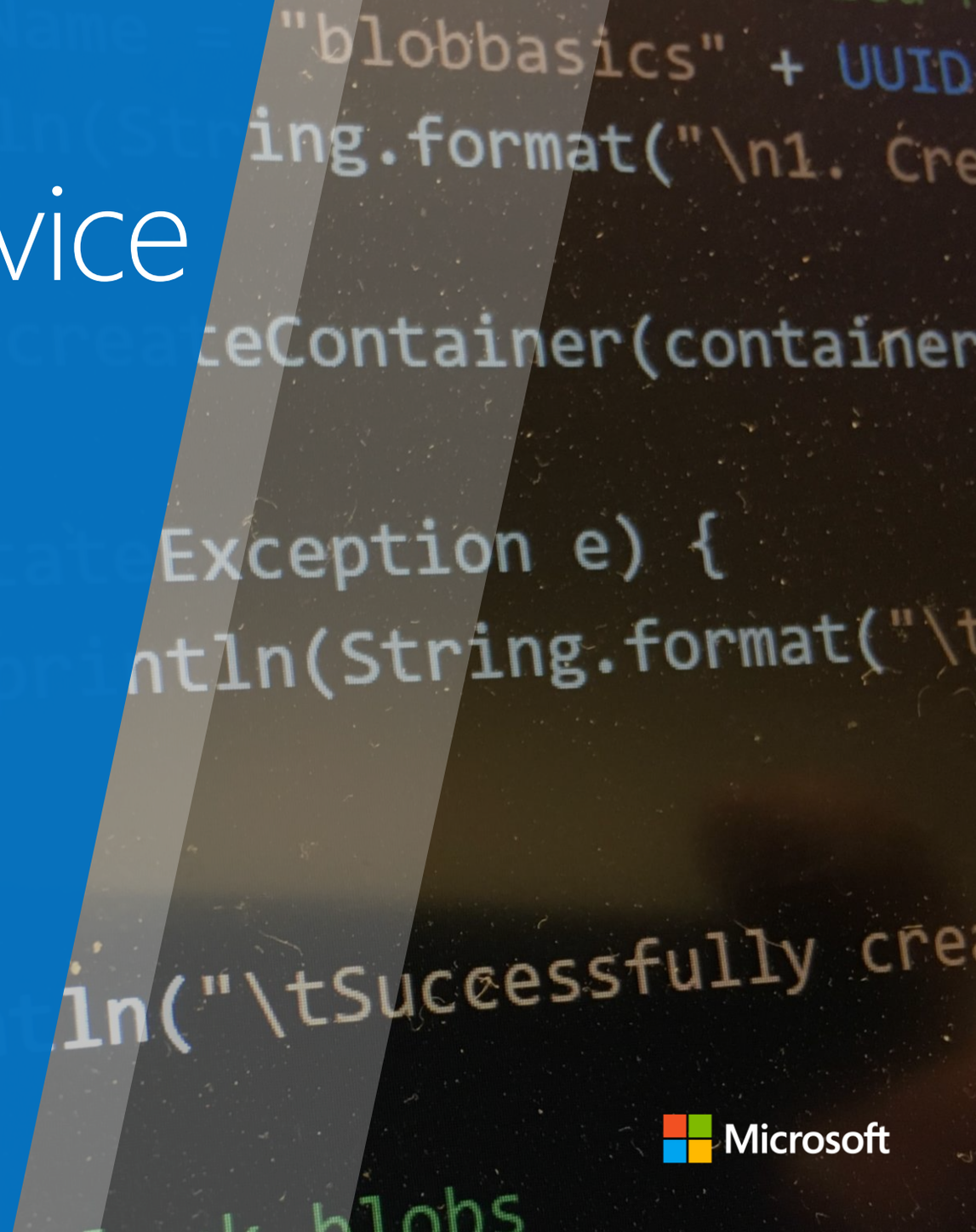
Azure Kubernetes Service in Action

Matjaz Perpar, Senior Cloud Solution Architect
Uros Kastelic, Specialist – Azure App Development



Microsoft

Agenda

10.00 – 10.30	Fundamentals of Kubernetes on Microsoft Azure
10.30 – 10.40	LAB: Getting started with Lab environment
10.40 – 10.50	LAB: Deploying AKS cluster
10.45 – 12.00	LAB: Deploying application to AKS
12.00 – 13.00	Lunch
13.00 – 14.15	LAB: Deploying application to AKS
14.15 - 14.30	Break
14.30 - 16.00	LAB: Azure DevOps and AKS better together - Continuous Integration and Continuous Delivery
16.00 - 16.30	Q&A

AKS Workshop Hands-on lab (2/2)

4. Kubernetes tutorial / guide

(Study material to help you get started to learn Kubernetes)

http://aksworkshop.io/

- Our GitHub site:

https://github.com/matjazperpar/AKS-Workshop-Ukraine

5. Install VS Code: https://code.visualstudio.com/download

6. How to use Cloud Shell inside VS Code

(Cloud Shell in VS Code requires Node.js 6+ on machine)

https://microsoft.github.io/AzureTipsAndTricks/blog/tip49.html

Azure Pass vouchers

- https://www.microsoftazurepass.com/

Containers Momentum

[1]IDC FutureScape, 2018

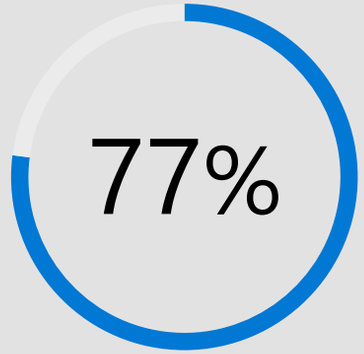
Containers **momentum**

"By 2020, more than **50%** of enterprises will run **mission-critical, containerized cloud-native applications** in production."

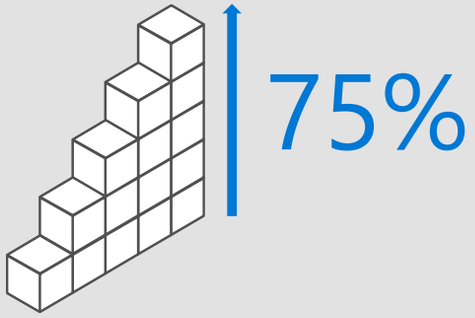
Gartner

Half of container environment is orchestrated.[1]

77% of companies[2] who use container orchestrators choose Kubernetes.

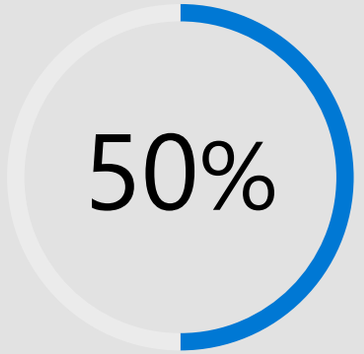


The average size of a container deployment has grown 75% in one year. [1]



Larger companies are leading the adoption.[1]

Nearly **50%** of organizations[1] running 1000 or more hosts have adopted containers.

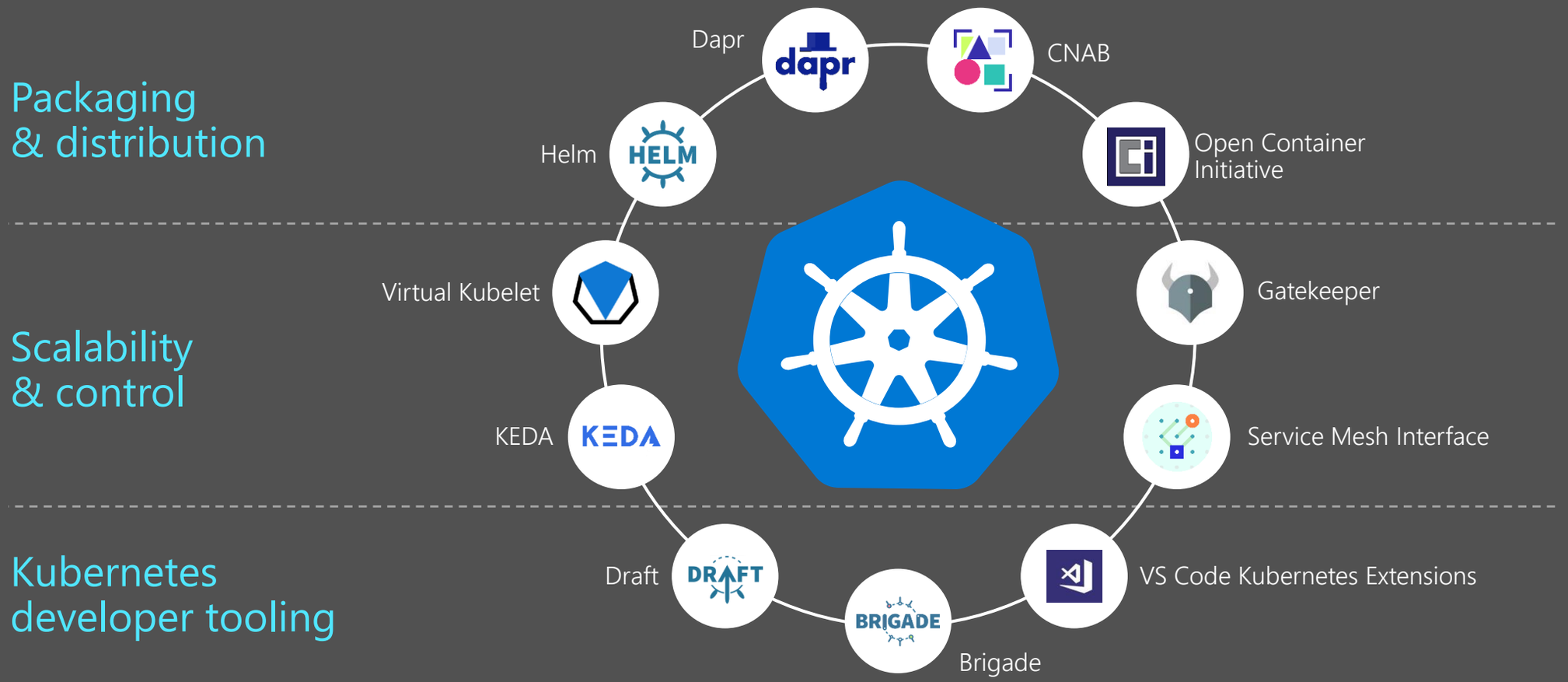


[1] Datadog report: 8 Surprising Facts About Real Docker Adoption
[2] CNCF survey: cloud-native-technologies-scaling-production-applications


By 2022
>75%
of global organizations will be running containerized applications in production[1]

[1]Gartner.

Microsoft's contributions to the community



The benefits of using containers

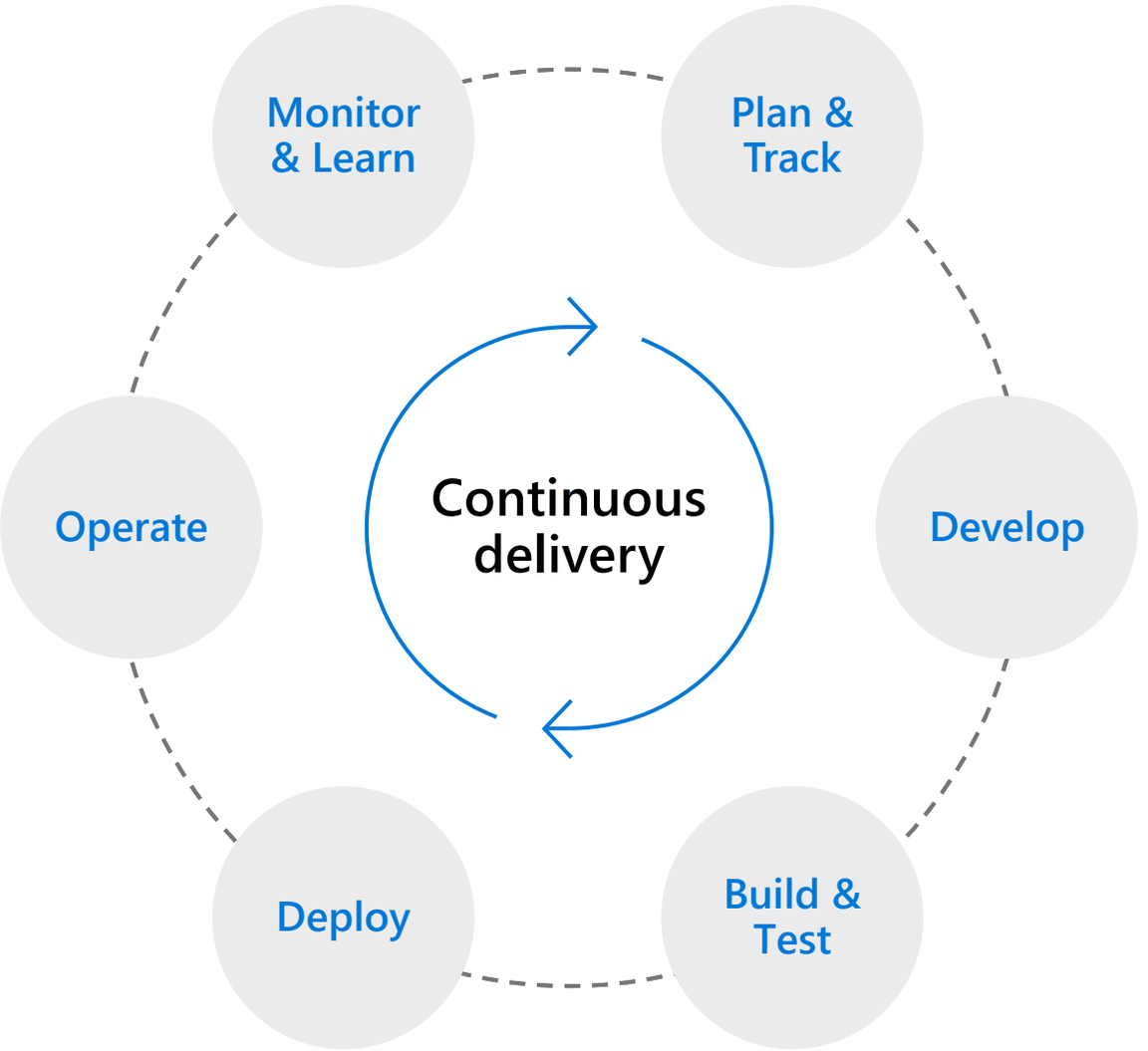
Any OS	Anywhere	Any app	Any language
Linux	On-premises	Monolith	Java
Windows	Cloud	Microservice	.Net
			Python
			Node

DevOps

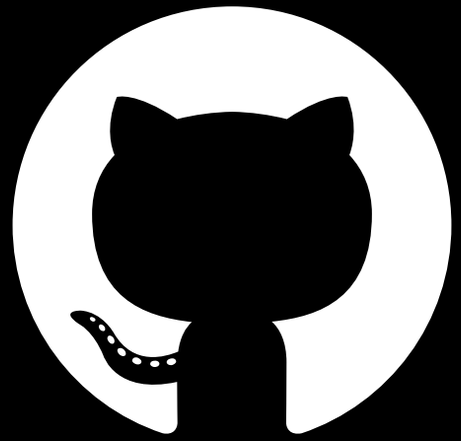
What is DevOps?

People. Process. Products.

"DevOps is the union of people, process, and technology to enable continuous delivery of value to your end users."



Loved by
Developers

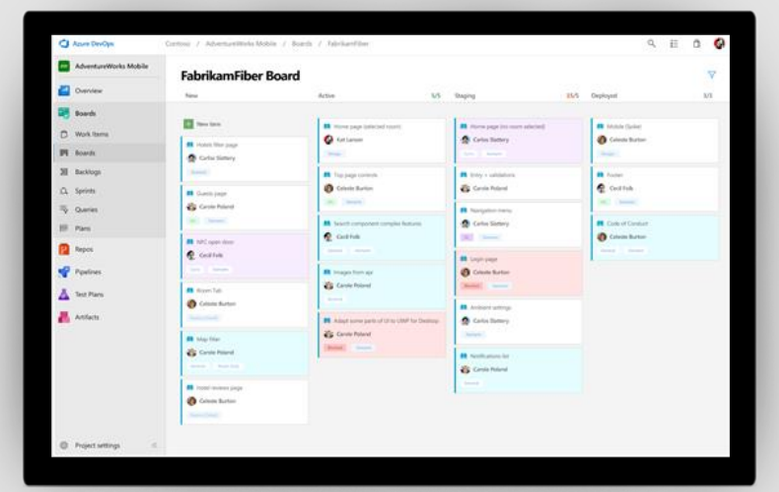


Trusted by
Enterprises

GitHub

Microsoft, trusted by enterprise development teams for decades

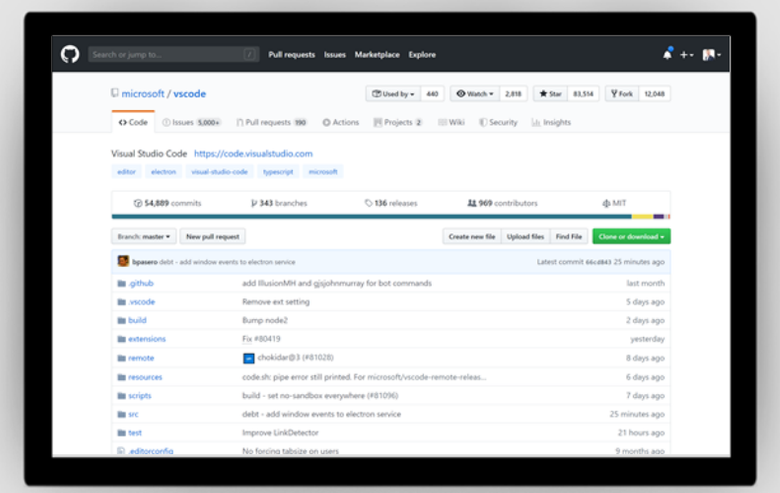
Now committed to the world's largest developer community



Azure DevOps

Empowering 1000s of enterprises

15+ years of developer innovation



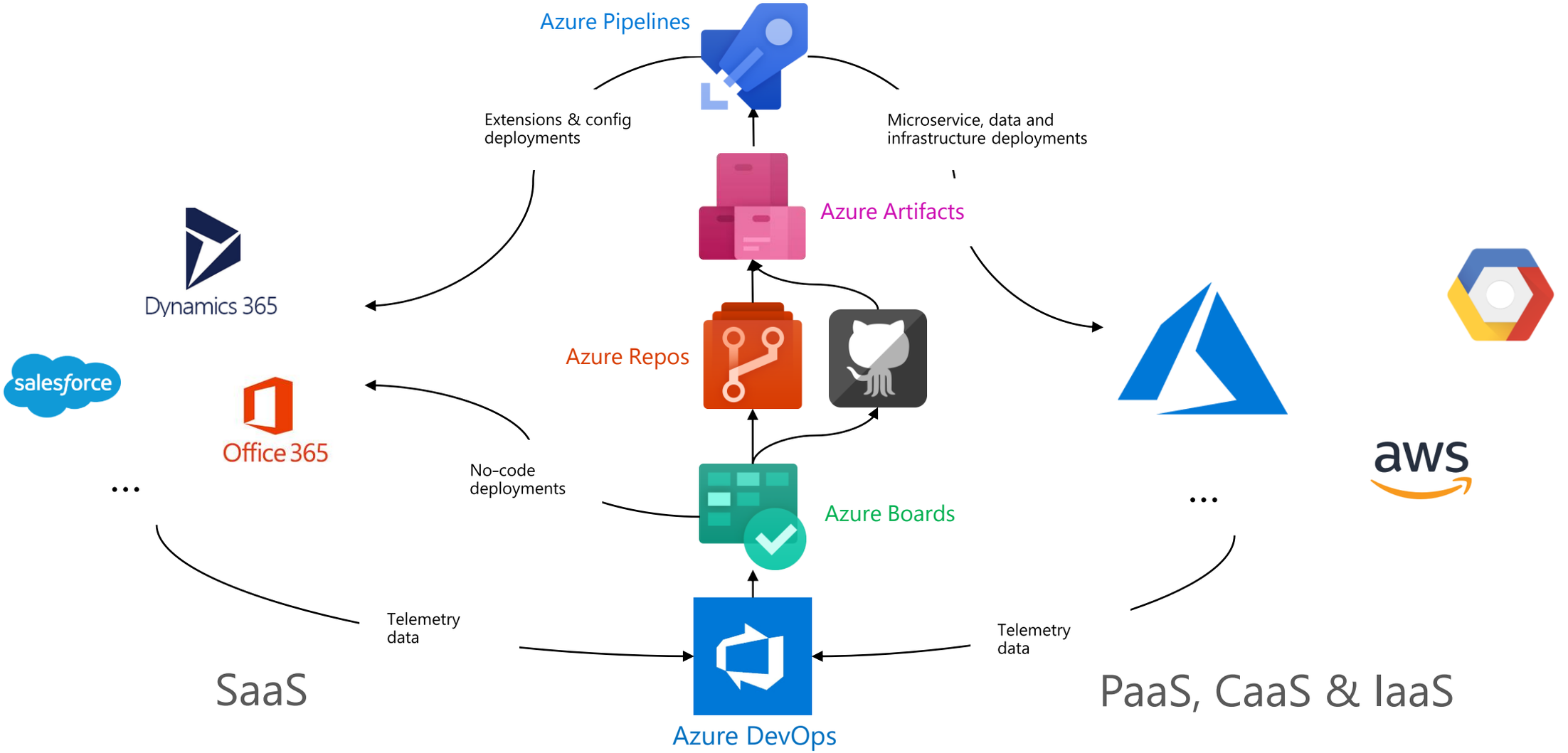
GitHub

The home of open source software

40 million+ users

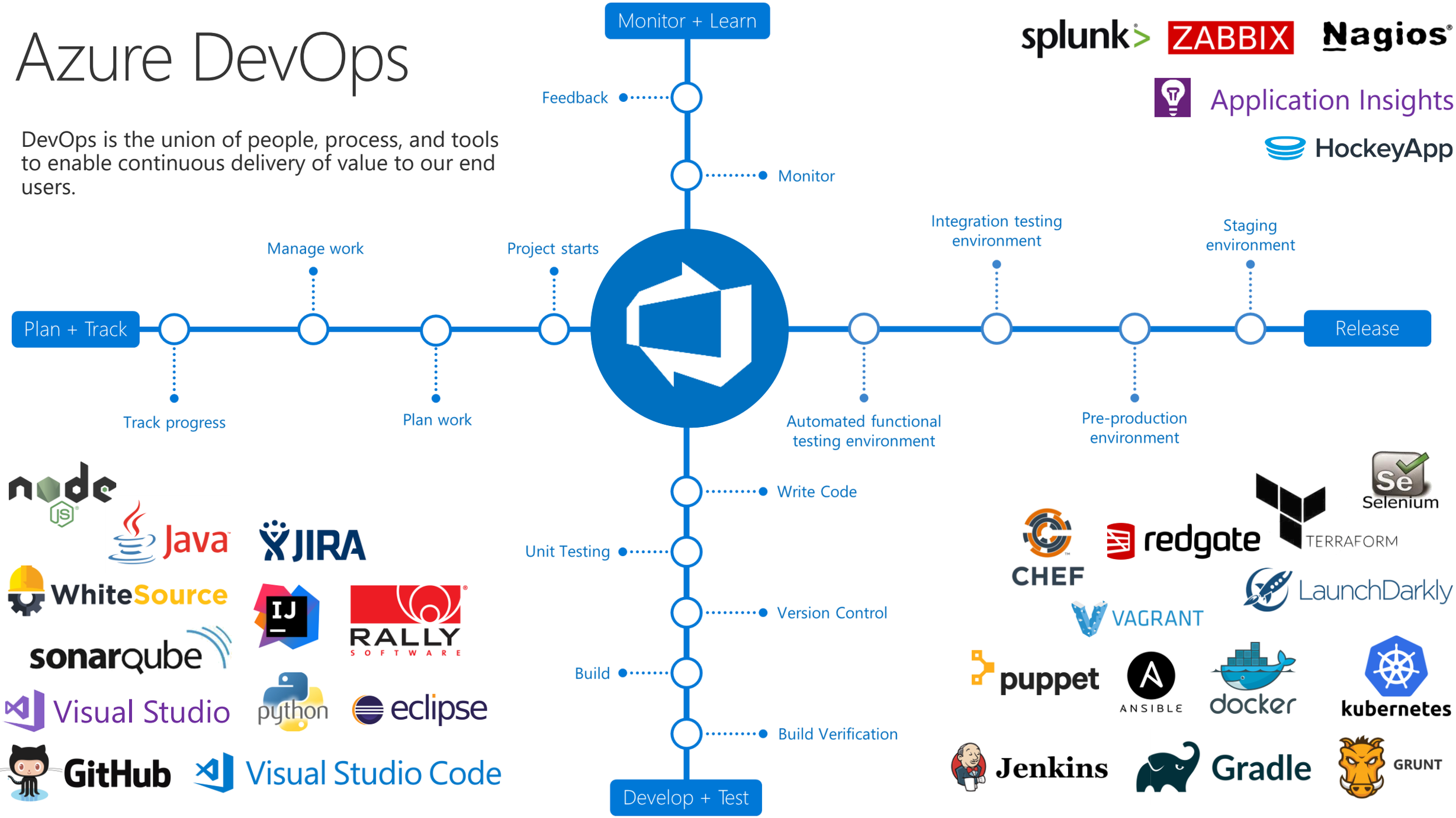
Azure DevOps For All

- **Even if your project or deployment is no-code (like an Office or other SaaS deployment), DevOps practices and tools can help improve the velocity, functionality and feedback loop for users**



