubernetes to package applications

Consists of one or more containers, Configuration, storage resources, and networking support

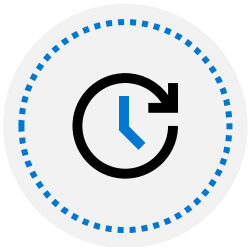
Can be grouped into services to create microservices

Described using YAML or JSON



Deploy an application to Kubernetes using the kubectl CLI:

Deploy and manage Kubernetes pods from Azure DevOps



Continuous delivery:

Build-and-release pipelines are run for every check-in on the Source repository

Demonstration: Deploying and connecting to an AKS cluster

Part 1: Deploy an AKS cluster using the Azure CLI

- 1 Open Azure Cloud Shell and choose the **Bash** environment
- 2 Create an Azure resource group and an AKS cluster
- 3 Using `az aks get-credentials`, connect to the Kubernetes cluster, and verify the connection status
- 4 Create a YAML file called `azure-vote.yaml`
- 5 Deploy the application using the `kubectl` command. It should result in the below output:

```
deployment "azure-vote-back" created  
service "azure-vote-back" created  
deployment "azure-vote-front" created  
service "azure-vote-front" created
```

Demonstration: Deploying and connecting to an AKS cluster

Part 2: Monitor the health of cluster and pods running in the application

- 6 When the application runs, a Kubernetes service exposes the application front end to the internet
- 7 Initially the **EXTERNAL-IP** for the azure-vote-front service is shown as pending
- 8 Eventually changes from pending to an actual public IP address
- 9 To view the application, open a web browser to the external IP address of your service
- 10 Monitor health and logs under Monitoring in the Azure portal
- 11 Examine container logs for the azure-vote-front pod
- 12 Delete resources when finished to avoid incurring charges

Continuous deployment

In Kubernetes, you can update a service with a rolling update

Existing traffic to a container is first drained, and the container replaced

Syntax for running a rolling update

```
kubectl apply -f  
nameofyamlfile
```

The **apply** command works for an initial deployment or an update to an existing deployment

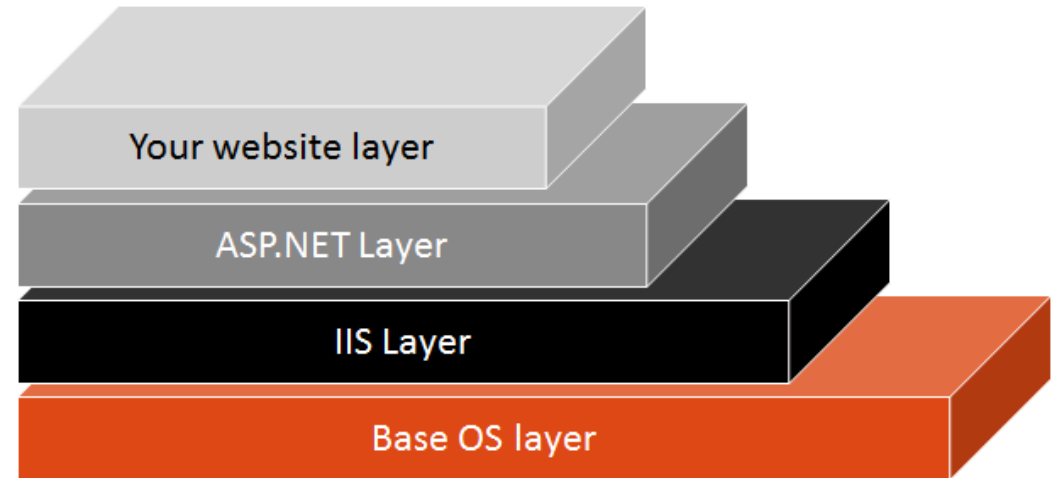
Kubernetes applies a rolling update when the name of the image for a service changes in the YAML file

The cluster will take care of updating the images without downtime (on the assumption the application container is built as stateless)

Updating images

Ensure you update images for your container applications regularly

- A large part of a container image is the base OS layer
- Updating an image's base OS layer is usually a matter of getting the latest update layer
- Create a new image for every change you make in your own code, and ensure that all layers receive regular patching
- To patch an image's OS layer, change the image version number (if using a Docker file to create the image)



Lesson 03: Kubernetes tooling



kubectl

Command-line tool for running commands against Kubernetes clusters. Deploy applications, manage cluster resources.