Azure DevOps

DevOps is the union of people, process, and tools to enable continuous delivery of value to our end users.



splunk> ZABBIX Nagios

Application Insights

HockeyApp

node JS Java JIRA

WhiteSource IJ RALLY SOFTWARE

sonarqube python eclipse

Visual Studio

GitHub Visual Studio Code

Selenium

redgate TERRAFORM

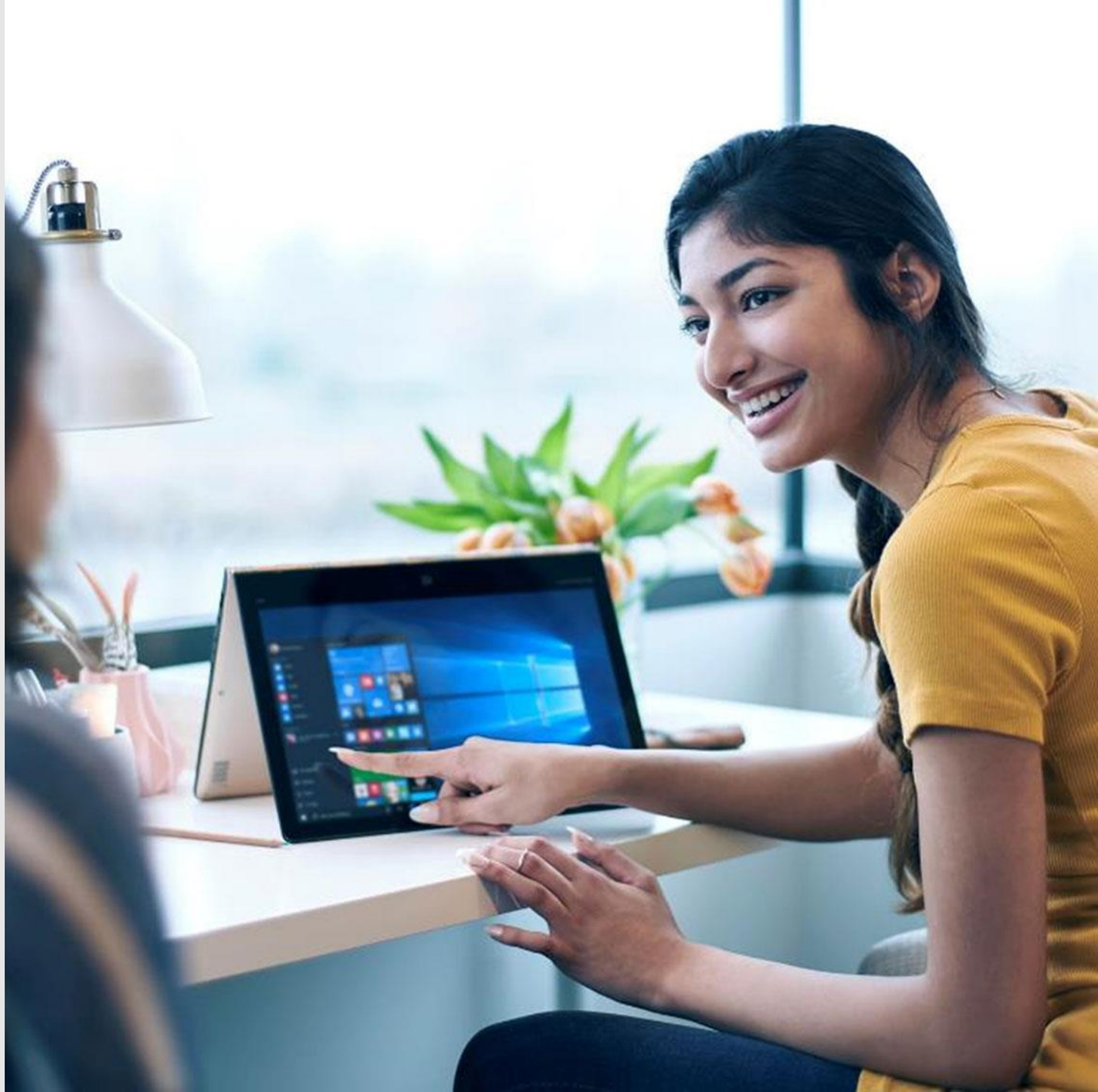
CHEF LaunchDarkly

VAGRANT

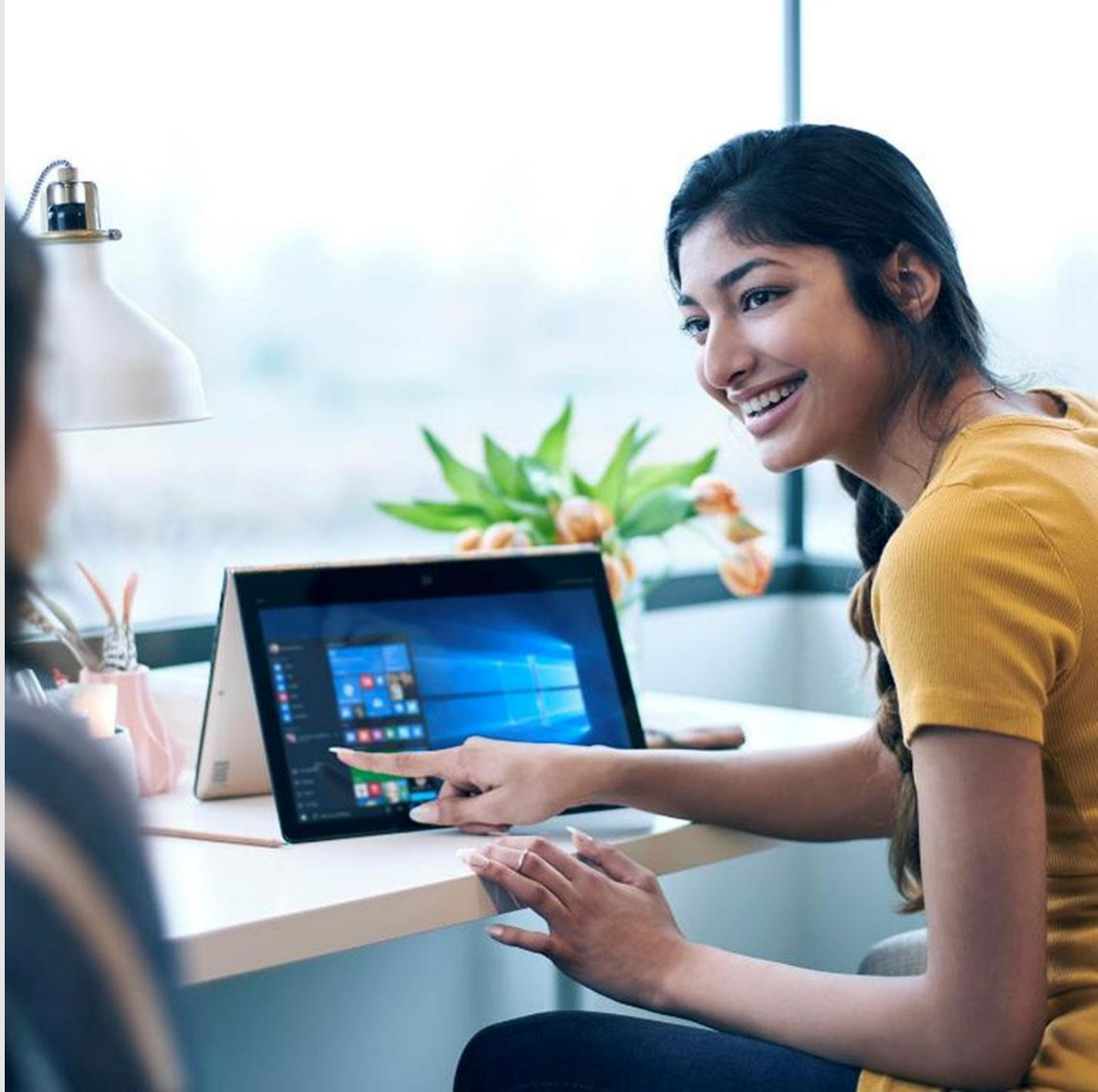
puppet ANSIBLE docker kubernetes

Jenkins Gradle GRUNT

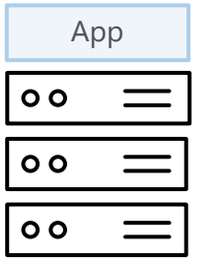
Kubernetes fundamentals



What is container?

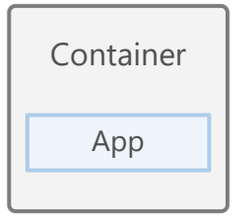


What is a **container**?



Virtual machines

- Virtualize the hardware
- VMs as units of scaling

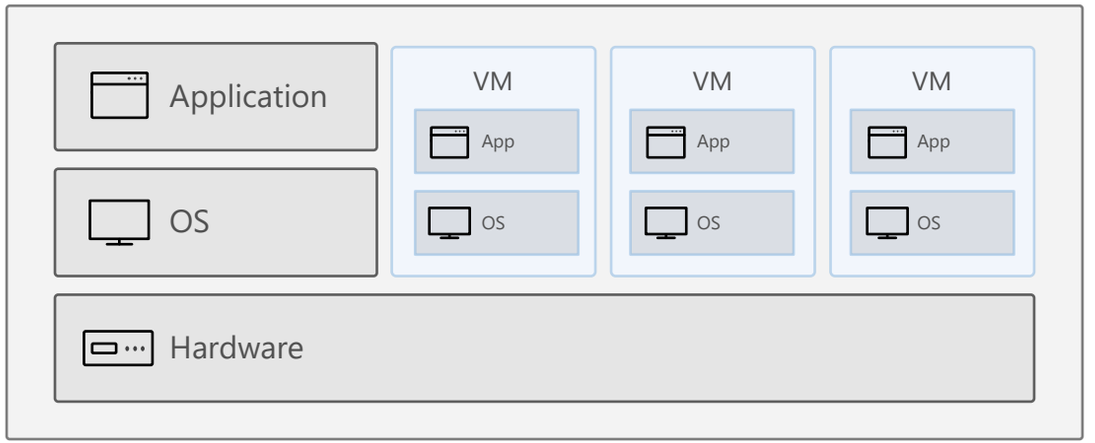


Containers

- Virtualize the operating system
- Applications as units of scaling

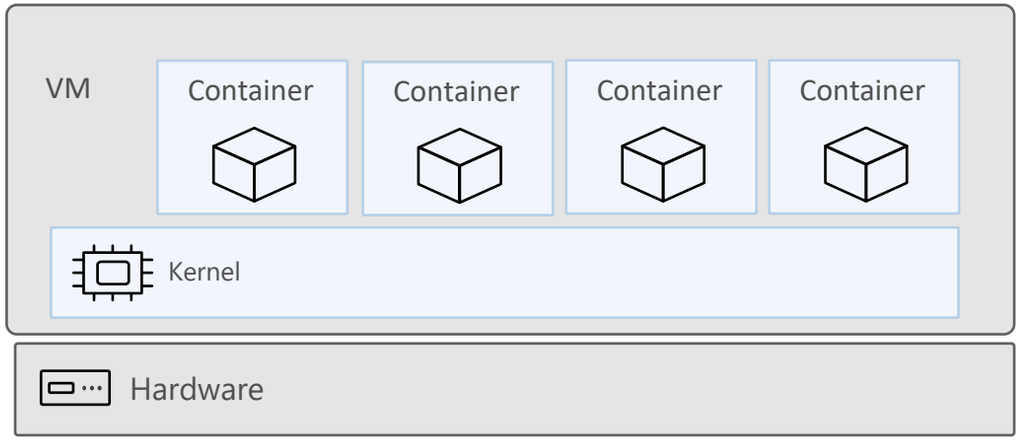
What is a container?

Traditional virtual machines = hardware virtualization



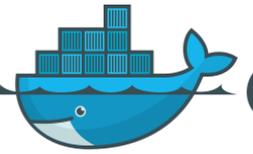
- Large footprint
- Slow to bootup
- Poor utilization of computing resource
- Ideal for long running tasks

Containers = operating system (Kernel) virtualization



- Small footprint
- Quick bring-up time
- Portable
- Better utilization of computing resource
- Agile – ideal for running short-lived/ephemeral tasks

Average lifetime of a container = 2.5 days, VM = 23 days. Containers churn 9 times more than VMs

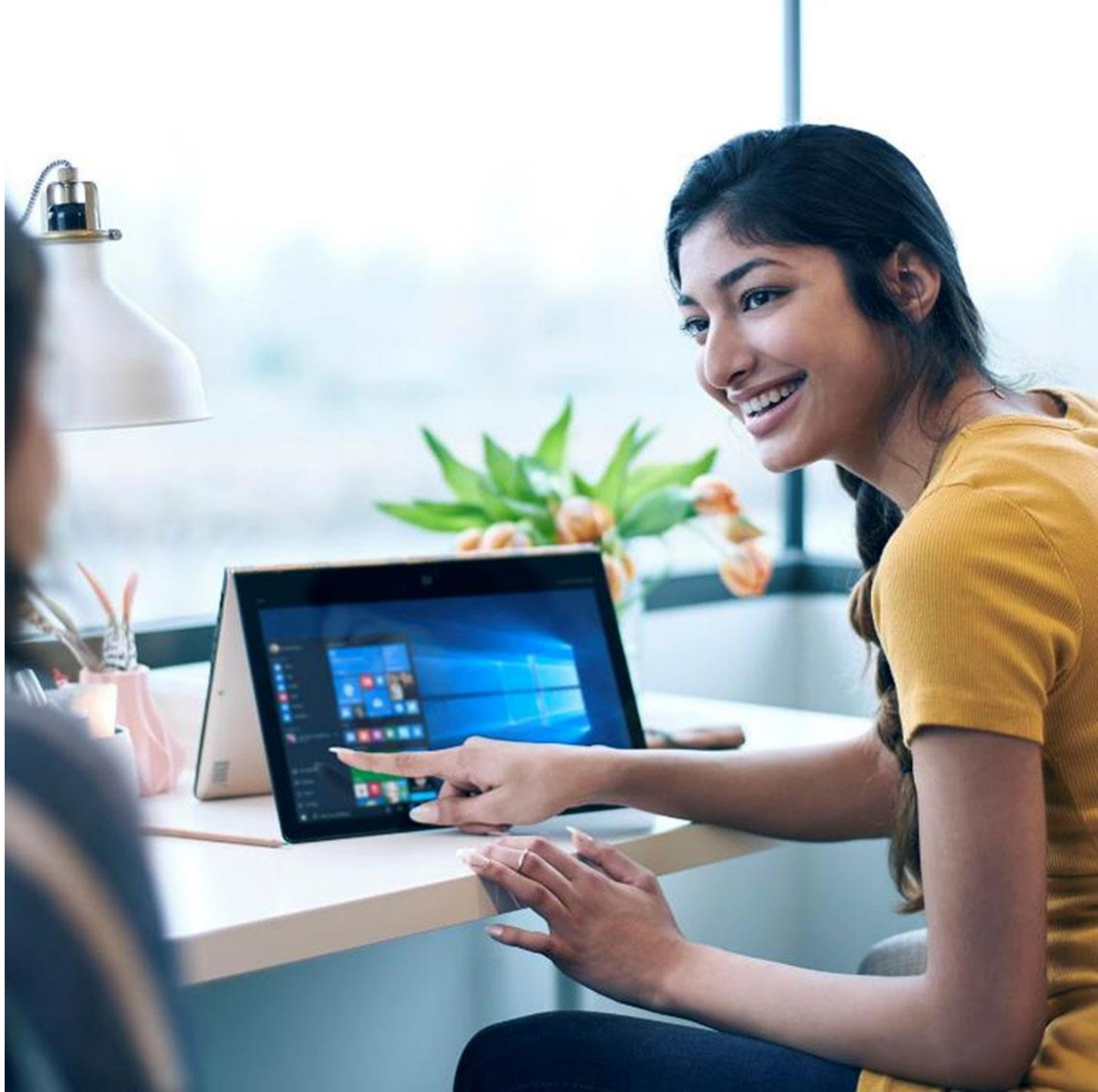
What is  docker?

An open source container runtime

Mac, Windows and Linux support

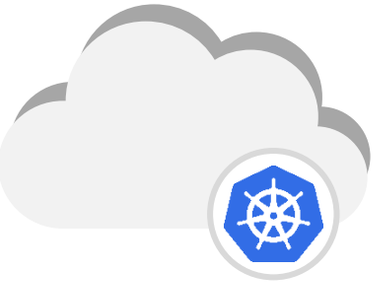
```
# The world’s simplest Dockerfile
$ cat Dockerfile
FROM scratch
COPY hello /
CMD ["/hello"]

# Let’s create a docker image “tagged” hello-world
$ docker build –t hello-world .

# And run it…
$ docker run hello-world
```

What is Kubernetes?



What is Kubernetes (k8s)?

- **Kubernetes** is "an open-source software for automating deployment, scaling, and management of containerized applications".
- **Kubernetes**, in Greek κυβερνήτης, means the Helmsman, or pilot of the ship.
- Keeping with the maritime theme of **Docker** containers, **Kubernetes** is the pilot of a ship of containers.



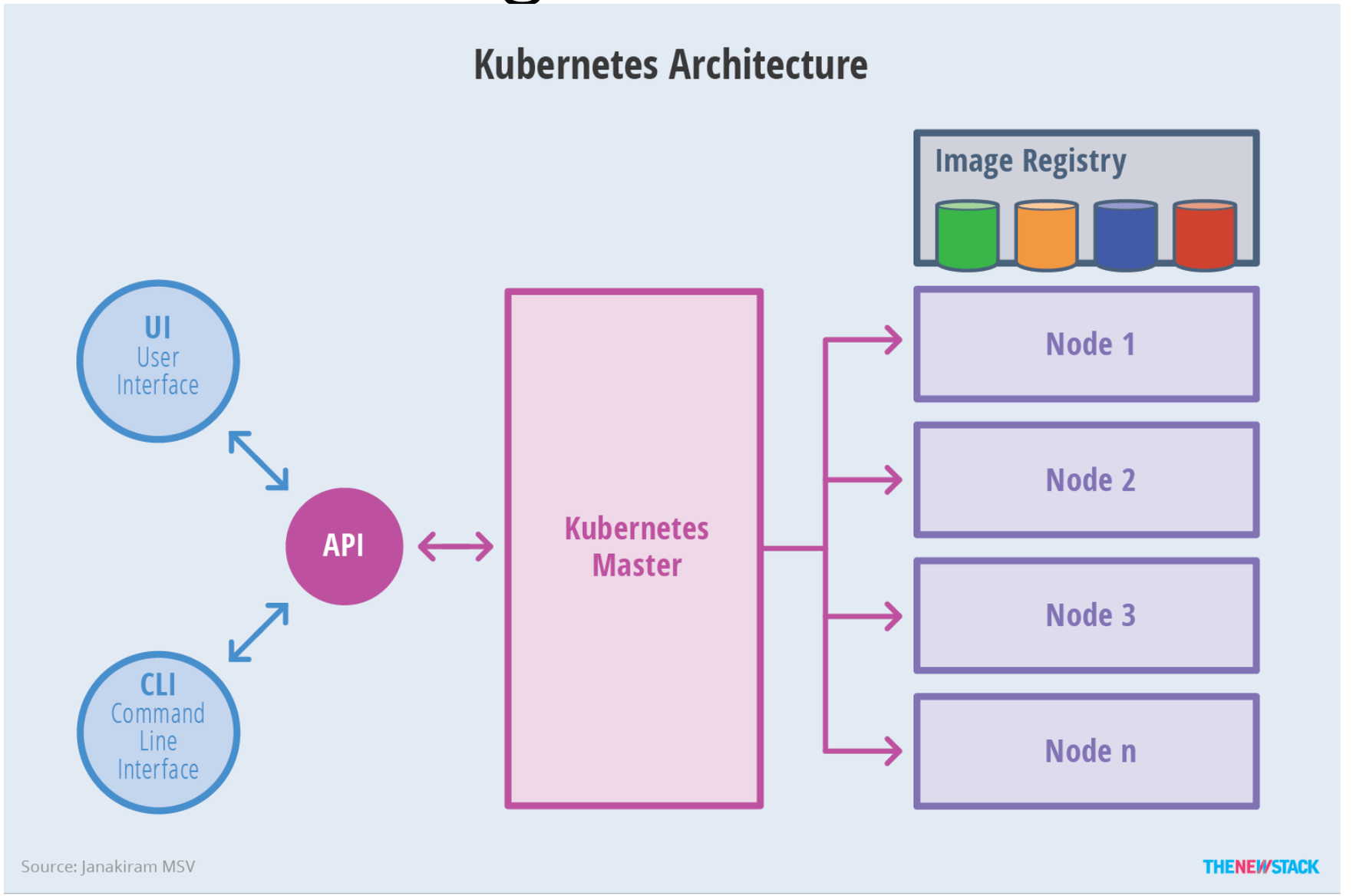
What is Kubernetes (k8s)?

History

- Originally designed by Google and is now maintained by the Cloud Native Computing Foundation (CNCF).
- Google still actively involved
- Kubernetes v1.0 was released on July, 2015 by Joe Beda, Brendan Burns and Craig McLuckie
- Most discussed repo in GitHub last year.
- Over 1,700 authors and releases every three month

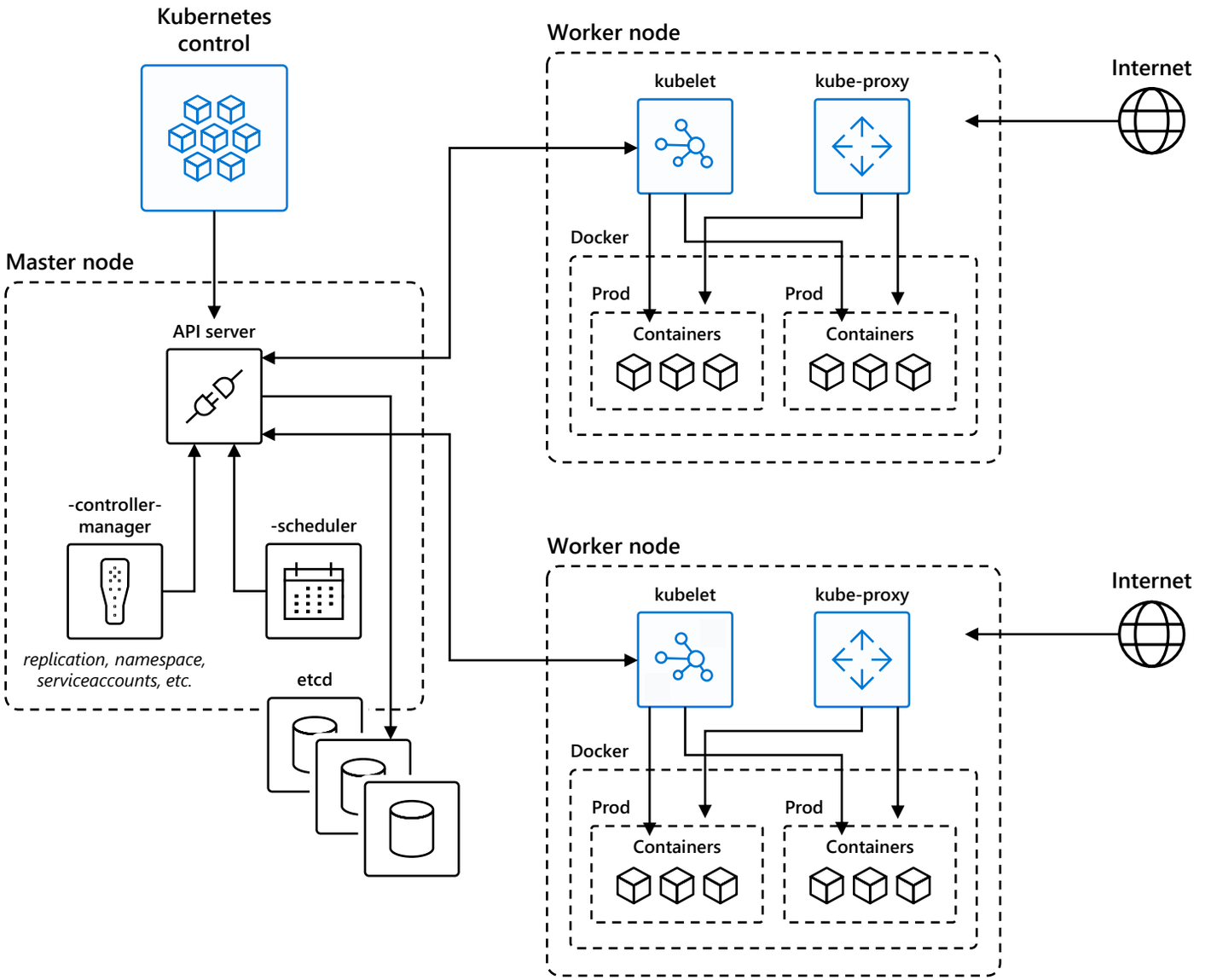
- To learn more about the ideas behind Kubernetes: read the Large-scale cluster management at Google with Borg paper

Kubernetes: High Level Architecture



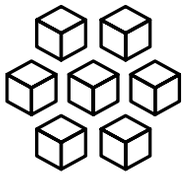
How Kubernetes works

1. Kubernetes users communicate with API server and apply desired state
2. Master nodes actively enforce desired state on worker nodes
3. Worker nodes support communication between containers
4. Worker nodes support communication from the Internet

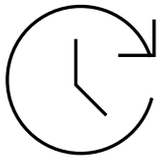


Azure Kubernetes Service (AKS)

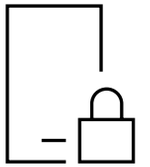
Ship faster, operate easily, and scale confidently with managed Kubernetes on Azure



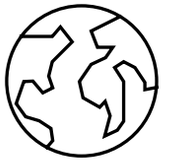
Manage Kubernetes with ease



Accelerate containerized development



Build on an enterprise-grade, secure foundation

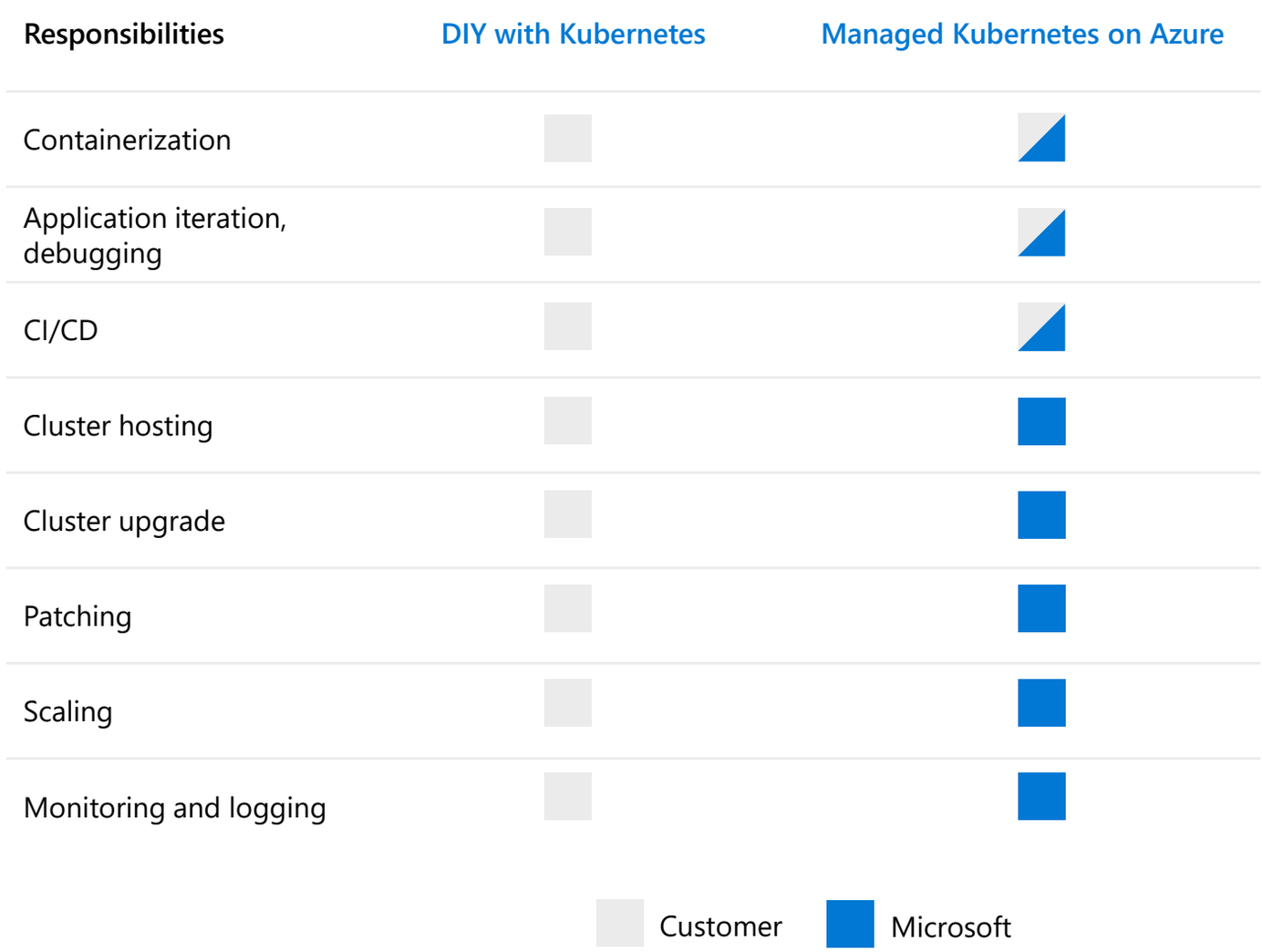


Run anything, anywhere

From infrastructure to innovation

Managed Kubernetes empowers you to achieve more

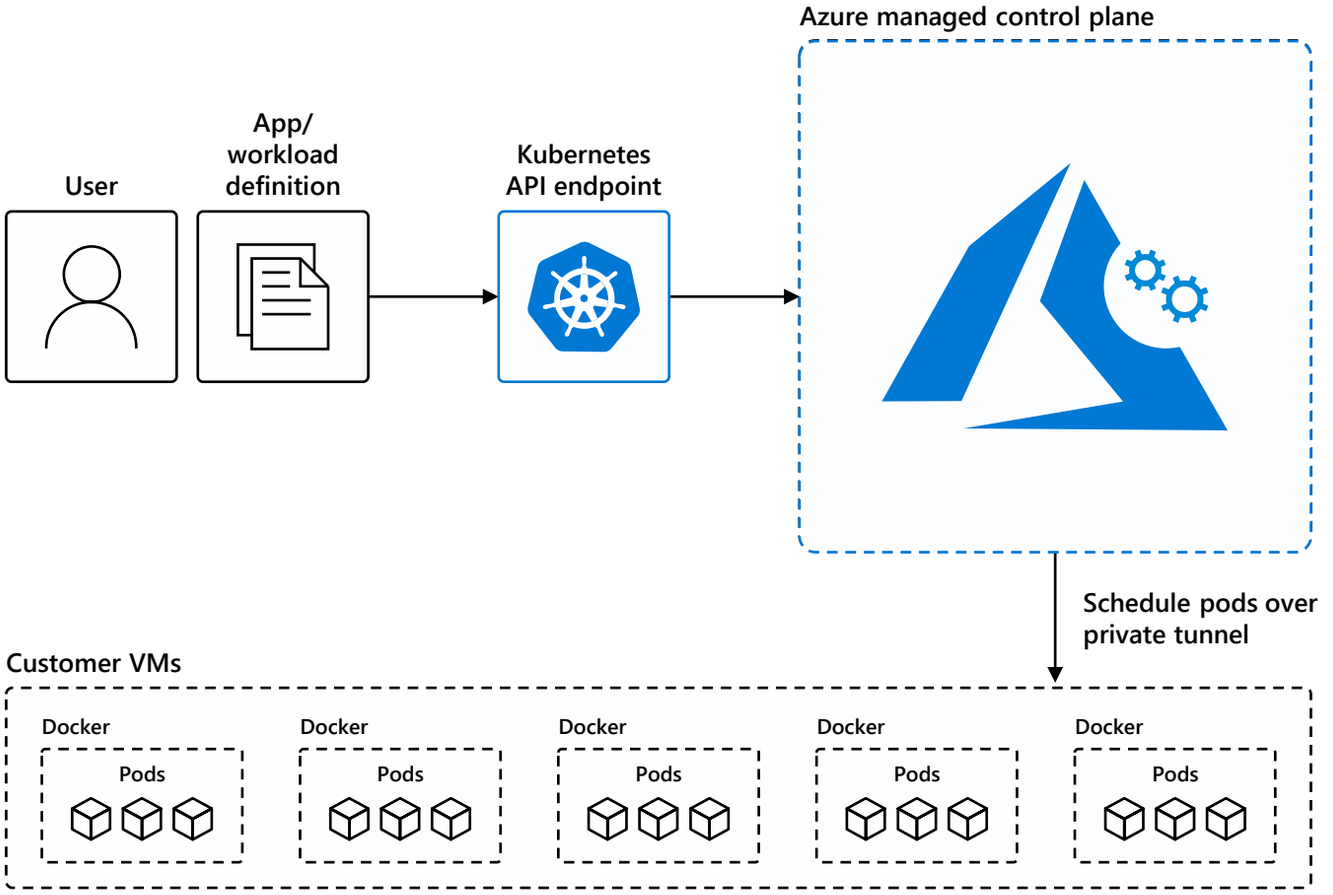
Focus on your containers and code, not the plumbing of them



Manage Kubernetes with ease

Infrastructure automation

- Automated provisioning, upgrades, patches
- High reliability, availability
- Easy, secure cluster scaling
- Self-healing
- API server monitoring
- At no charge



Build on a secure, enterprise-grade platform

Control access through AAD and RBAC

Secure network communications with VNET and network policy

Put guardrails in your development process with Azure Policy

10.35 nadaljevanje

Create your first AKS Cluster

```
az aks create \
    --resource-group event-matper-aks-rg \
    --name aks01 \
    --node-count 3 \
    --enable-addons monitoring \
    --generate-ssh-keys \
    --disable-rbac \
    --node-vm-size Standard_B4ms
```

Microsoft Azure

DEMO
Create AKS cluster

Microsoft

Kubernetes Resources

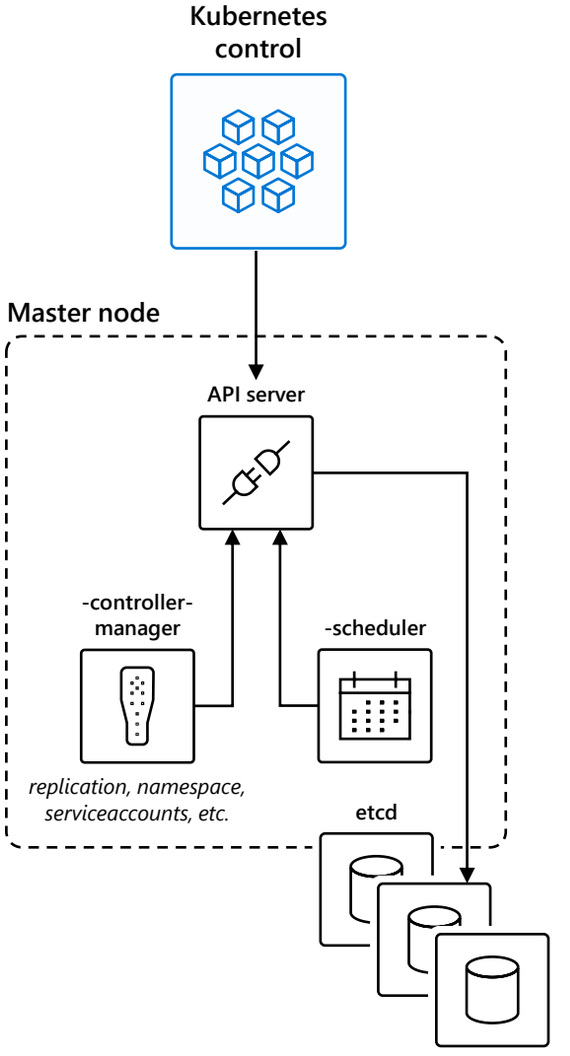
Kubernetes Resources

pod	deployment	StatefulSet
service	volumes	ReplicaSet
ingress	DaemonSet	jobs
namespace	secret	config-map

kubectl

master components

- CLI to run commands against a Kubernetes cluster
  - Swiss Army Knife: run deployments, exec into containers, view logs, etc.
  - Pronounced “koob sea tee el” or “koob cuddle”
  - Available for Windows and Linux – of course available in Azure Cloud Shell



Declarative vs. Imperative

- Commands like kubectl run and kubectl expose are **imperative** commands (do this thing now)

```
$> kubectl run -i --tty busybox --image=busybox --restart=Never -- sh
```

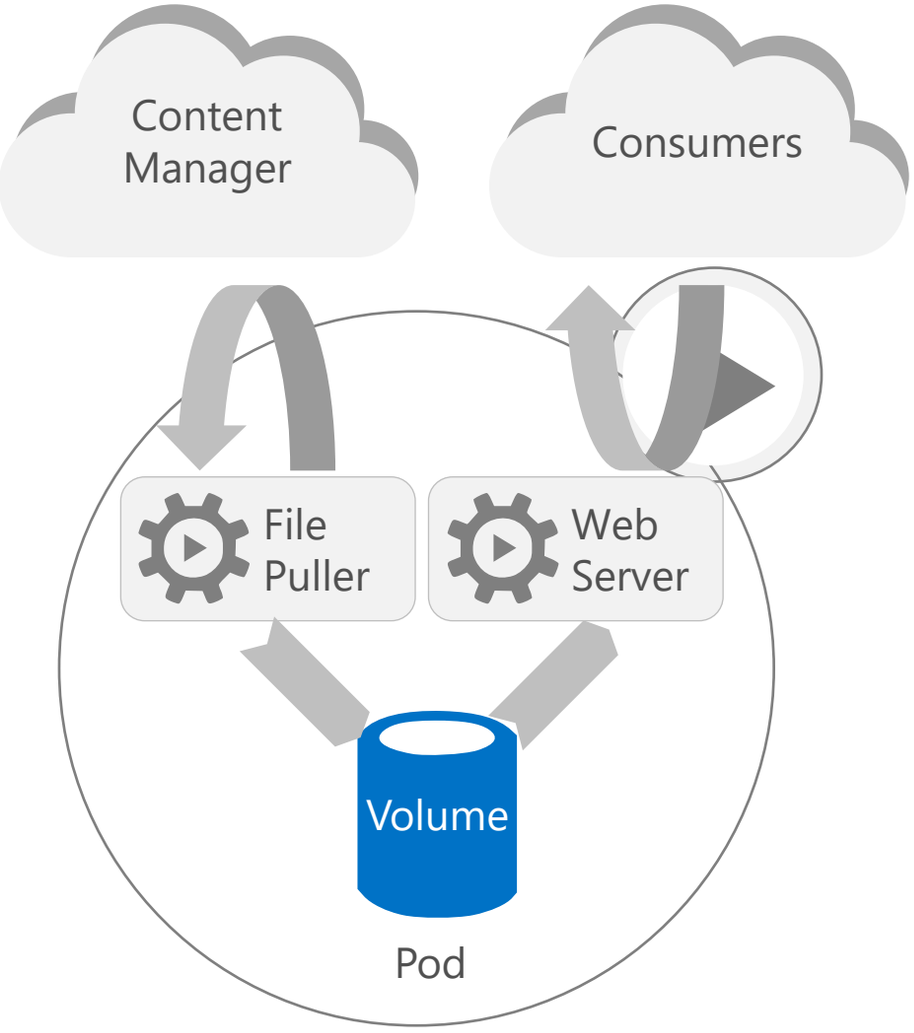
- **Declarative** way – Describe the state of resources in a file (JSON or YAML).

  kubectl apply –f webresource.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
```

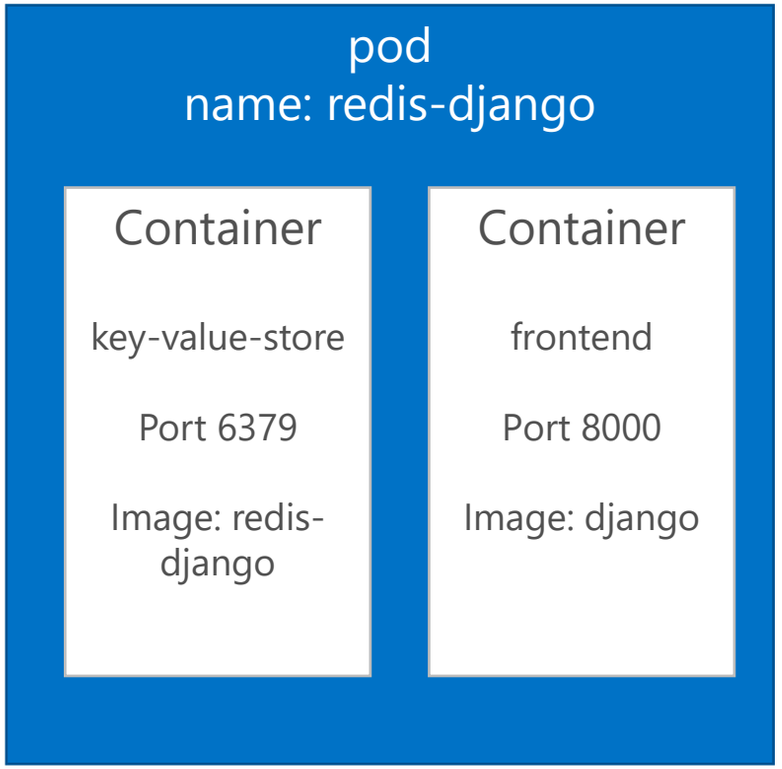
What is a pod?

- Pod is the basic building block in Kubernetes
- Pods are how containers are delivered
- Can be multiple containers (e.g. side car)
- Encapsulates container(s), storage, network IP, and options on how to run



Kubernetes manifest: Pod

```
apiVersion: v1
kind: Pod
metadata:
  name: redis-django
  labels:
    app: web
spec:
  containers:
    - name: key-value-store
      image: redis
      ports:
        - containerPort: 6379
    - name: frontend
      image: django
      ports:
        - containerPort: 8000
```



Controllers - Deployment

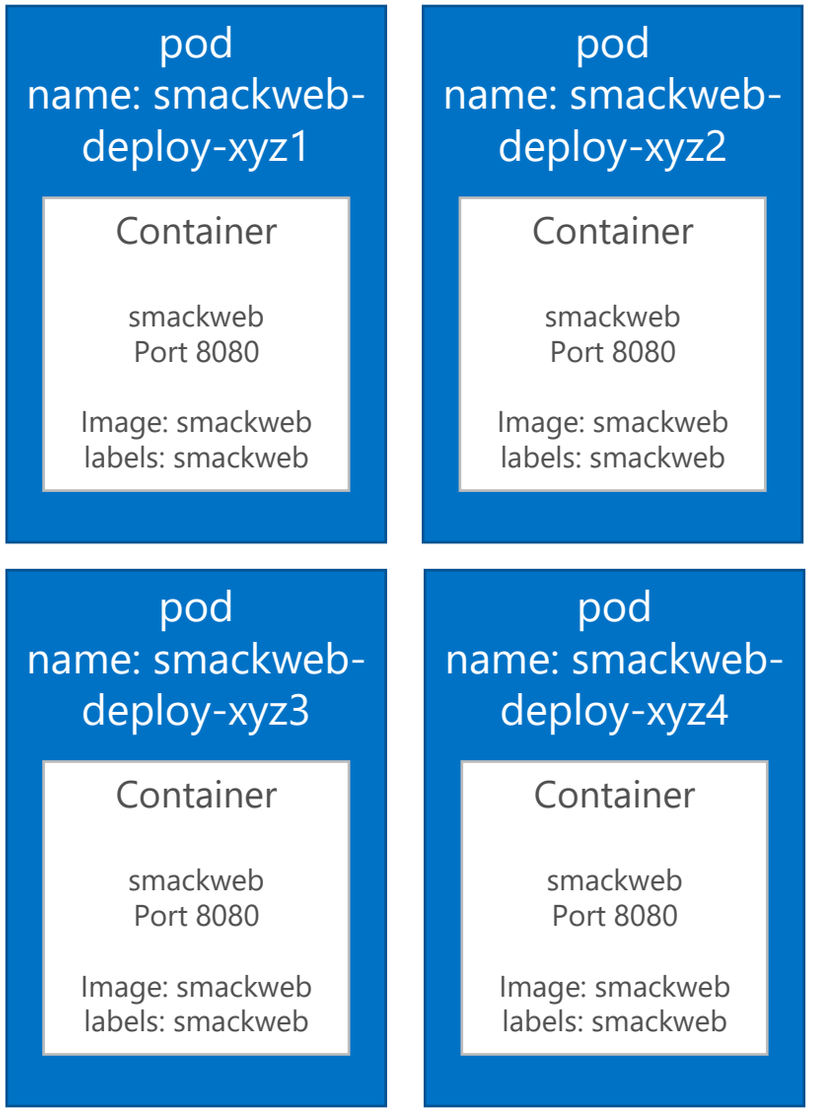
Deployment provides declarative updates for Pods and ReplicaSets.

Use Cases:

- Create deployment to rollout **ReplicaSet**
- Declare new state for pods (eg – new imageTag)
- Rollback to earlier revision
- Scale up or down
- Check rollout history
- Clean-up older ReplicaSets

Kubernetes manifest: Deployment