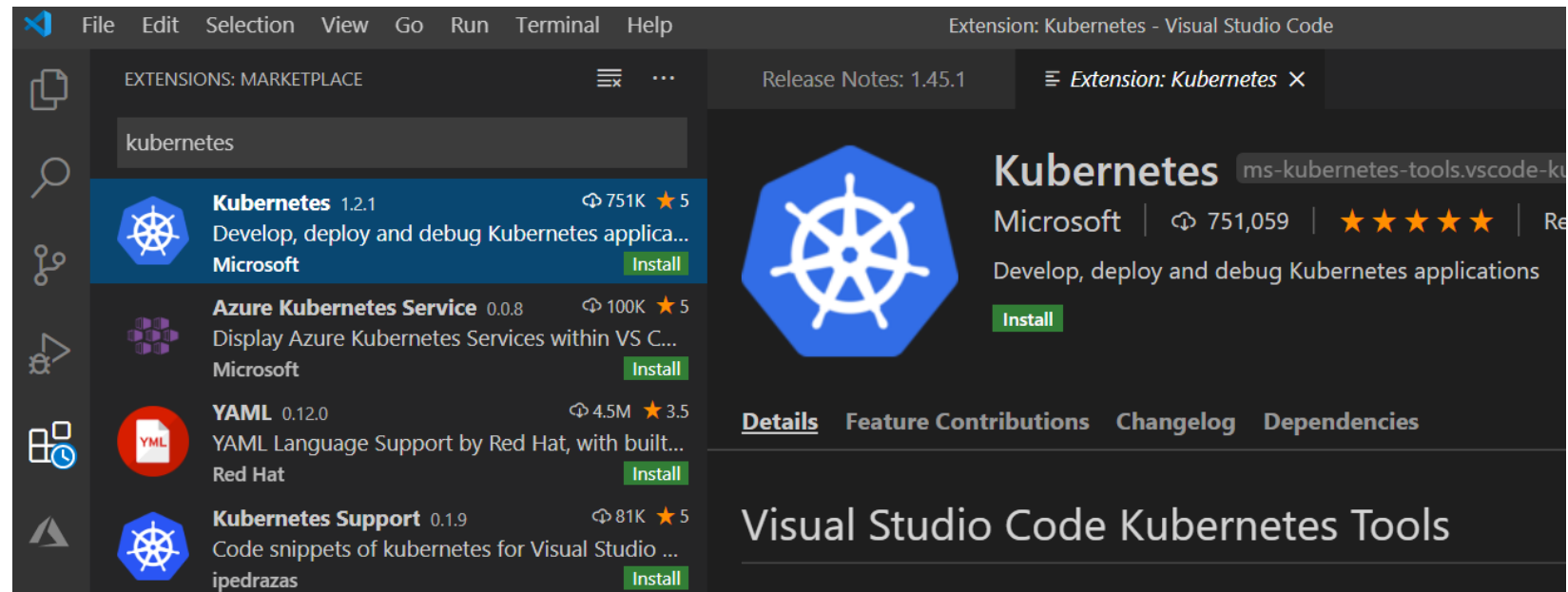
Common Commands	
annotate	Add/update annotations for resources
apply	Apply configuration changes
autoscale	Scale pods managed by a replication controller
certificate	Modify certificate resources
cluster-info	Display endpoint information about master and services
config	Modify kubeconfig files
cp	Copies files to/from containers
describe	Show detailed state about resources
exec	Execute a command in a container
label	Add/update labels for resources
logs	Print the logs for a container
run	Run an image on a cluster

Helm

- 1 [Package manager for Kubernetes](#)
- 2 Makes it easier to package, configure, and deploy applications and services
- 3 Command line tool: helm (user interface for Helm functionality)
- 4 Server component: tiller (ran on k8s cluster and accepted commands from helm – but removed from Helm 3 and later)
- 5 Helm packages are called charts (implemented as YAML)

Kubernetes extension for Visual Studio Code

- 1 Requires existing installation of Docker and kubectl
- 2 Work with local minikube cluster or AKS cluster
- 3 Debug containers iteratively
- 4 Deploy to Azure
- 5 Helps develop k8s manifests and Helm charts



Lesson 04: Integrating AKS with pipelines



Integrating AKS with pipelines

- 1 Pipelines can deploy containers from images
- 2 AKS cluster needs permission to retrieve images from the container registry
- 3 Task: Create deployments and services in AKS (#mw the lab has this)
- 4 Task: Update image in AKS (#mw the lab has this)

Kubernetes and Azure Key Vault

Use Kubernetes and Azure Key Vault together to get the best security benefits:

Azure Key Vault Secret store

Kubernetes ConfigMaps

Kubernetes Secrets

Readiness and liveness probes

Check container health

Detect unresponsive containers (e.g. deadlocks)

Wait for containers to start up

ports:

- name: liveness-port
containerPort: 8080
hostPort: 8080

livenessProbe:

httpGet:

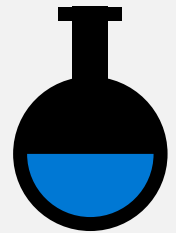
path: /health-check
port: liveness-port
failureThreshold: 2
periodSeconds: 15

startupProbe:

httpGet:

path: /health-check
port: liveness-port
failureThreshold: 40
periodSeconds: 15

Lesson 05: Lab



Deploying a multi-container application to Azure Kubernetes Services



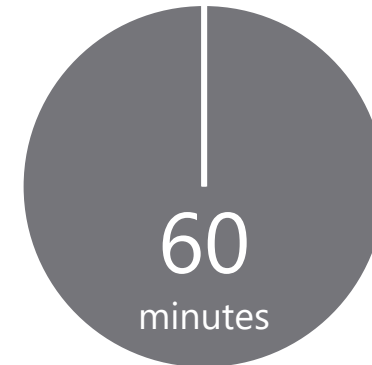
Lab overview:

In this lab, you will use Azure DevOps to deploy a containerized ASP.NET Core web application MyHealthClinic (MHC) to an AKS cluster.

Objectives:

- Create an Azure DevOps team project with a .NET Core application using the Azure DevOps Demo Generator tool.
- Use Azure CLI to create an Azure Container registry (ACR), an AKS cluster and an Azure SQL database
- Configure containerized application and database deployment by using Azure DevOps
- Use Azure DevOps pipelines to build to automatically deploy containerized applications

Duration:



Lesson 06: Module review and takeaways



What did you learn?



Deploy and configure Managed Kubernetes Cluster

Module review questions

1

True or False: Azure Policy natively integrates with AKS, allowing you to enforce rules across multiple AKS clusters. Track, validate and configure nodes, pods and container images for compliance

2

Kubernetes CLI is called?

3

For workloads running in AKS Kubernetes Web Dashboard allows you to view _____.

4

Pods can be described using which of the following languages? Select all that apply