```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: smackweb-deploy
spec:
  selector:
    matchLabels:
      app: smackweb
  replicas: 4
  template:
    metadata:
      labels:
        app: smackweb
    spec:
      containers:
      - name: smackweb
        image: chzbrgr71/smackweb
        ports:
        - containerPort: 8080
```

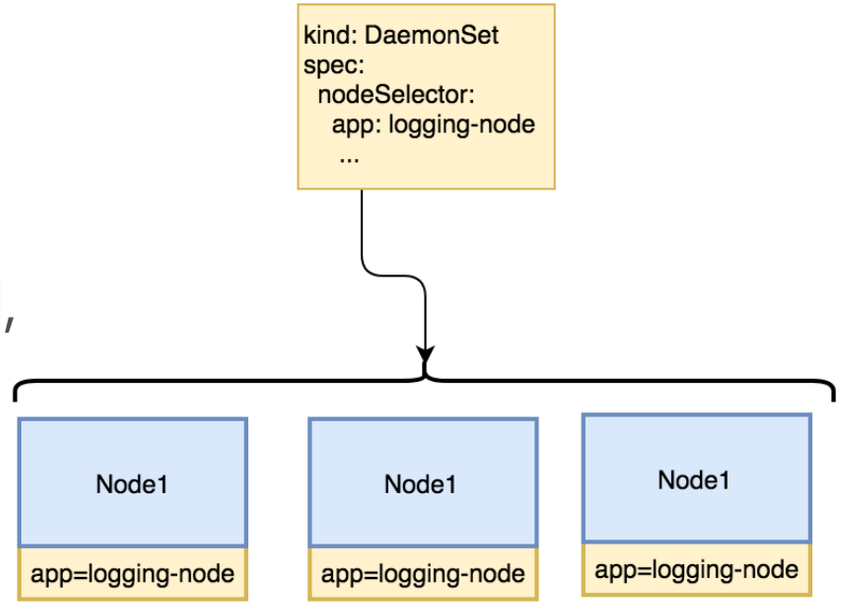


Controllers - DaemonSets

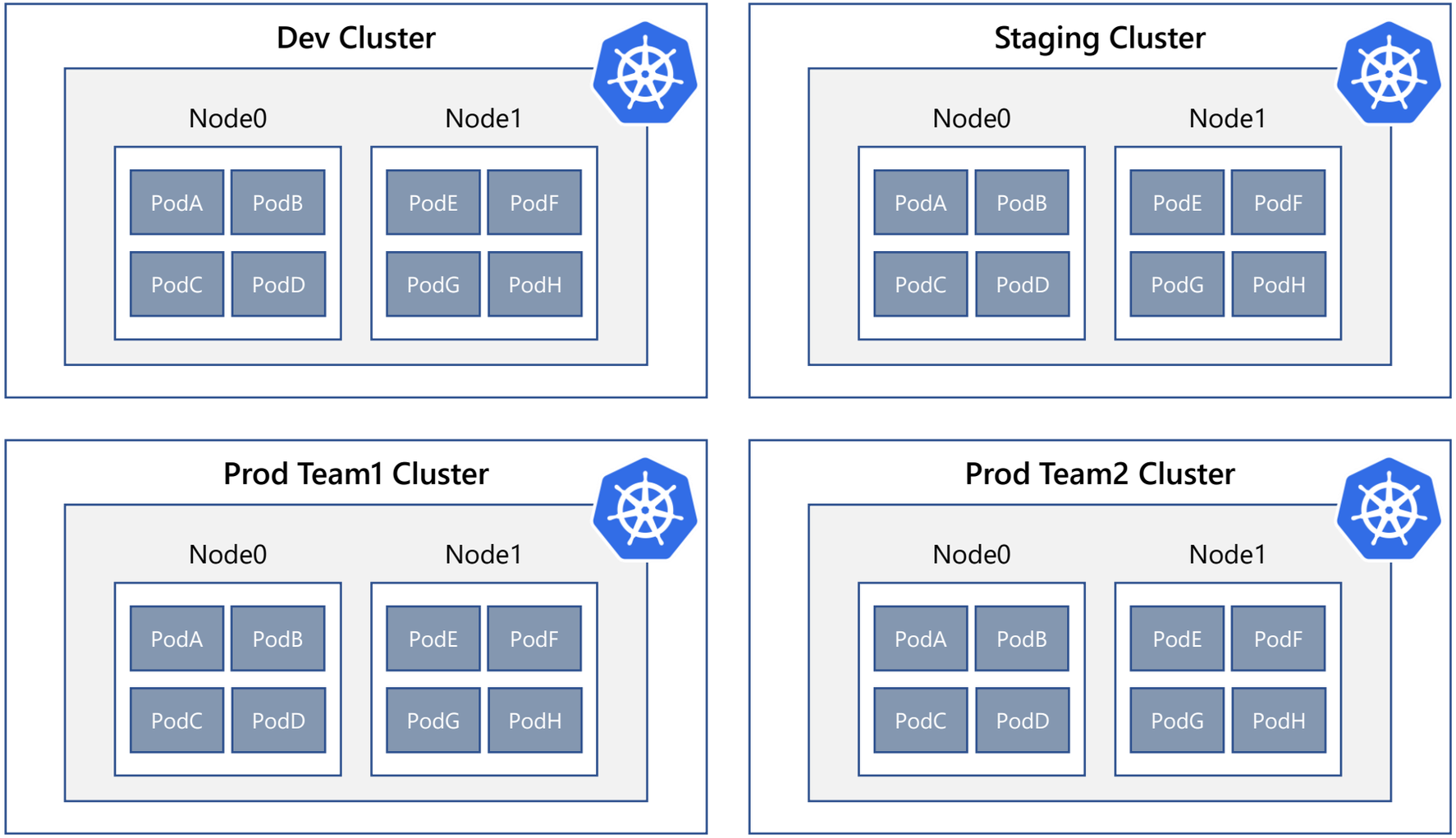
DaemonSets ensure that all (or some) Nodes run a copy of a Pod.

- As worker nodes are
  - added to the cluster, Pods are added to them.
  - removed from the cluster, those Pods are garbage collected.

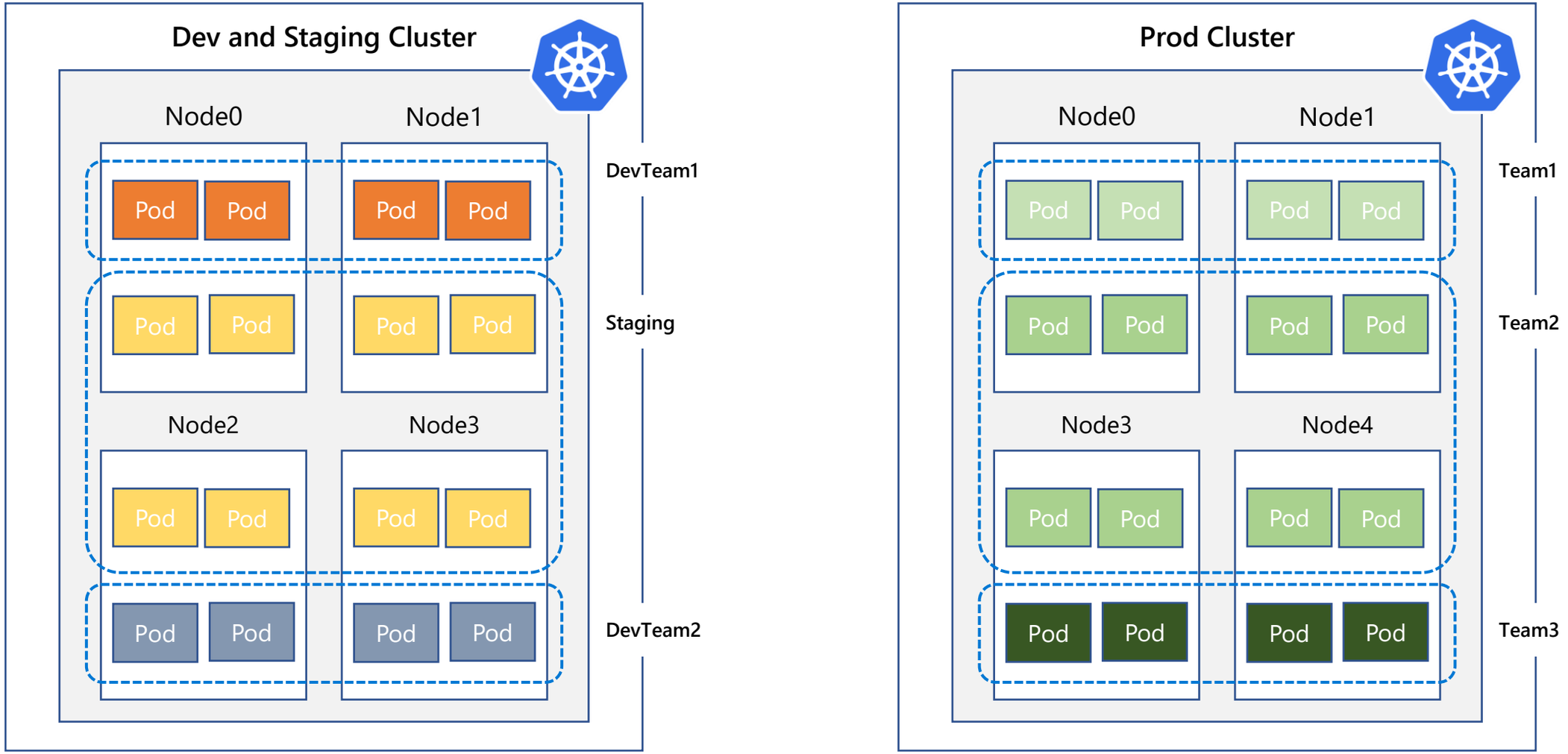
- Some typical uses of a DaemonSet are:
  - logs collection daemon (i.e. fluentd, logstash)
  - Malware scan (install AV)
  - node monitoring daemon (i.e. Prometheus, collectd, Datadog, New Relic)



Cluster Isolation Patterns: Physical Isolation



Cluster Isolation Patterns: Logical Isolation



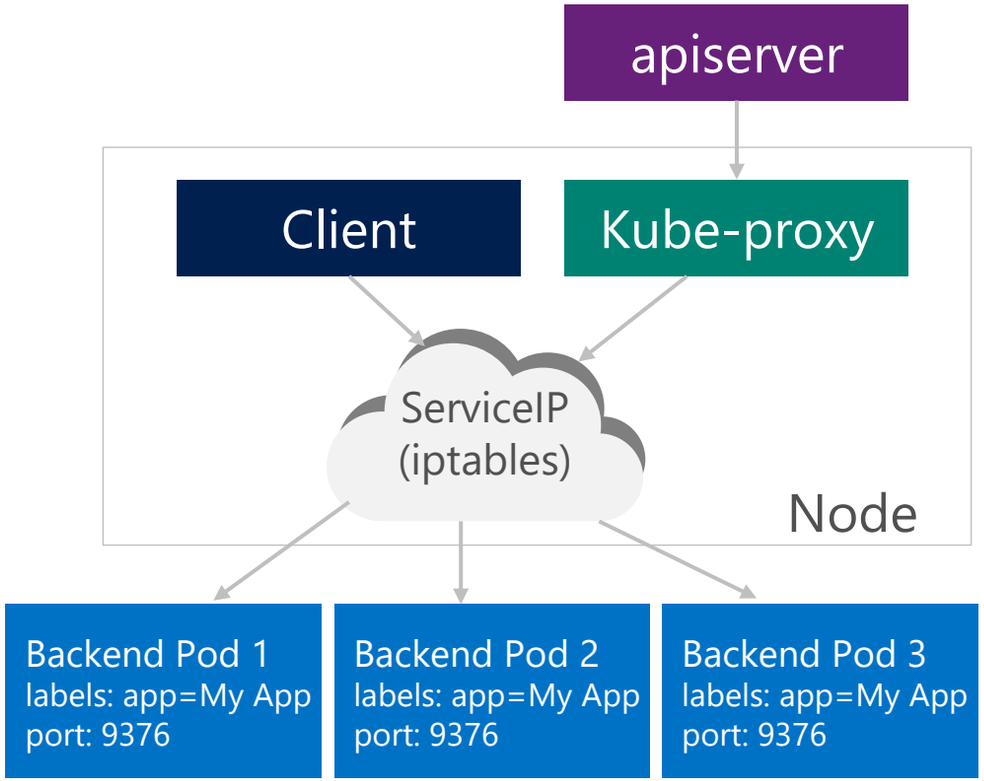
Namespaces

- multiple virtual clusters backed by the same physical cluster
- logical separation/isolation
- Every resource type is scoped to a namespace
  (except for nodes, persistentVolumes, etc.)
- Intended for environments with many users, teams, projects
- Kube-system namespace for dashboard etc.

```
wslroot@MININT-O84LOJC:~$ kubectl get namespaces
NAME          STATUS    AGE
default       Active    3d
kube-public   Active    3d
kube-system   Active    3d
```

Kubernetes Services

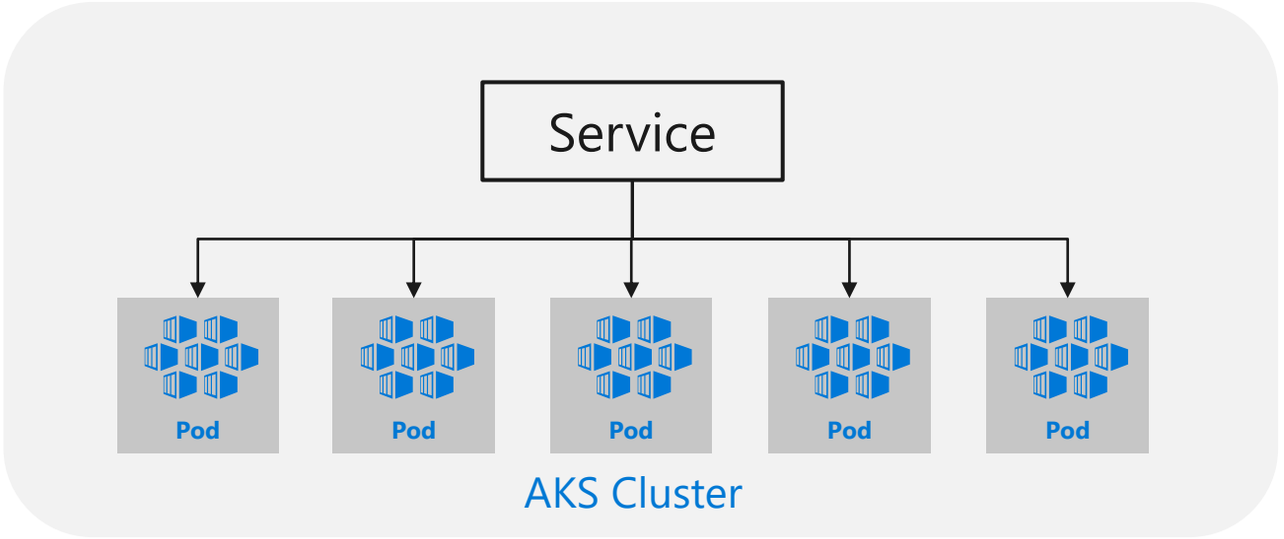
- Defines a logical set of pods
- Identified/selected using Labels



- Essentially a virtual load balancer in front of pods

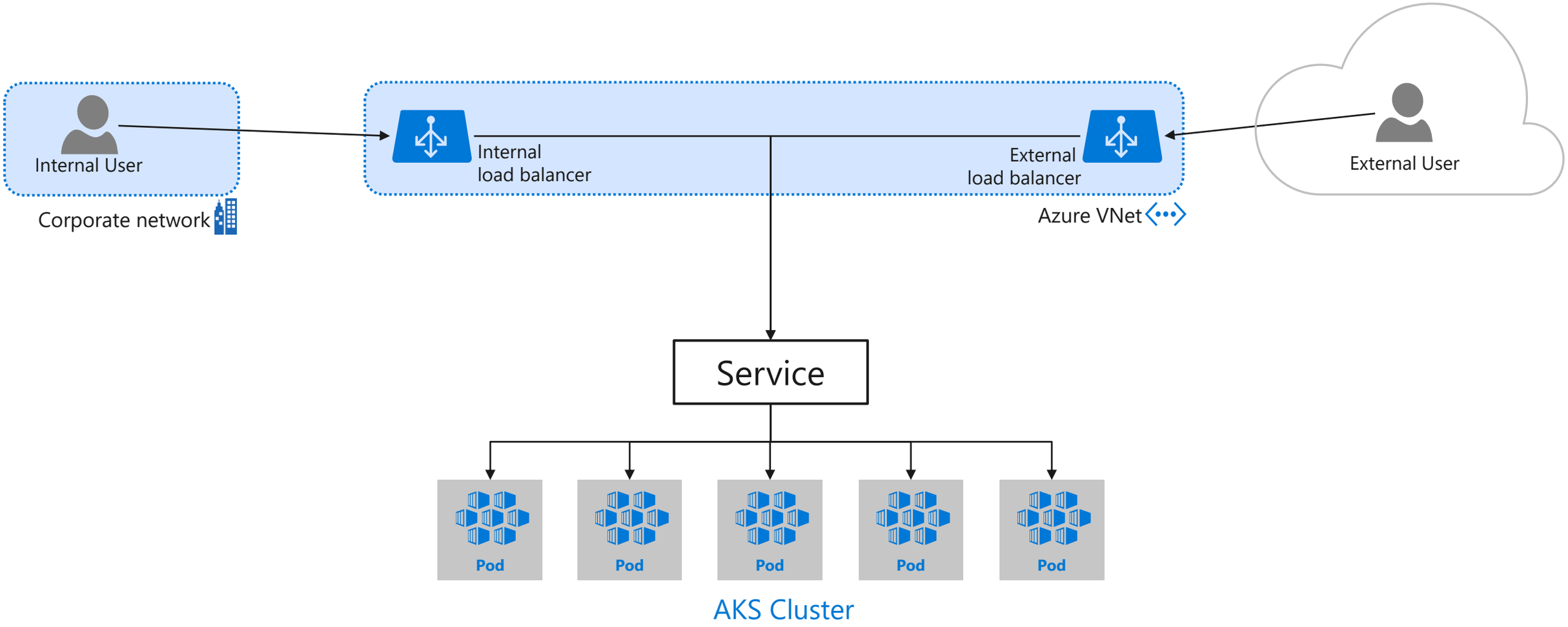
Exposing services (application)

Inside the cluster (ClusterIP)



Exposing services

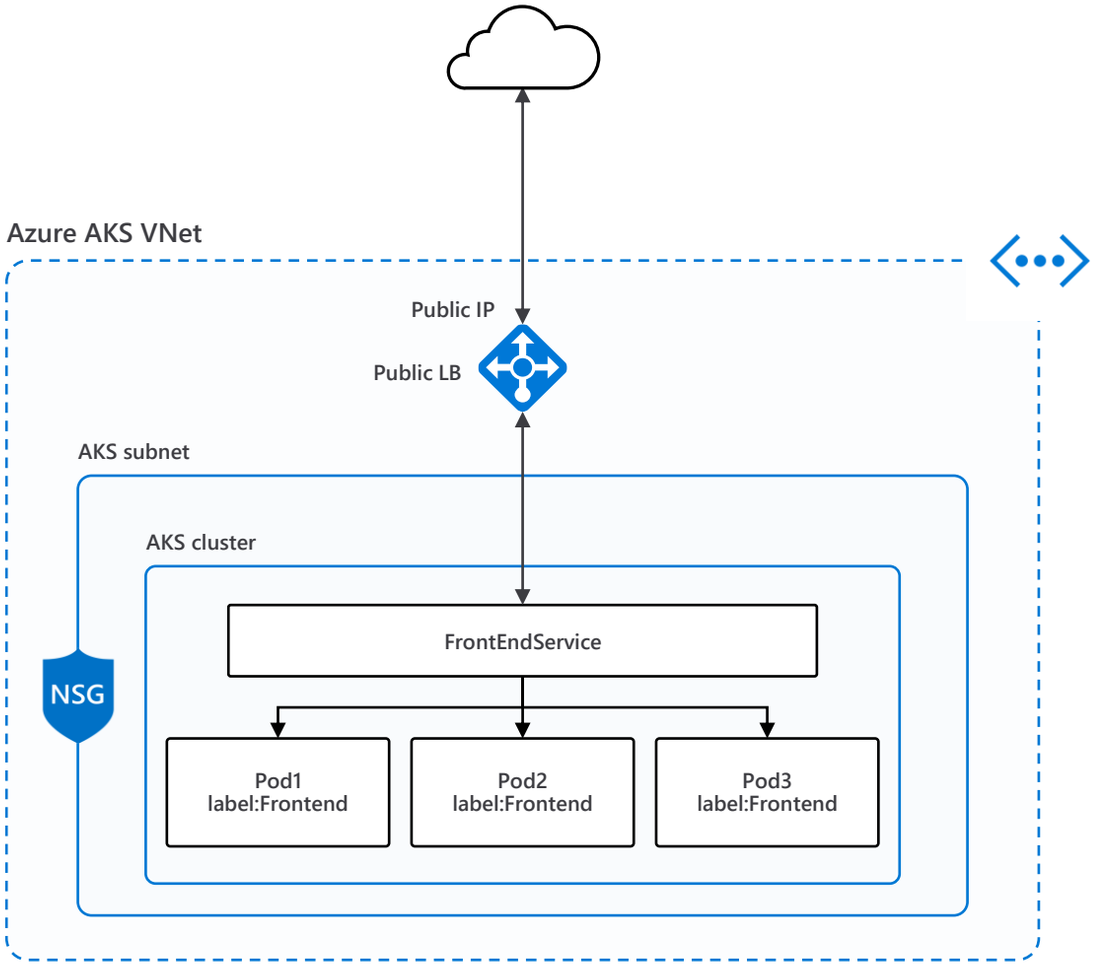
Outside the cluster – internal or external network (layer 4)



Public LoadBalancer Service

- Service Type LoadBalancer
- Basic Layer4 Load Balancing (TCP/UDP)
- Each service has assigned an IP on the ALB

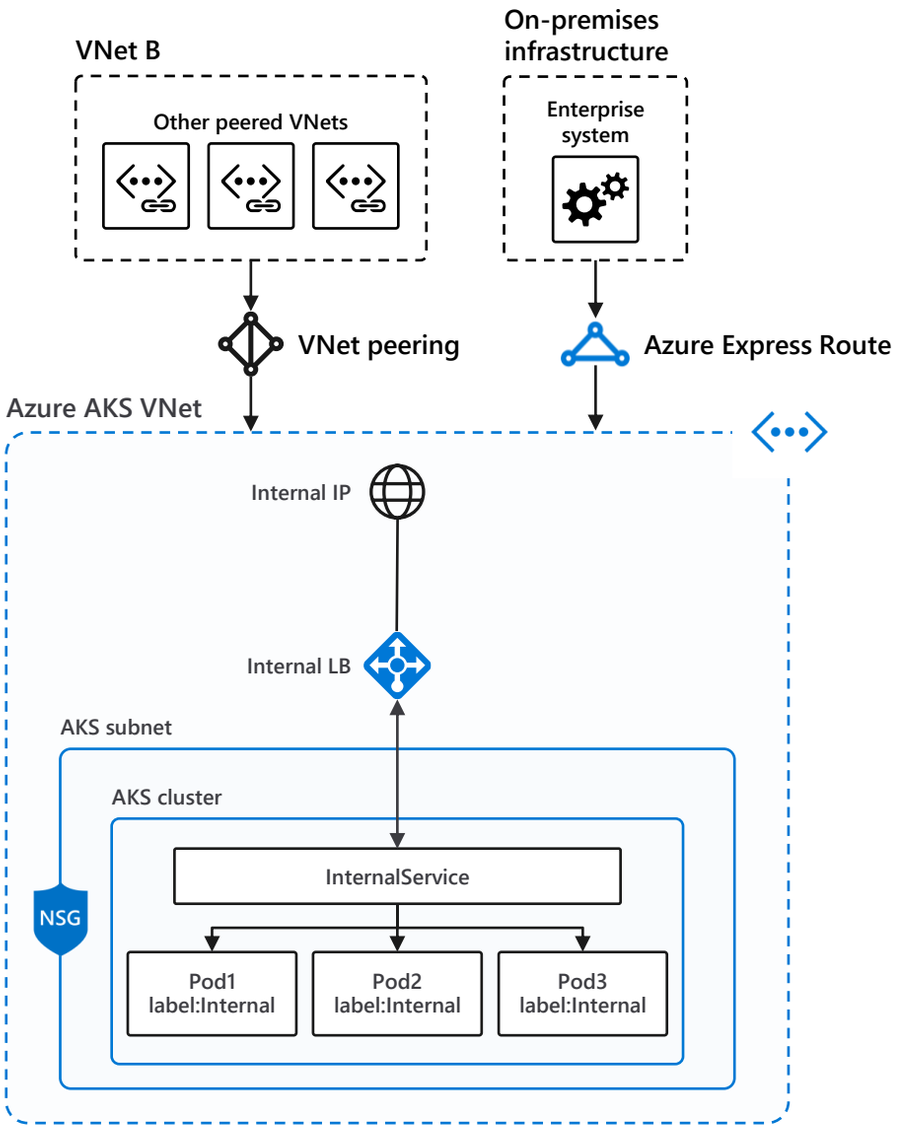
```yaml
apiVersion: v1
kind: Service
metadata:
  name: frontendservice
spec:
  loadBalancerIP: X.X.X.X
  type: LoadBalancer
  ports:
  - port: 80
  selector:
    app: frontend
```



Internal Service

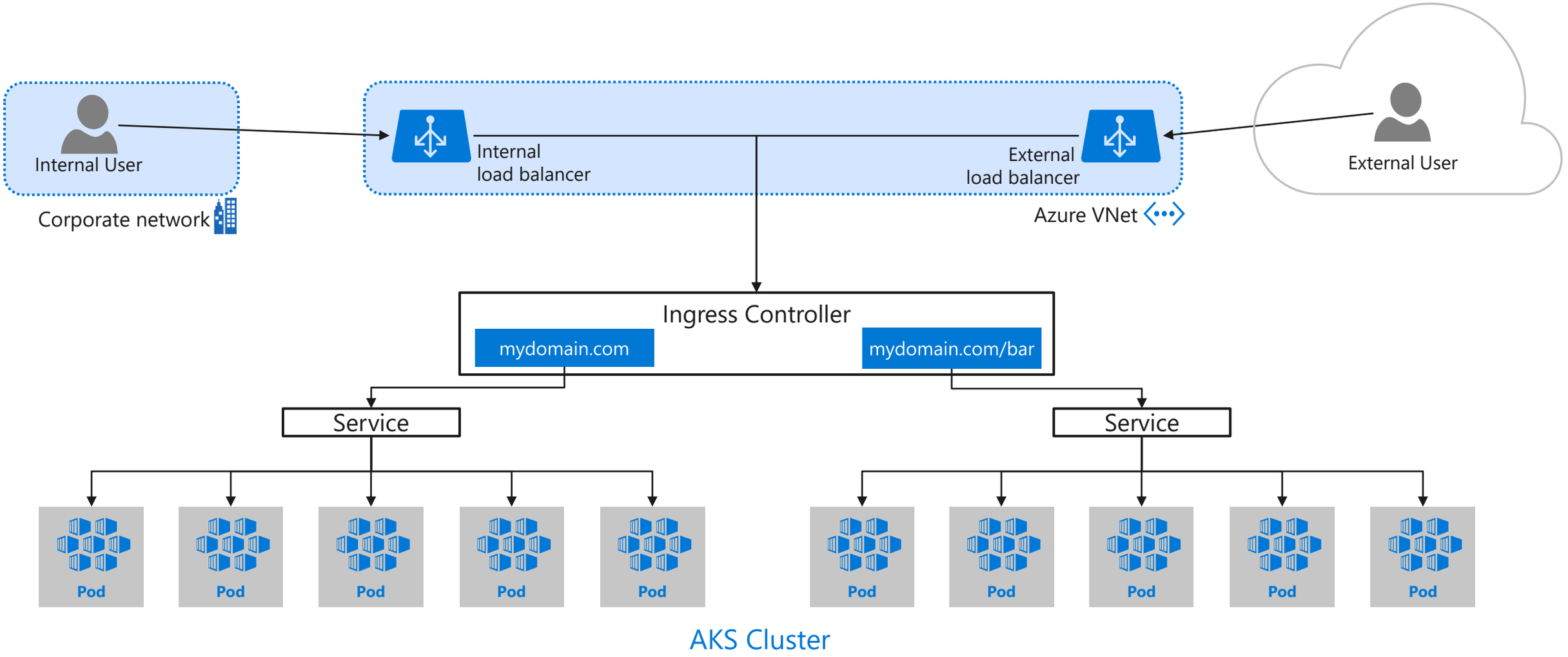
- Used for internal services that should be accessed by other VNETs or On-Premise only

```
apiVersion: v1
kind: Service
metadata:
  name: internalservice
  annotations:
    service.beta.kubernetes.io/azure-load-balancer-internal:
"true"
spec:
  type: LoadBalancer
  loadBalancerIP: 10.240.0.25
  ports:
  - port: 80
  selector:
   app: internal
```



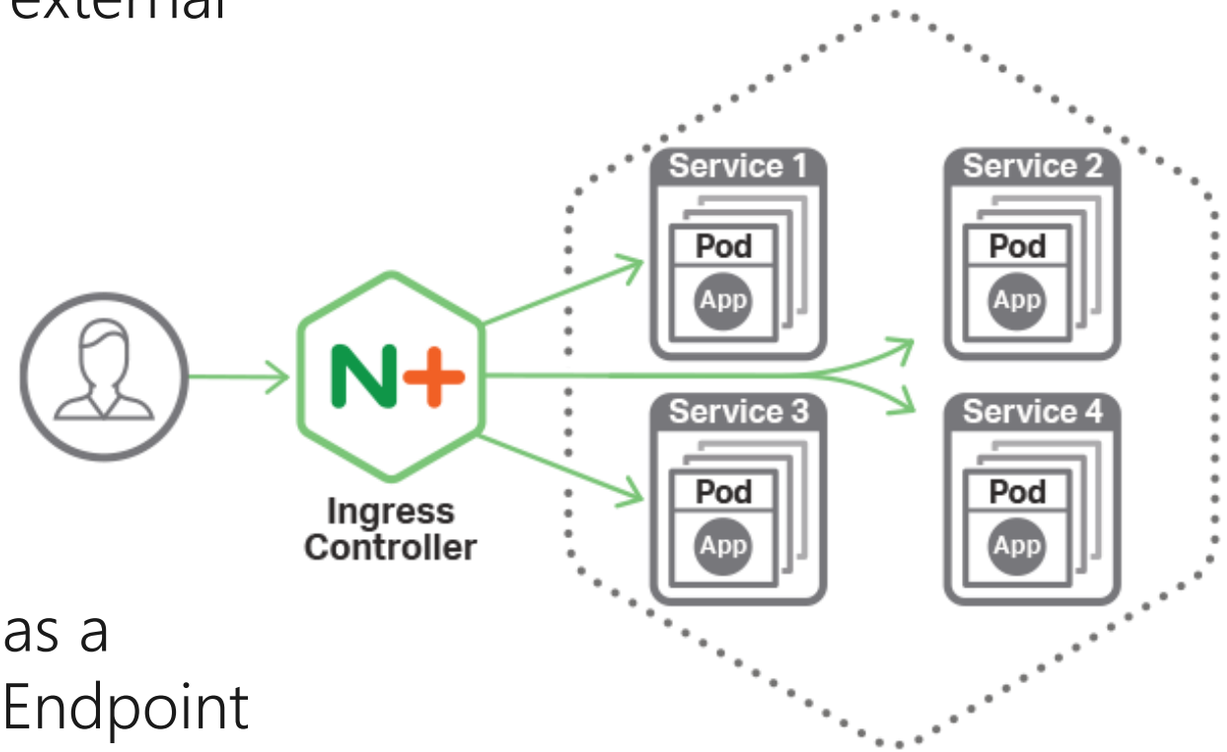
Exposing services

Outside the cluster – internal or external network (HTTP application routing)



Ingress and Ingress Controllers

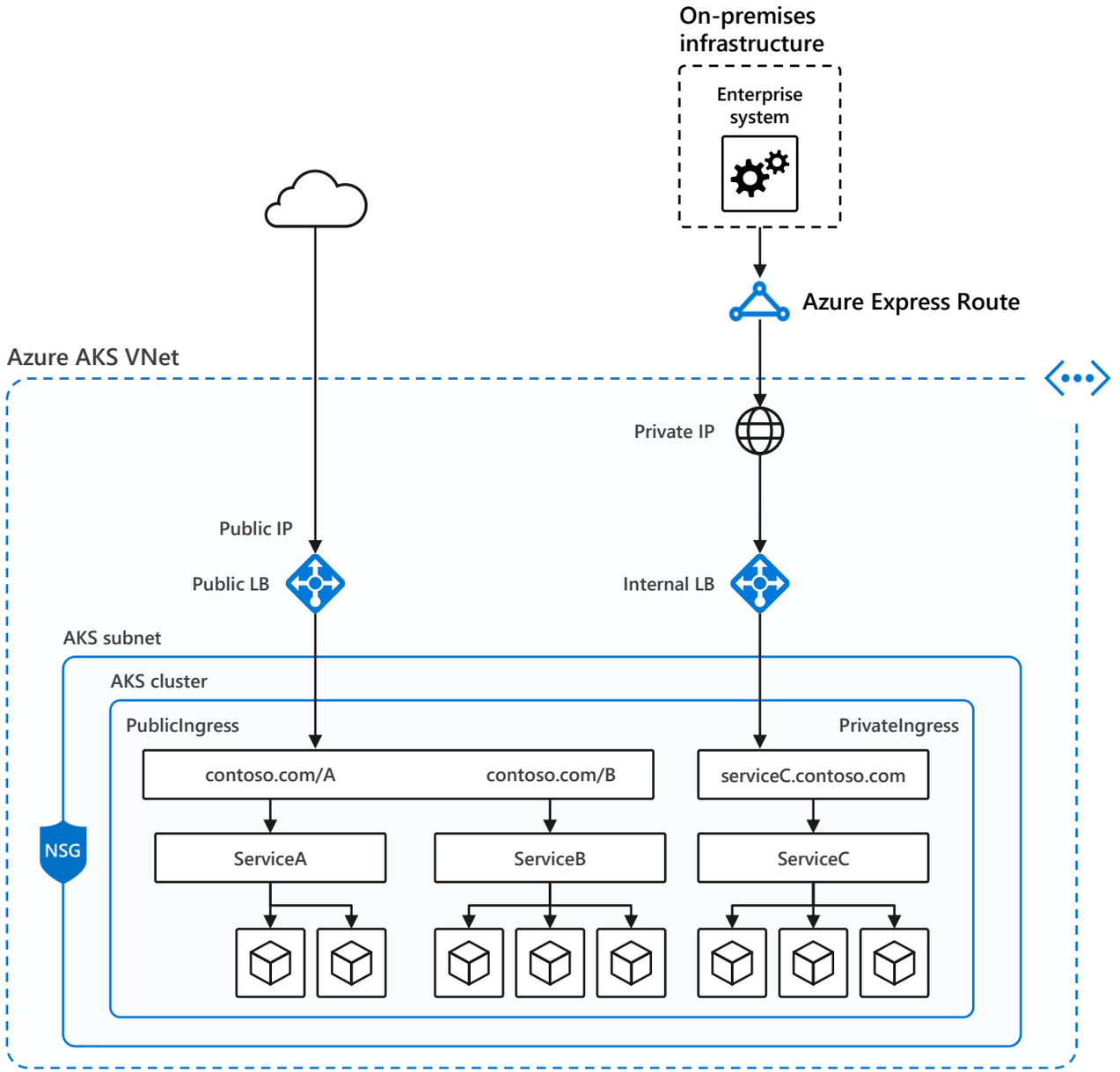
- Ingress is a Kubernetes API that manages external access to the services in the cluster
  - Supports HTTP and HTTPs
  - Path and Subdomain based routing
  - SSL Termination
  - Save on public IPs



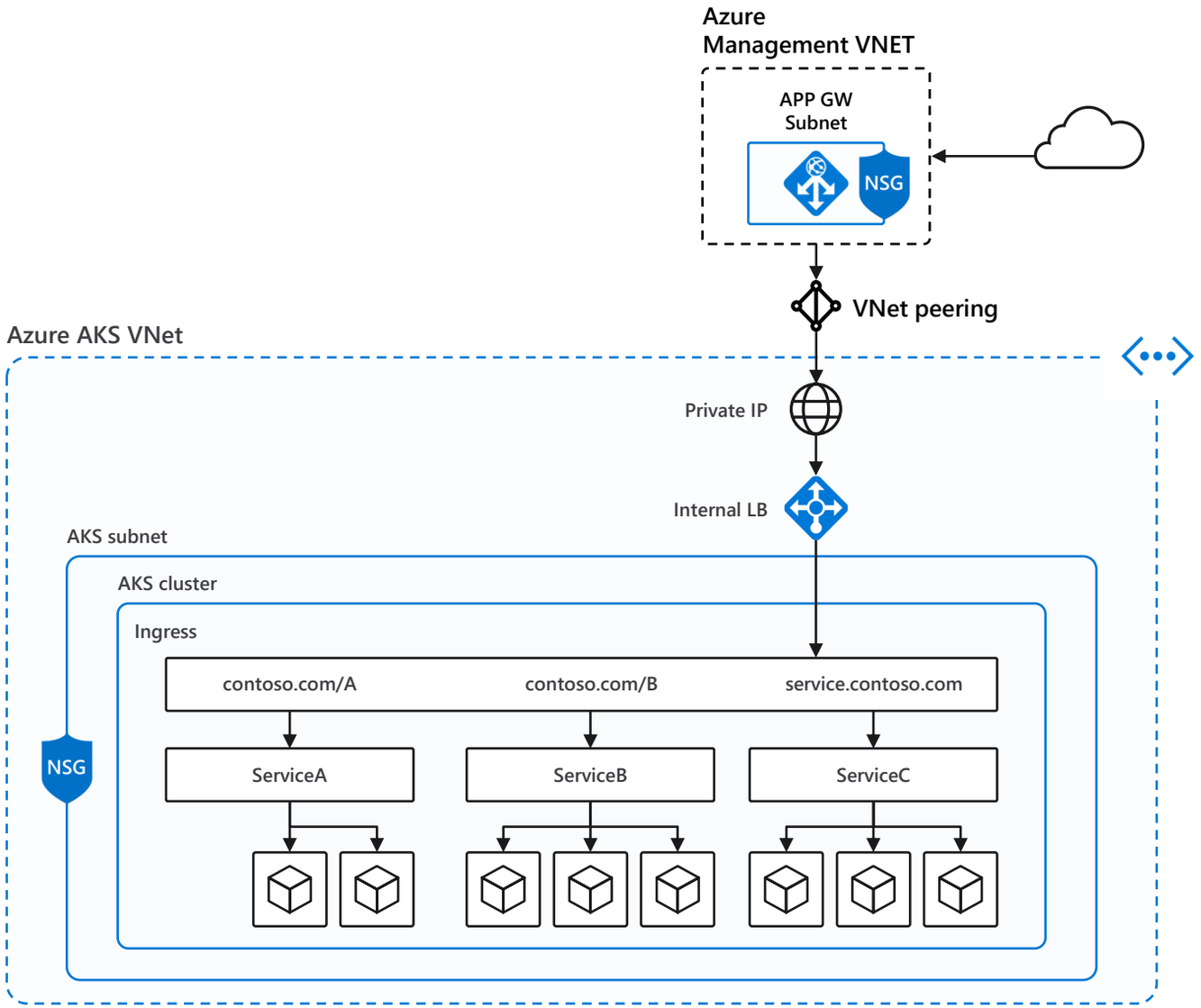
- Ingress controller is a daemon, deployed as a Kubernetes Pod, that watches the Ingress Endpoint for updates. Its job is to satisfy requests for ingresses. Most popular one being Nginx.

Ingress

```
kind: Ingress
metadata:
 name: contoso-ingress
  annotations: kubernetes.io/ingress.class:
”PublicIngress"
spec:
 tls:
 - hosts:
  - contoso.com
  secretName: contoso-secret
 rules:
  - host: contoso.com
    http:
     paths:
     - path: /a
       backend:
        serviceName: servicea
        servicePort: 80
     - path: /b
       backend:
        serviceName: serviceb
        servicePort: 80
```



Securing Kubernetes Services with WAF



Secrets, Config Maps

- **Secrets** are intended to hold sensitive information such as passwords, tokens. Secrets are for Confidential data. Secrets are encoded with Base64 encoding
- **ConfigMaps** help you to store non-confidential application configuration data. This helps to decouple configuration artifacts from image content

```
$ kubectl create secret generic db-user-pass --from-file=./username.txt --from-file=./password.txt
secret "db-user-pass" created
```

kubectl Cheat Sheet

Updating Resources

As of version 1.11 `rolling-update` have been deprecated (see CHANGELOG-1.11.md), use `rollout` instead.

```
kubectl set image deployment/frontend www=image:v2               # Rolling update "www" containers of "frontend" deployment, updating the image
kubectl rollout undo deployment/frontend                         # Rollback to the previous deployment
kubectl rollout status -w deployment/frontend                    # Watch rolling update status of "frontend" deployment until completion

# deprecated starting version 1.11
kubectl rolling-update frontend-v1 -f frontend-v2.json           # (deprecated) Rolling update pods of frontend-v1
kubectl rolling-update frontend-v1 frontend-v2 --image=image:v2  # (deprecated) Change the name of the resource and update the image
kubectl rolling-update frontend --image=image:v2                 # (deprecated) Update the pods image of frontend
kubectl rolling-update frontend-v1 frontend-v2 --rollback        # (deprecated) Abort existing rollout in progress

cat pod.json | kubectl replace -f -                              # Replace a pod based on the JSON passed into std

# Force replace, delete and then re-create the resource. Will cause a service outage.
kubectl replace --force -f ./pod.json

# Create a service for a replicated nginx, which serves on port 80 and connects to the containers on port 8000
kubectl expose rc nginx --port=80 --target-port=8000
```

kubectl Cheat Sheet
https://kubernetes.io/docs/reference/kubectl/cheatsheet/

Kubernetes: A Simple Example

Kube ▸ ! PowerFlowServer.yaml ▸ {} spec ▸ {} selector ▸ abc version

```yaml
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: powerflowserver-deployment
spec:
  replicas: 20
  template:
    metadata:
      labels:
        app: powerflowserver-app
        version: 16.01.0857
    spec:
      imagePullSecrets:
      - name: powerflowdeploymentauth
      containers:
      - name: powerflowserver
        image: griffink8stest.azurecr.io/powerflowserver:latest
        ports:
        - containerPort: 5000
        resources:
          limits:
            cpu: "1"
            memory: "200Mi"
          requests:
            cpu: "0.5"
            memory: "100Mi"
---
kind: Service
apiVersion: v1
metadata:
  name: powerflowserver-service
spec:
  type: ClusterIP
  selector:
    app: powerflowserver-app
    version: 16.01.0857
  ports:
  - protocol: TCP
    port: 2000
    targetPort: 5000
```

Deploy this image, please!

Keep 20 copies up at all times, no matter what happens!

Make sure it doesn't overwhelm my machine!

Light up a single DNS record and IP that routes traffic to the live instances of my container tagged with:

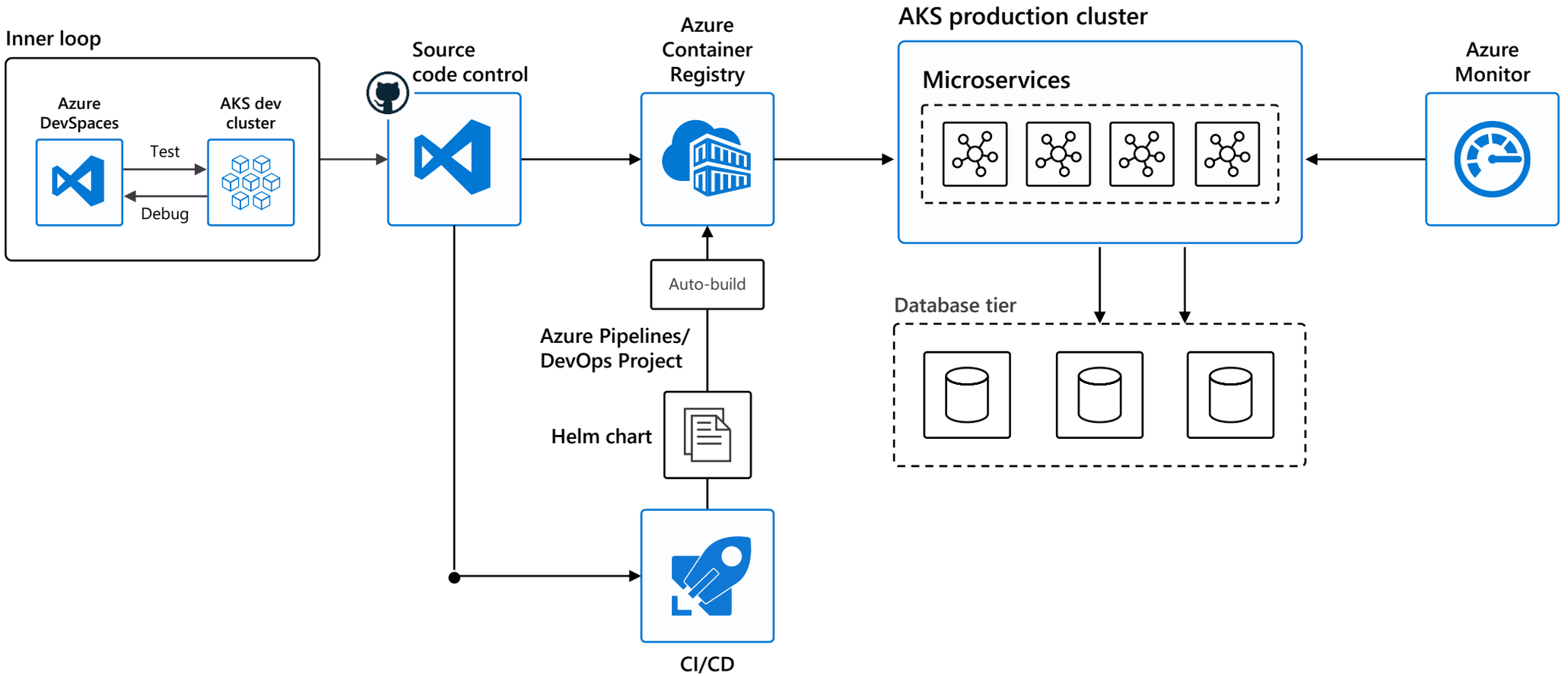
- app = powerflowserver-app
- version = 16.01.0857

WE continue 13.10

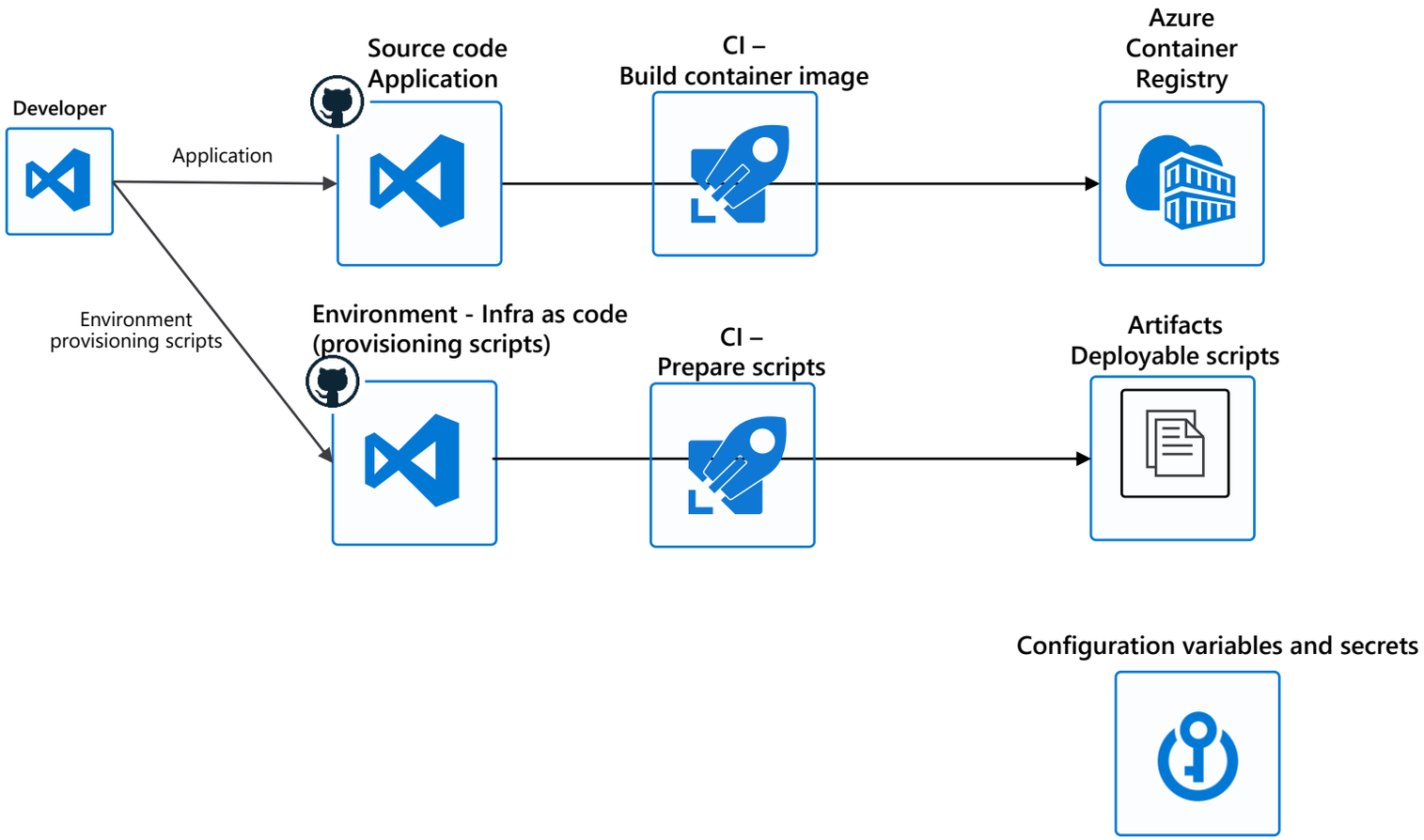
Microsoft Azure

DEMO

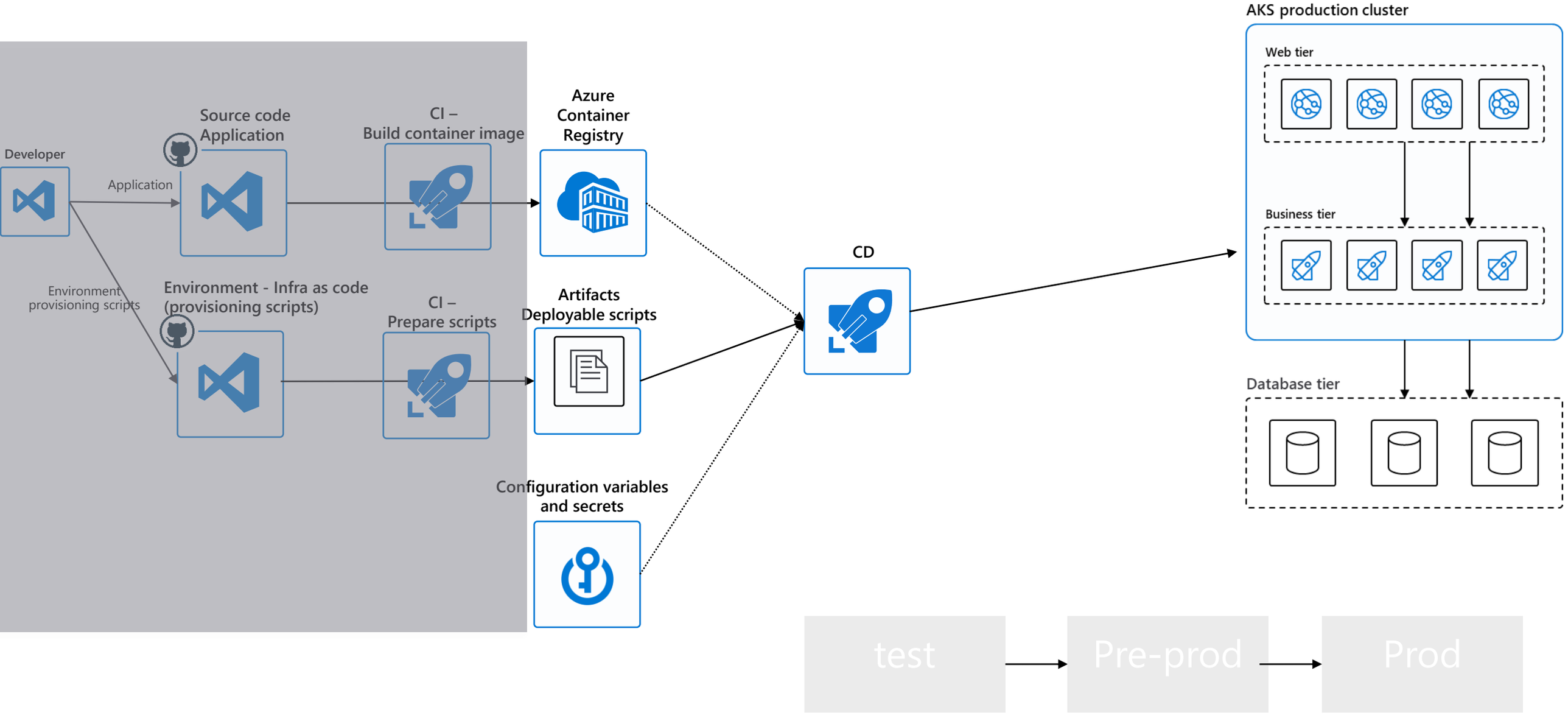
Integrated end-to-end Kubernetes experience



Integrated Kubernetes CI experience



Integrated Kubernetes CI/CD experience



Kubernetes is built and maintained by the community

Kubernetes collects **wisdom, code, and efforts** from hundreds of corporate contributors and thousands of individual contributors

150,000
commits

30,000
contributors

#1
GitHub project

Microsoft is part of this vibrant community and **leads in the associated committees** to help shape the future of Kubernetes and its ecosystem

CNCF
platinum member

CNCF
technical oversight committee

CNCF
governing board

Kubernetes
steering committee

Linux Foundation
board member

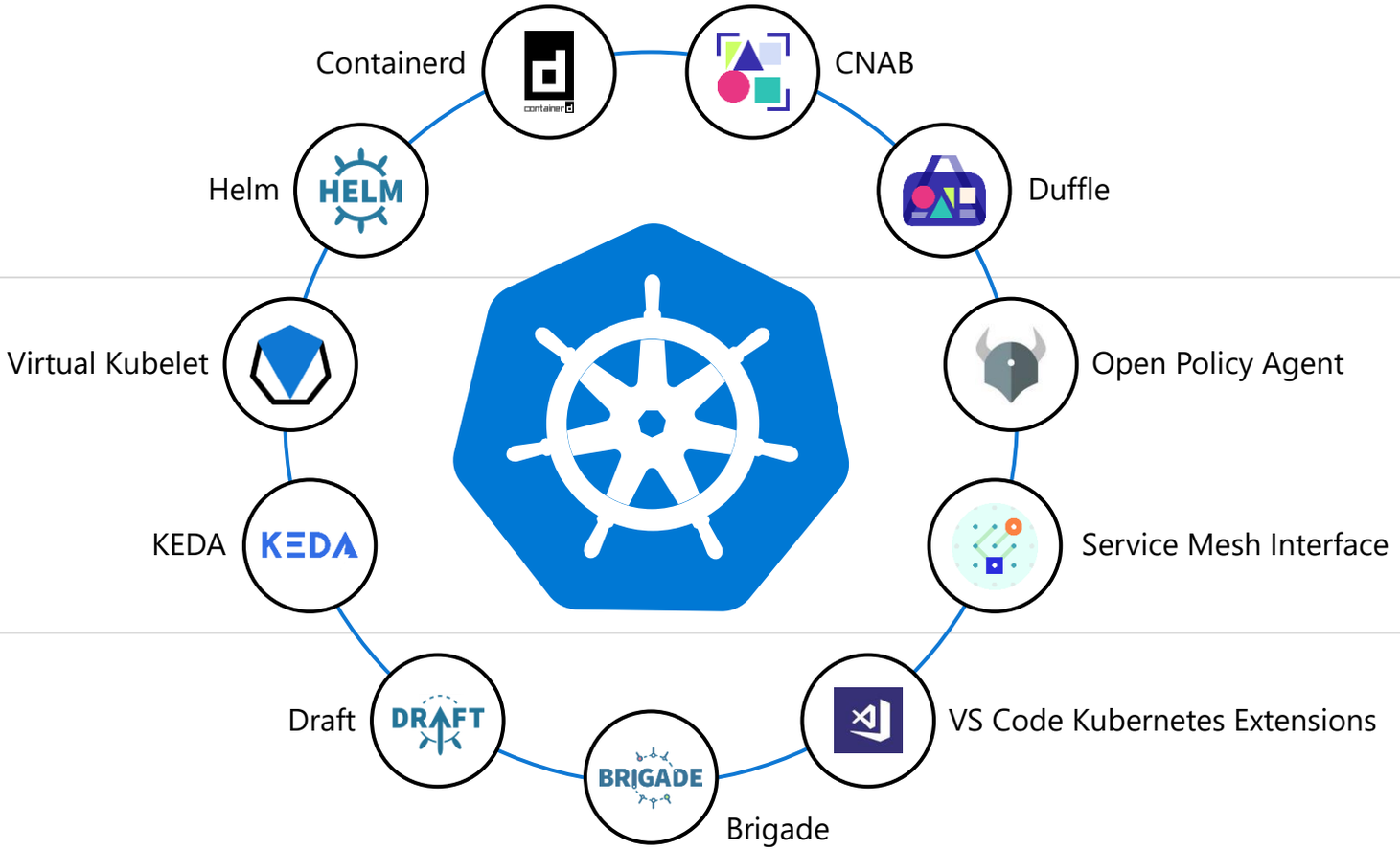
AKS is certified Kubernetes conformant, ensuring portability and interoperability of your container workloads

Microsoft contributions to the community

Packaging & distribution

Scalability & control

Kubernetes developer tooling



Microsoft contributions to the community

Top
code contributor to Windows support in Kubernetes

55,000
monthly downloads of Helm

68%
of Kubernetes users prefer Helm

1 of 3
top corporate contributors

3x
growth of employee contributors within three years

Created the Illustrated Children's Guide to Kubernetes, now part of CNCF

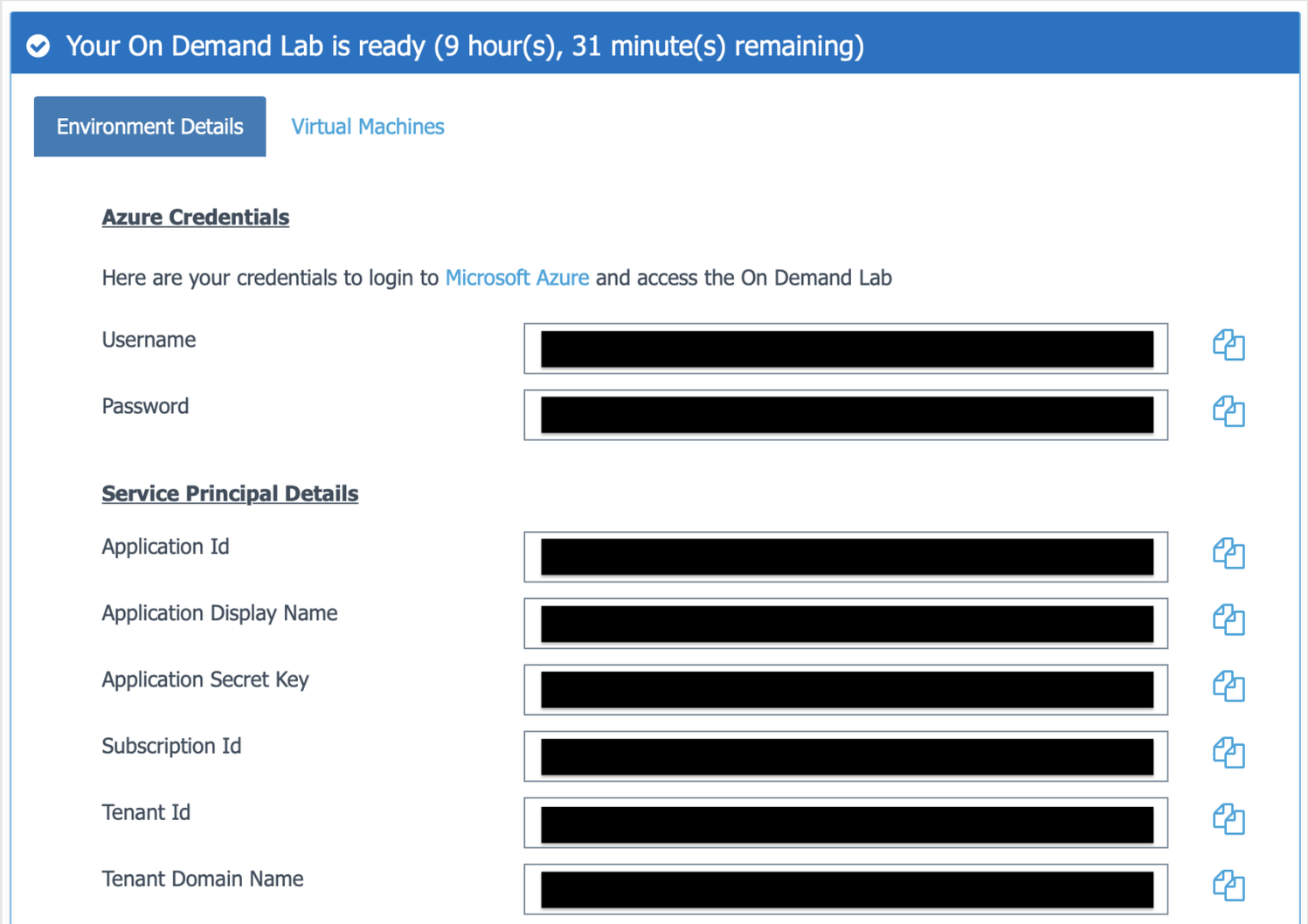
Work how you want with opensource tools and APIs

	Development	DevOps	Monitoring	Networking	Storage	Security
Take advantage of services and tools in the Kubernetes ecosystem	HELM DRAFT Virtual kubelet CNAB Service Mesh Interface	JFrog HashiCorp BRIGADE	Prometheus fluentd Grafana DATADOG	CNI Networking TIGERA	MAPR portworx	Twistlock aqua
Leverage 100+ turn-key Azure services	VS Code	Azure DevOps ARM Container Registry	Azure Monitor	Azure VNET	Azure Storage	Azure Policy AAD Key Vault

AKS LAB

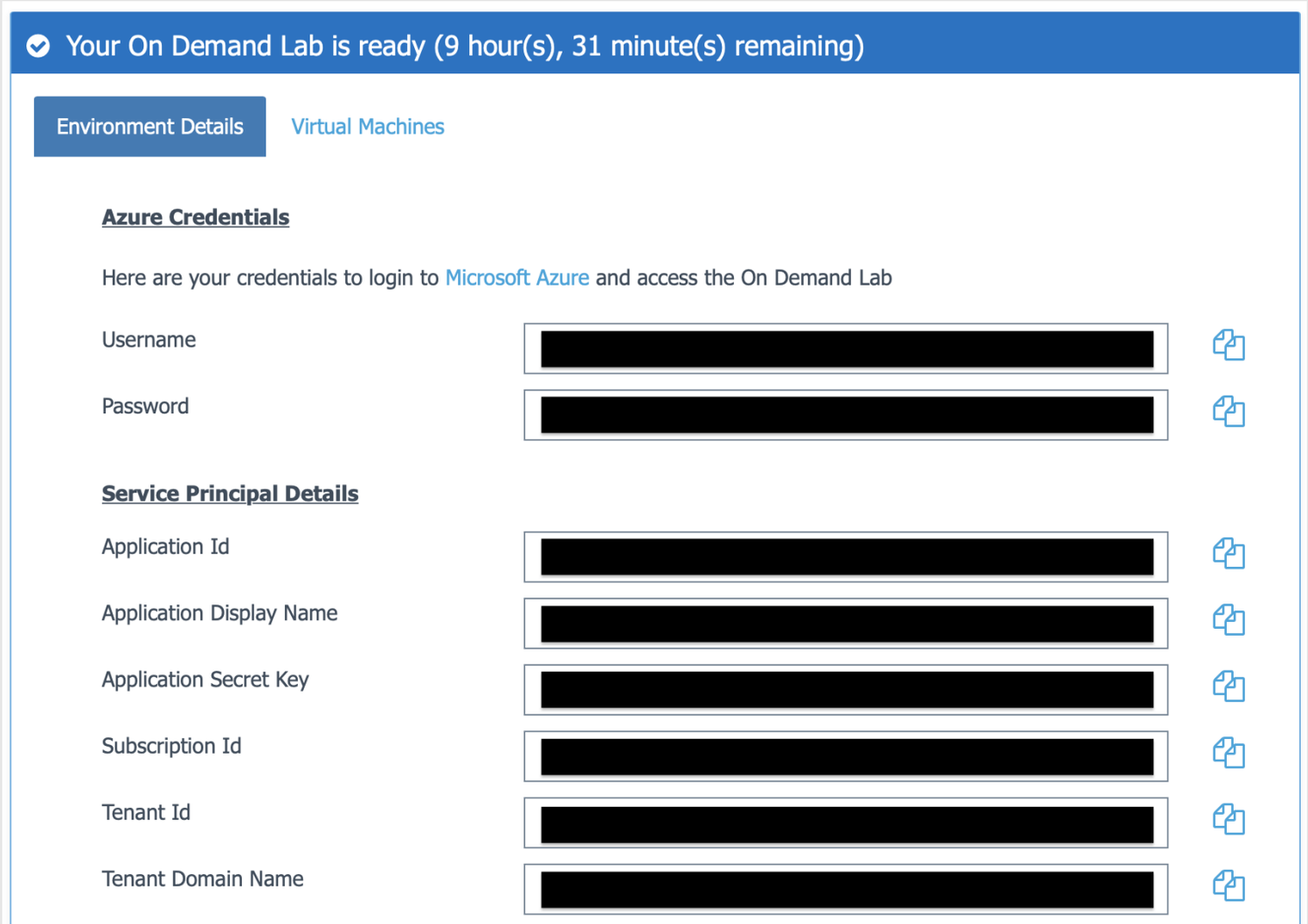
Step 1

- Do you have access to the environment?



Step 1

- Do you have access to the environment?

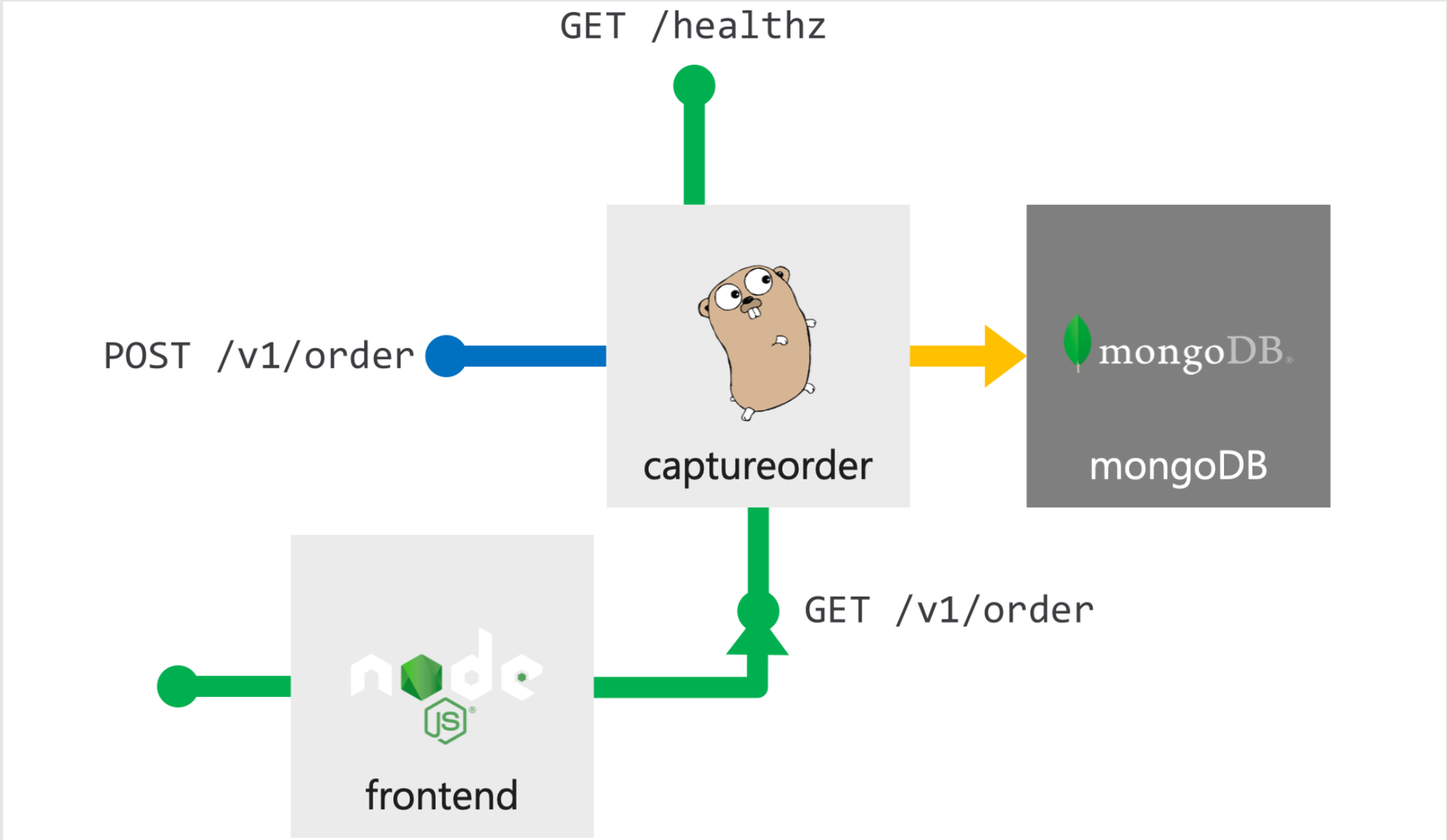


Next Steps

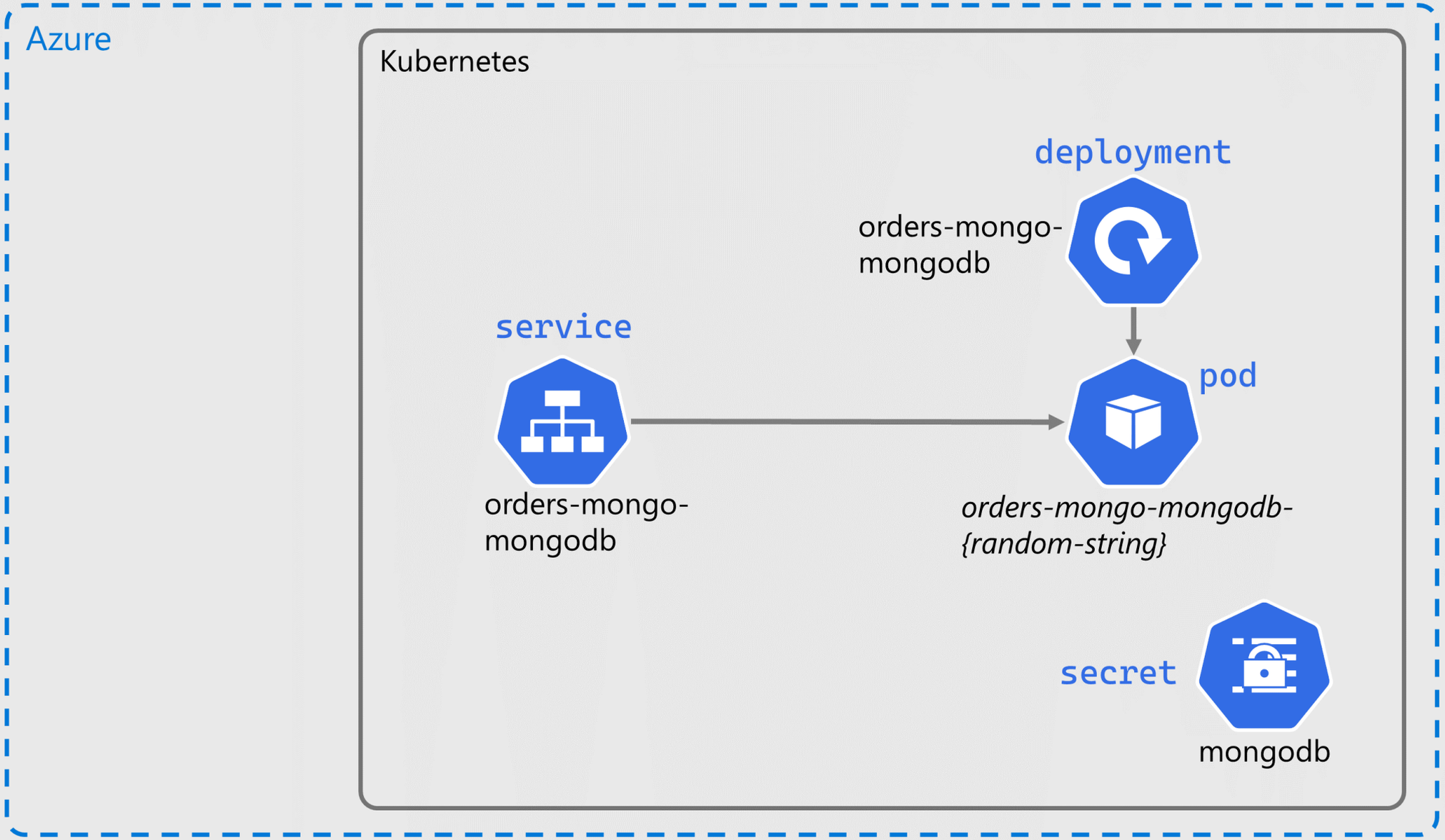
- Configure cloud shell
- Optional – configure VS Code and cloud shell on your laptop

Our application

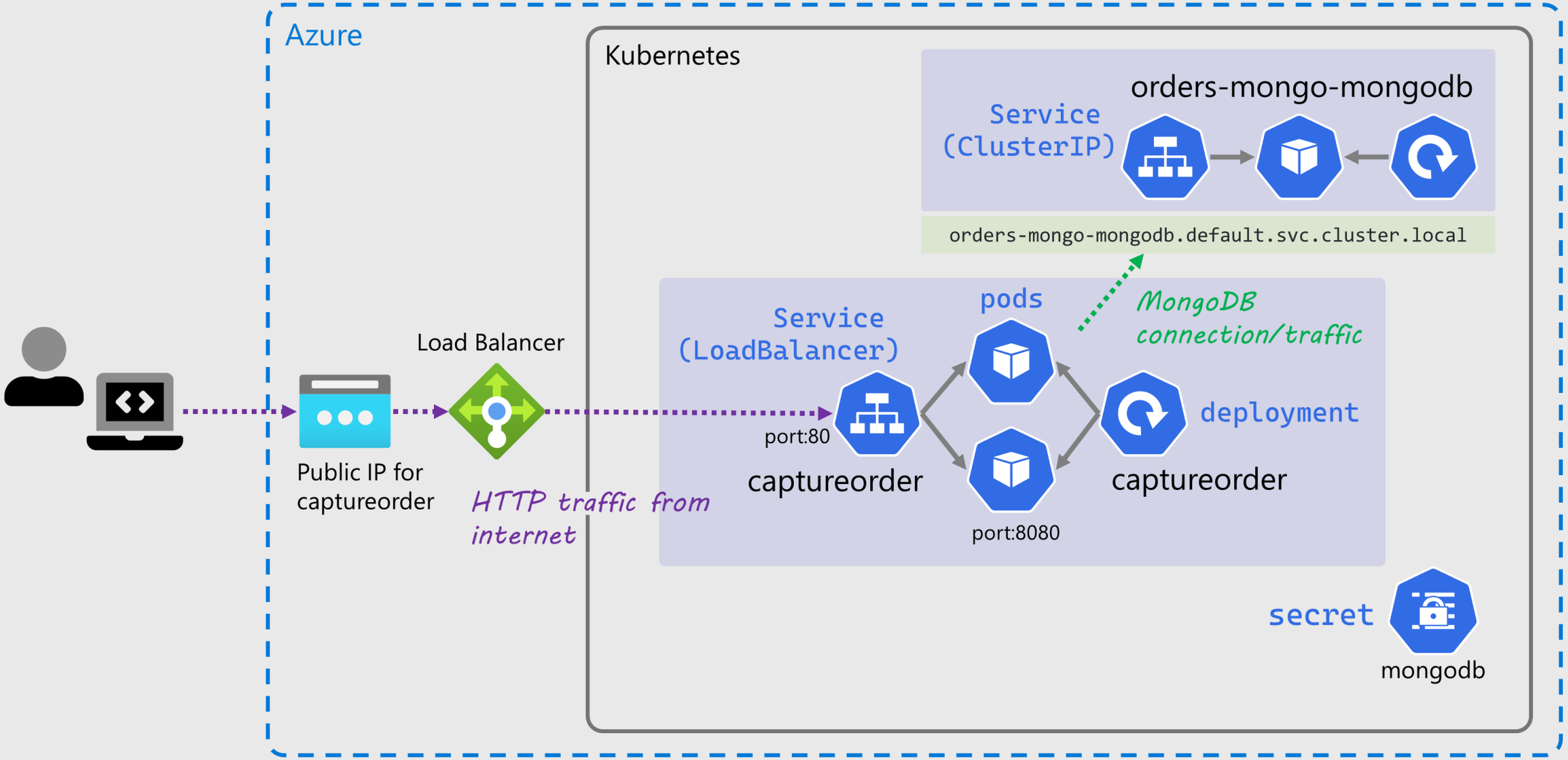
REGION: WESTEUROPE



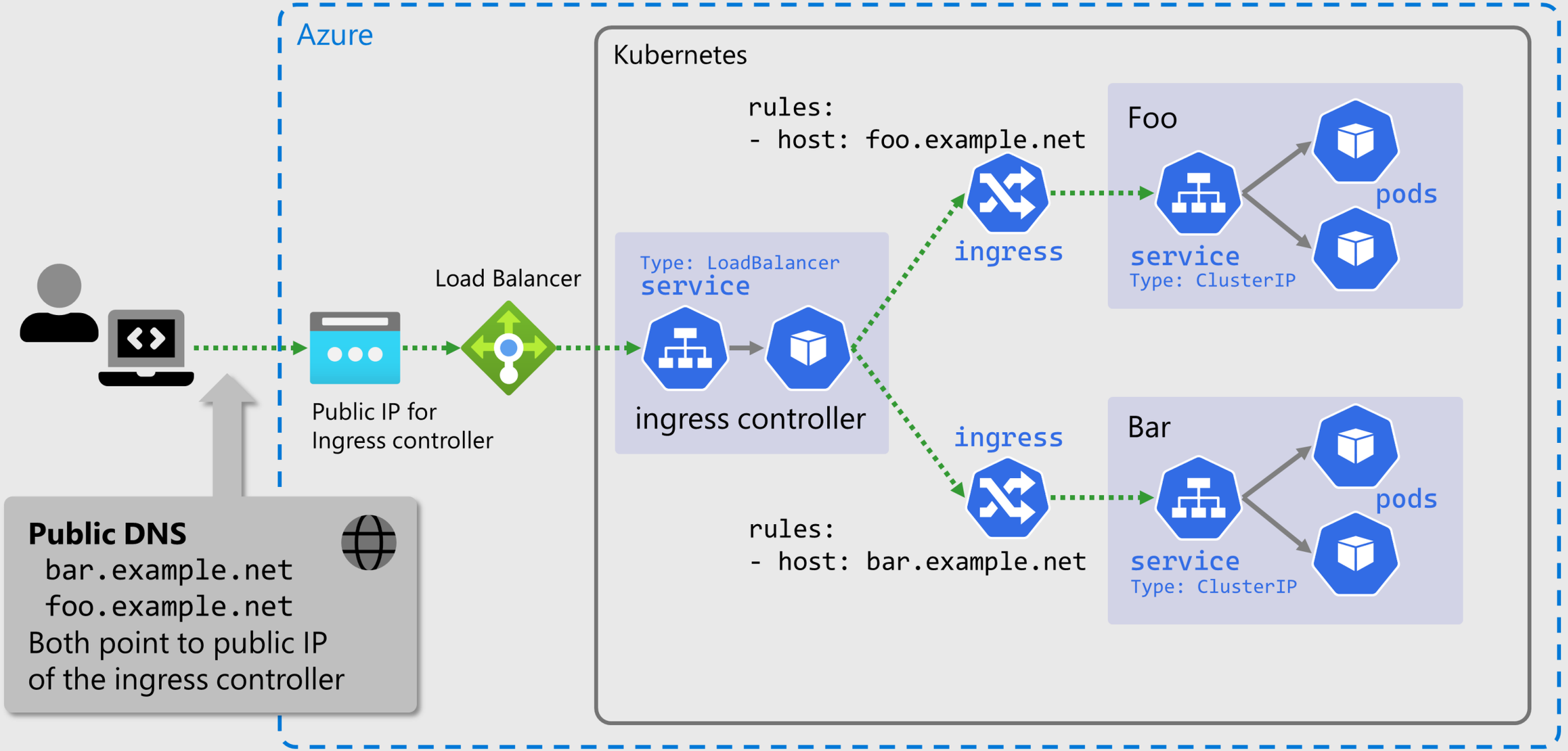
Deploy MongoDB – Architecture



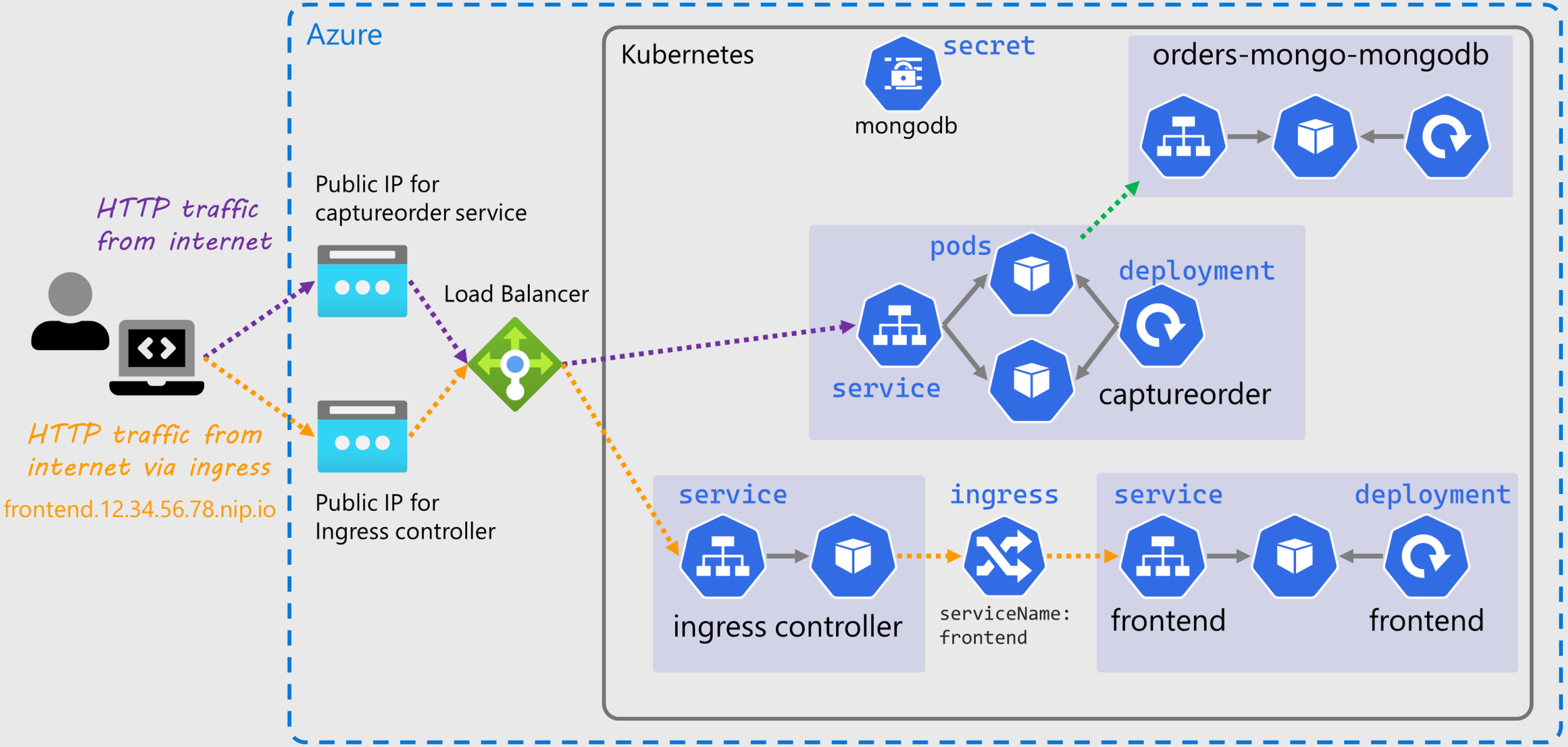
Deploy the Order Capture API – Architecture



Kubernetes Ingress – Simple Architecture



Deploy the Frontend Using Ingress – Architecture

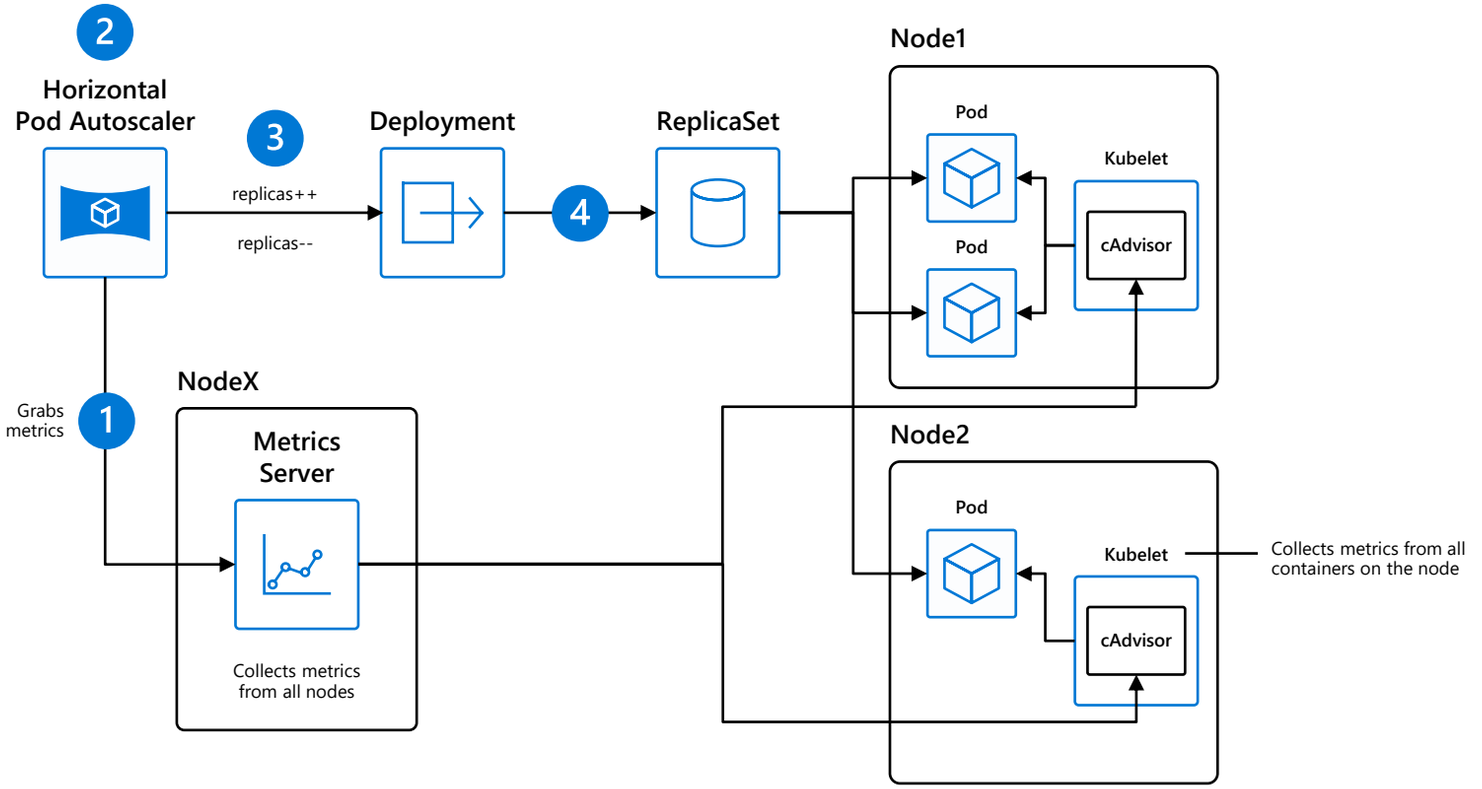


Additional slides

Horizontal Pod Autoscaler

The horizontal pod autoscaler (HPA) uses the Metrics Server in a Kubernetes cluster to monitor the resource demand of pods. If a service needs more resources, the number of pods is automatically increased to meet the demand.

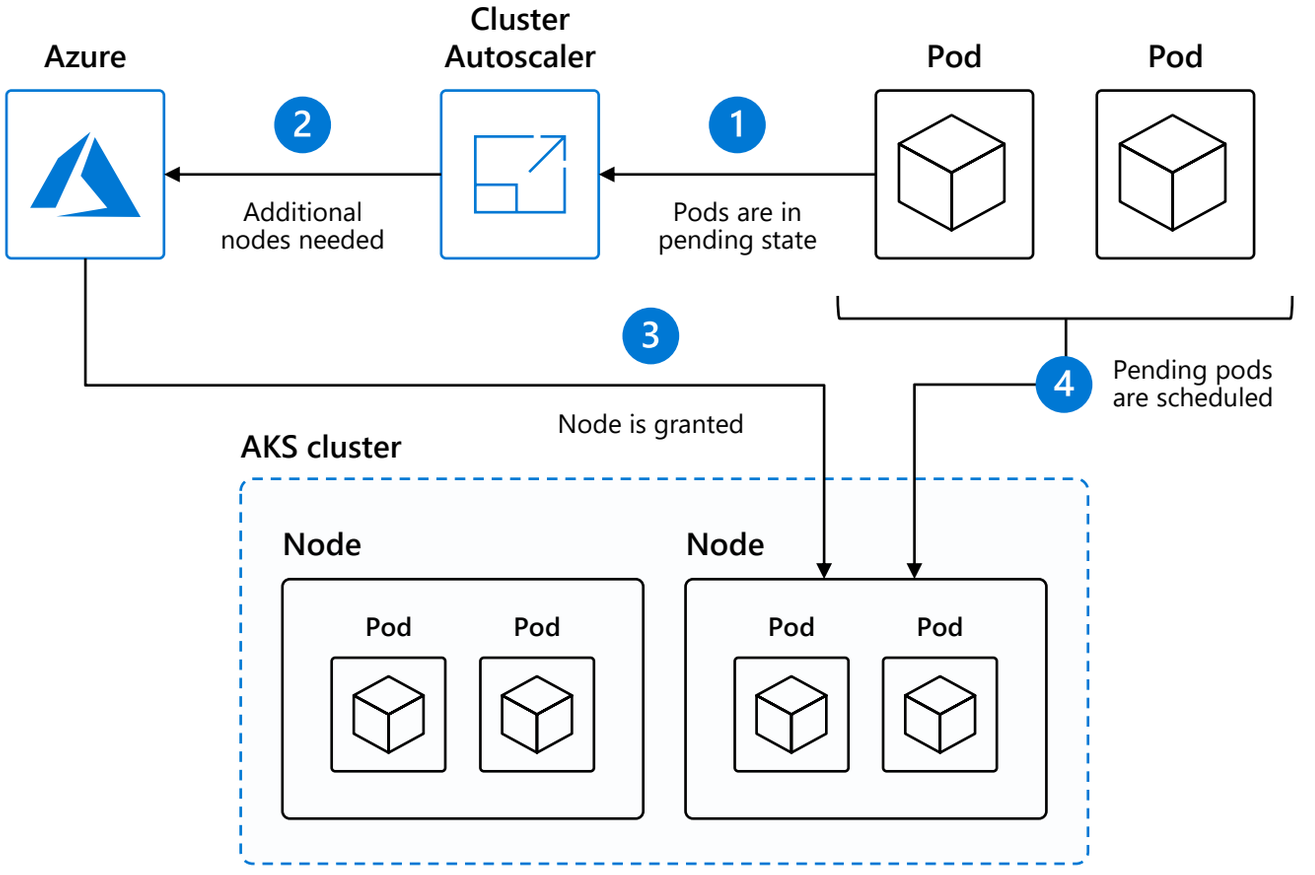
1. HPA obtains resource metrics and compares them to user-specified threshold
2. HPA evaluates whether user specified threshold is met or not
3. HPA increases/decreases the replicas based on the specified threshold
4. The Deployment controller adjusts the deployment based on increase/decrease in replicas



Cluster Autoscaler

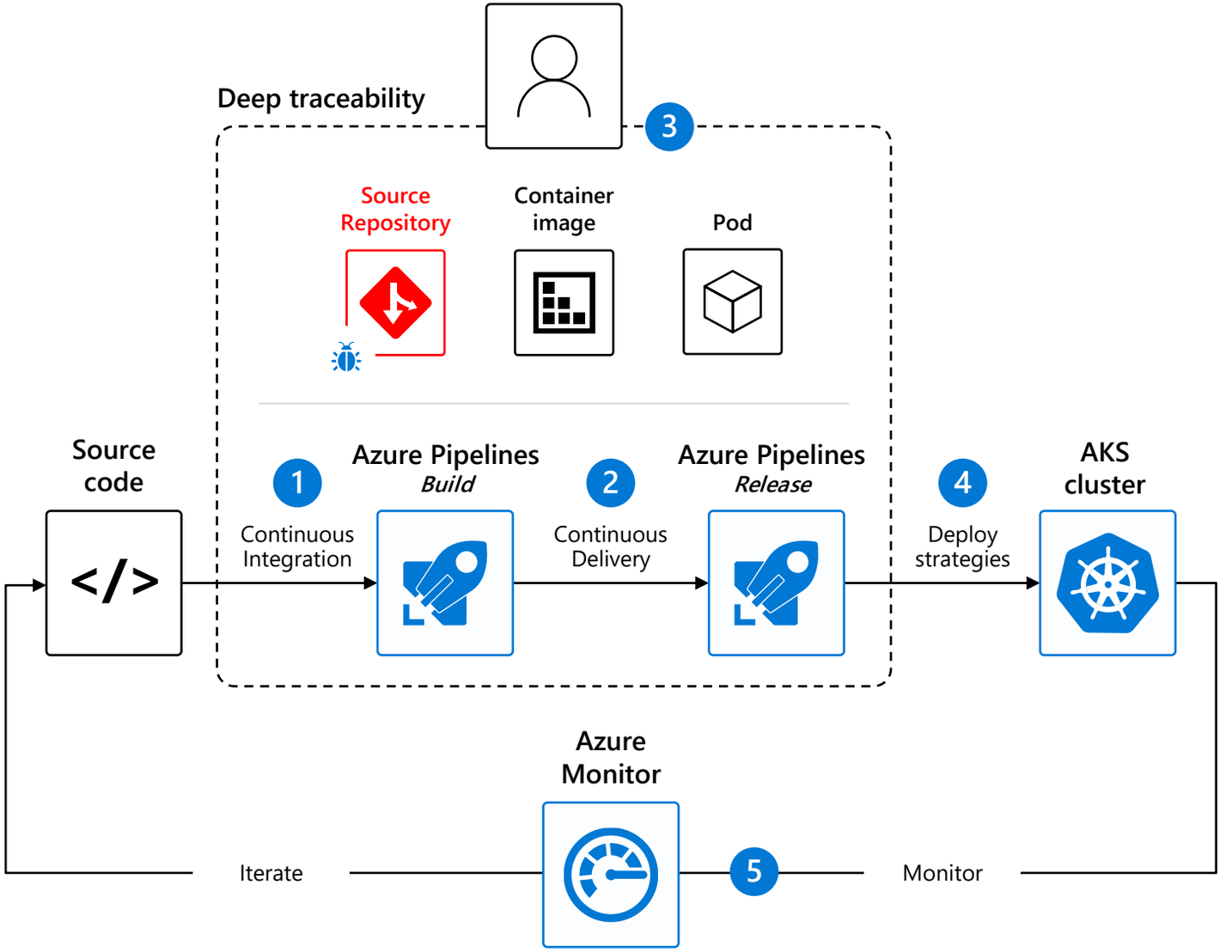
The cluster autoscaler watches for pods that can't be scheduled on nodes because of resource constraints. The cluster then automatically increases the number of nodes.

1. HPA obtains resource metrics and compares them to user-specified threshold
2. HPA evaluates whether user specified threshold is met or not
3. HPA increases/decreases the replicas based on the specified threshold
4. The Deployment controller adjusts the deployment based on increase/decrease in replicas



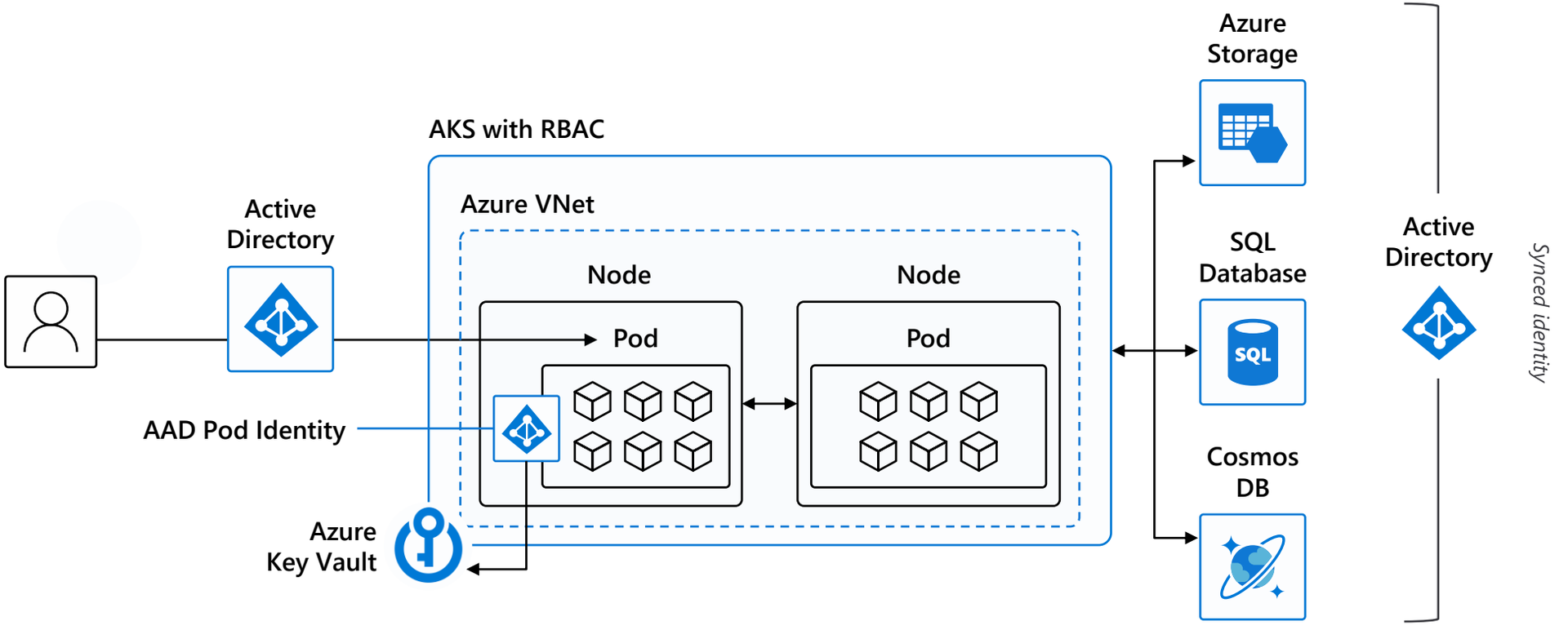
Azure Pipelines for AKS

1. As part of the CI, developers check in their code to a central repository; Azure pipelines automatically build application binaries, run unit test, and push container image into a registry
2. Developers then deploy the application to a testing environment and run integration test as part of the CD workflow
3. Developers can review which pod is running which container image, what source code is built into an image, and what tests are run against each image at any point of time
4. For production deployment, Azure Pipelines automatically execute pre-defined deployment strategy and progressively roll out application to an AKS cluster
5. Enable app telemetry, container health monitoring, and real-time log analytics; insights used to address issues and feed into next sprint plans



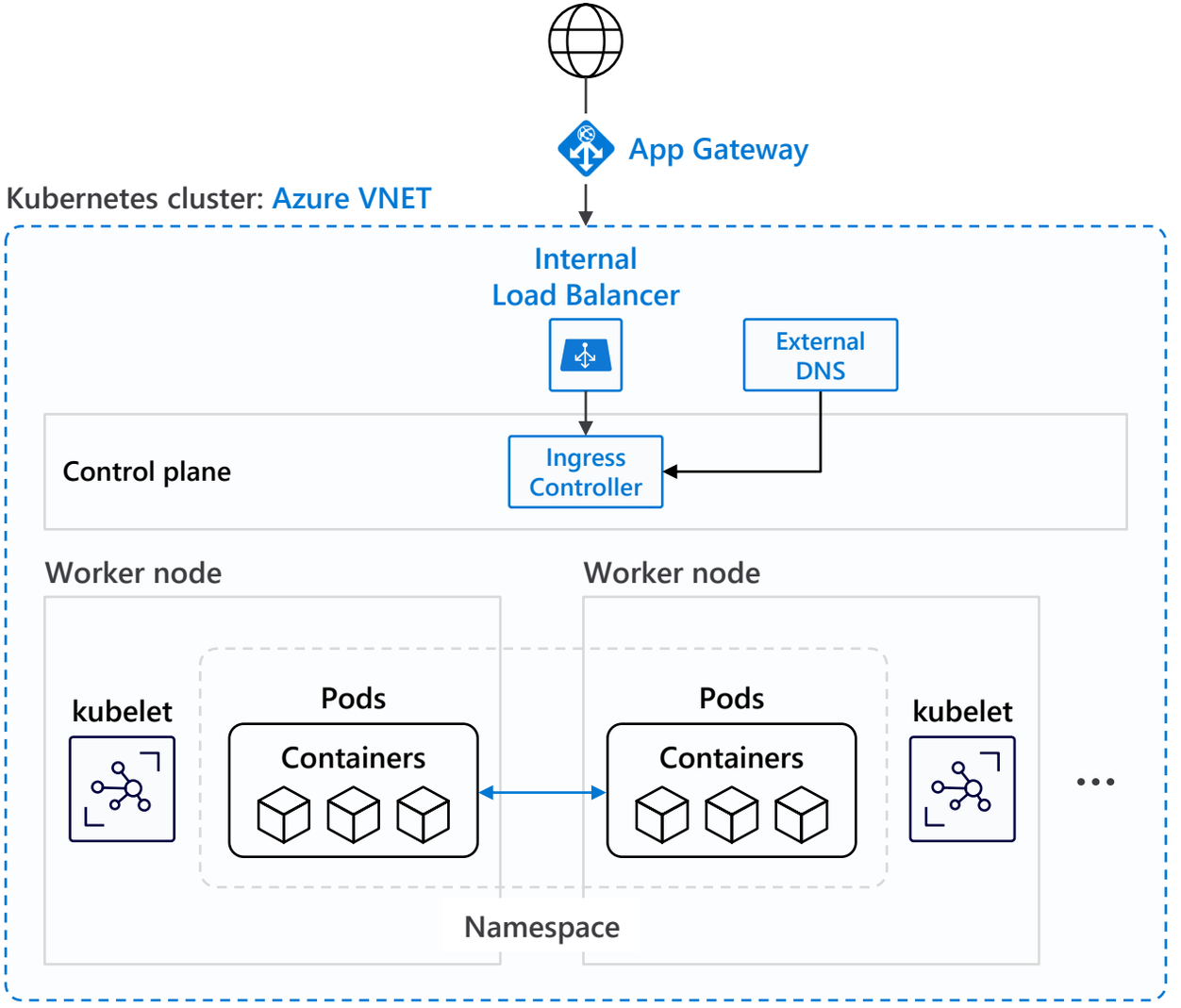
Identity

Use familiar tools like AAD for fine-grained identity and access control to Kubernetes resources from cluster to containers



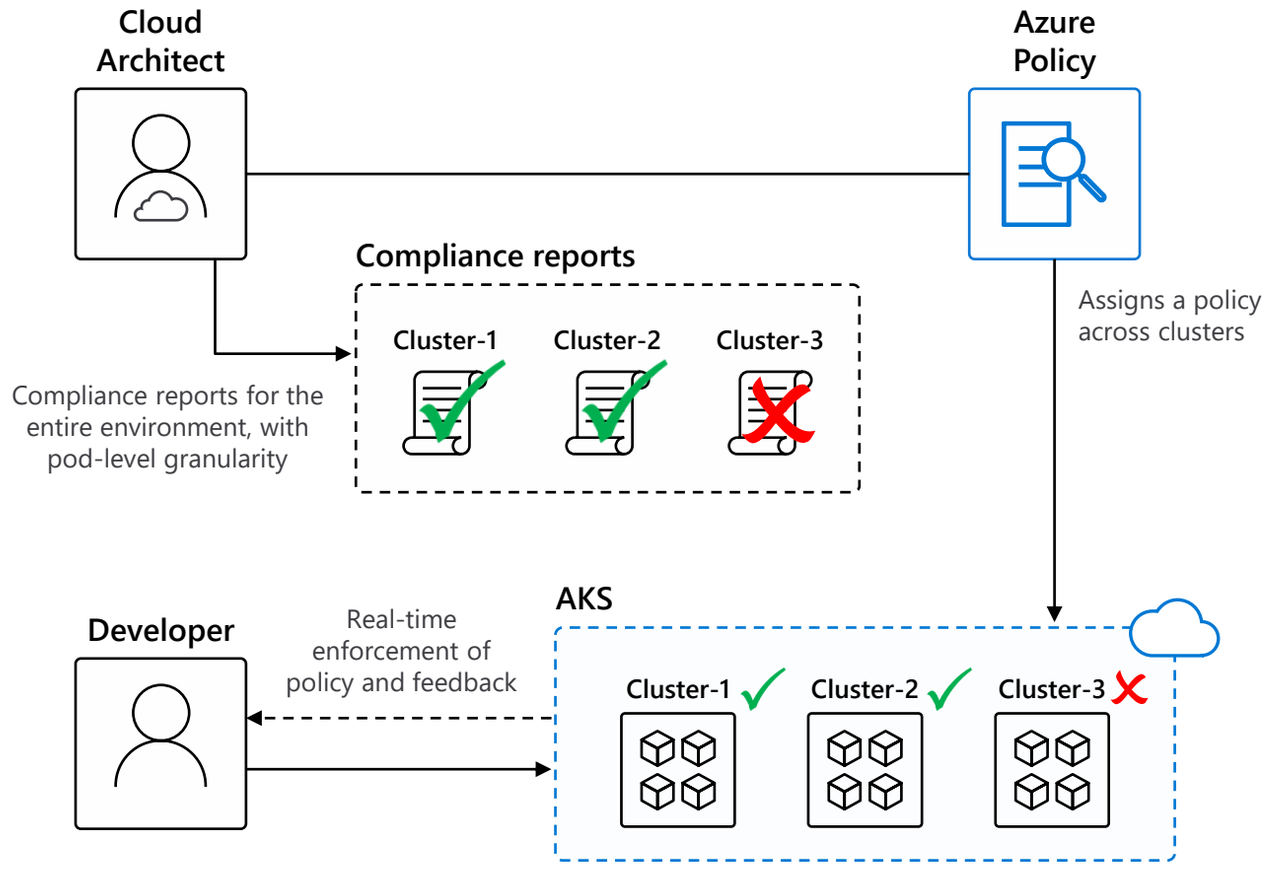
Networking

Secure your Kubernetes workloads with virtual network and policy-driven communication paths between resources



Governance

Dynamically enforce guardrails defined in Azure Policy across multiple clusters—nodes, pods, and even container images can be tracked and validated at the time of deployment or as part of CI/CD workflows



Run anything, anywhere

Your choice of…

Container

Windows Linux

Region

20+ regions worldwide

Environment

Public cloud

IoT Edge

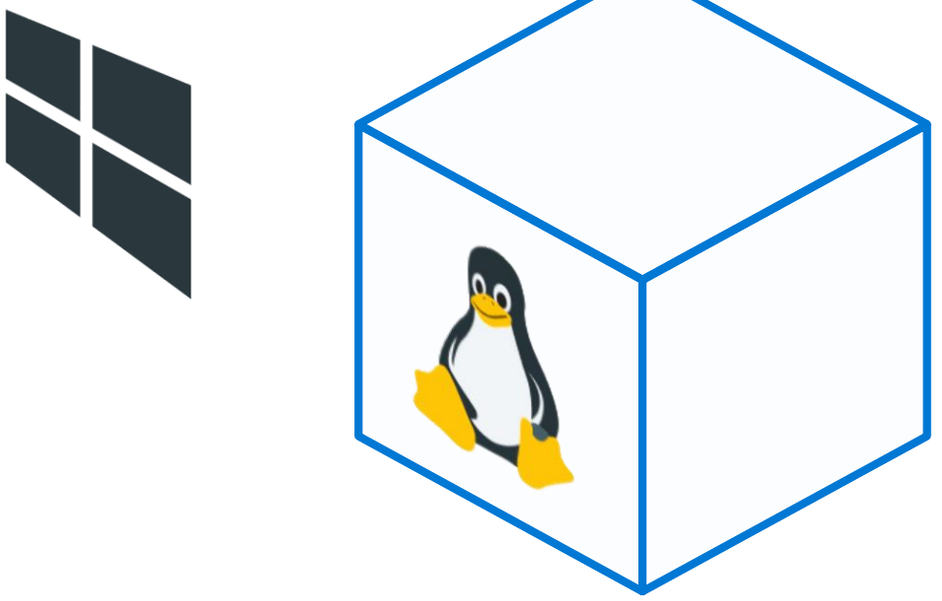
Azure Government *(coming soon)*

Azure Stack

Azure Kubernetes Service (AKS) support for Windows Server Containers

Now you can get the best of managed Kubernetes for all your workloads whether they're in Windows, Linux, or both

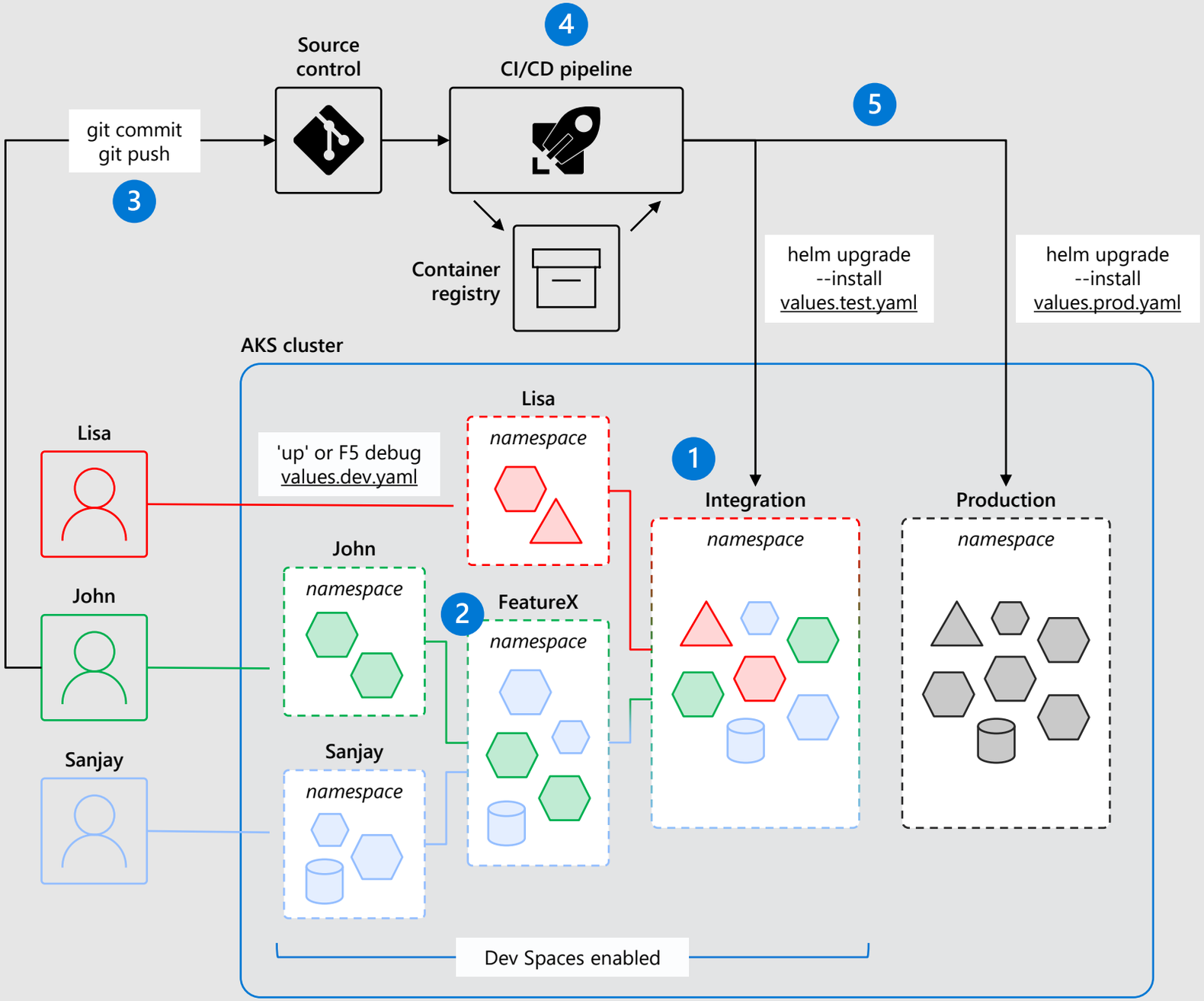
- Lift and shift Windows applications to run on AKS
- Seamlessly manage Windows and Linux applications through a single unified API
- Mix Windows and Linux applications in the same Kubernetes cluster—with consistent monitoring experience and deployment pipelines



Dev Spaces

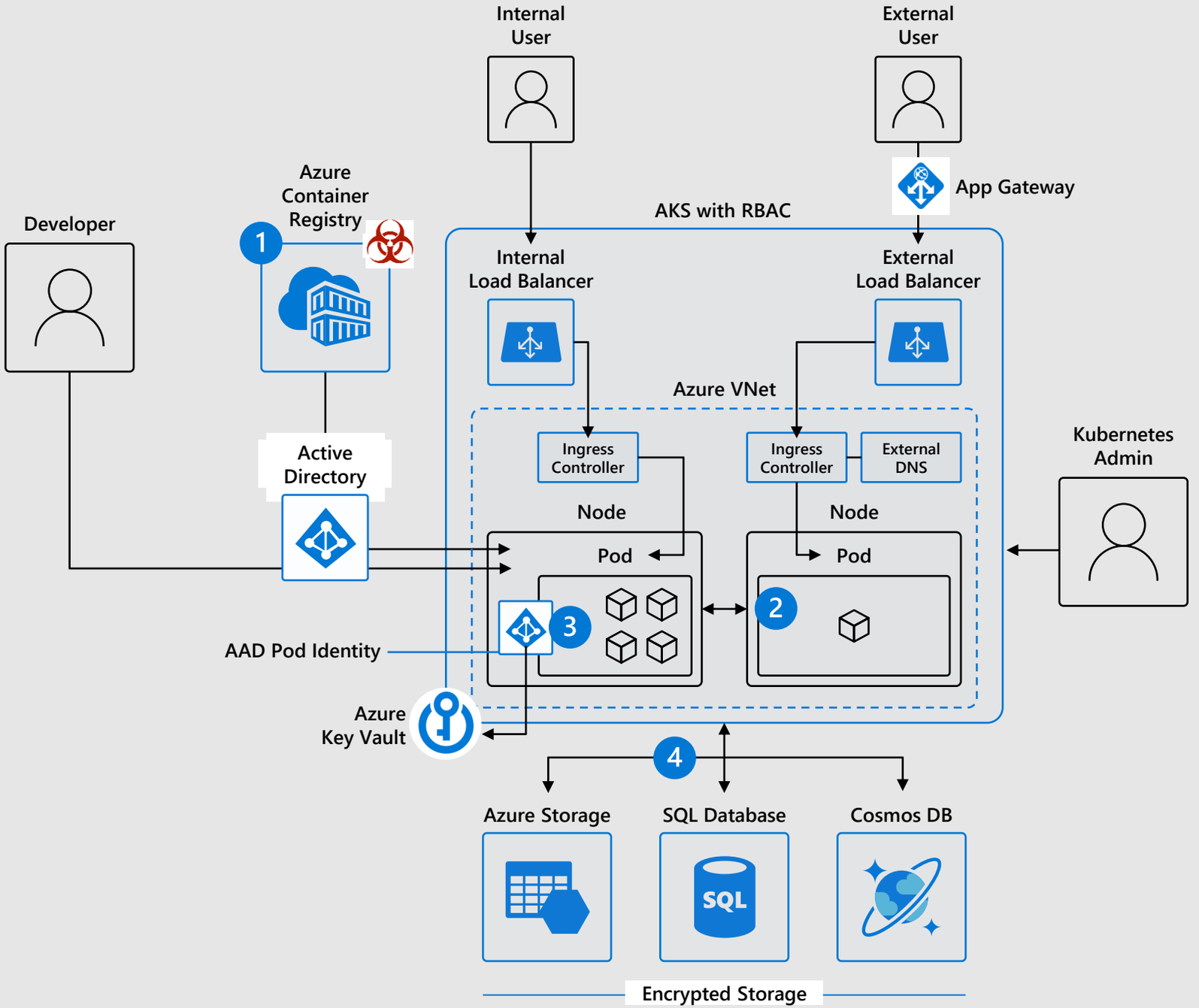
1. The “Integration” dev space is running a full baseline version of the entire application
2. John and Sanjay are collaborating on FeatureX; it is setup as a dev space and running all the modified services required to implement a feature
3. Code is committed to the master source control
4. A CI/CD pipeline can be triggered to deploy into “Integration,” which updates the team's baseline
5. The same Helm assets used during development are used in later environments by the CD system

*Dev Spaces is enabled per Kubernetes namespaces and can be defined as anything. Any namespace in which Dev Spaces is NOT enabled runs *unaffected*.*

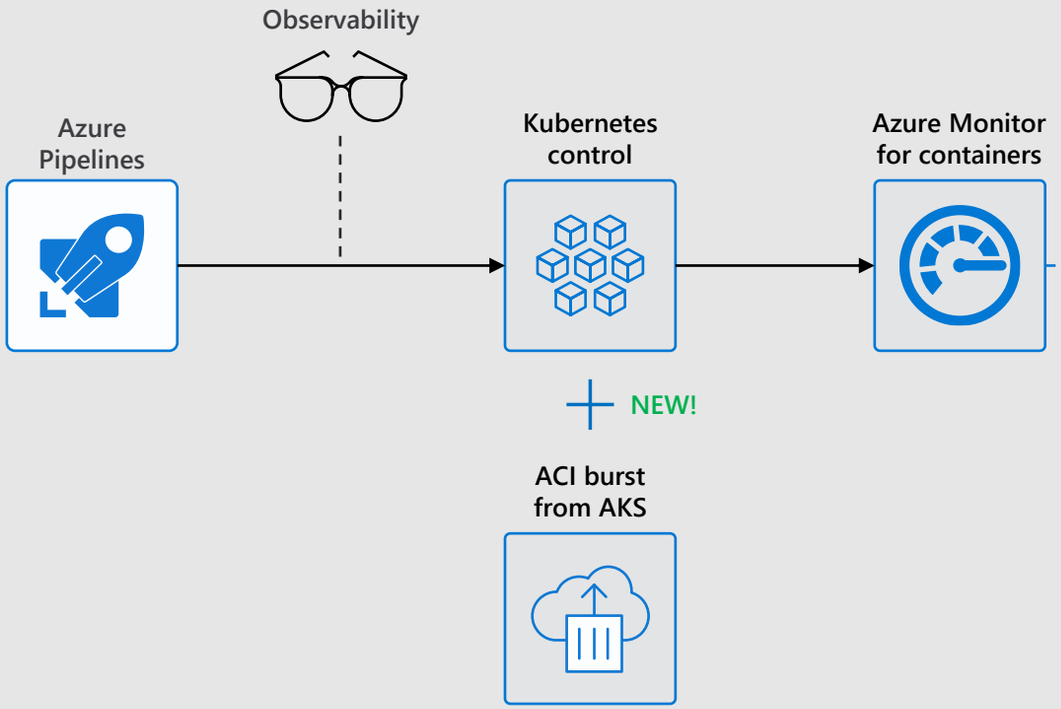


Security overview

1. Image and container level security
   - AAD authenticated Container registry access
   - ACR image scanning and content trust for image validation
2. Node and cluster level security
   - Automatic security patching nightly
   - Nodes deployed in private virtual network subnet w/o public addresses
   - Network policy to secure communication paths between namespaces (and nodes)
   - Pod Security Policies
   - K8s RBAC and AAD for authentication
3. Pod level security
   - Pod level control using AAD Pod Identity
   - Pod Security Context
4. Workload level security
   - Azure Role-based Access Control (RBAC) & security policy groups
   - Secure access to resources & services (e.g. Azure Key Vault) via Pod Identity
   - Storage Encryption
   - App Gateway with WAF to protect against threats and intrusions



Azure Monitor for containers



Visualization
Visualize overall health and performance from clusters to containers with drill downs and filters

Insights
Provide insights with multi-cluster health roll up view

Monitor & Analyze
Monitor and analyze Kubernetes and container deployment performance, events, health, and logs

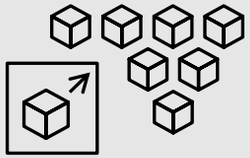
Response
Native alerting with integration to issue managements and ITSM tools

Observability
Observe live container logs on container deployment status

Top scenarios

Top scenarios for Kubernetes on Azure

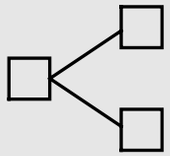
Lift and shift to containers



Cost saving

without refactoring your app

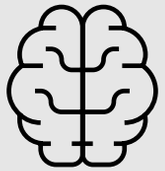
Microservices



Agility

Faster application development

Machine learning



Performance

Low latency processing

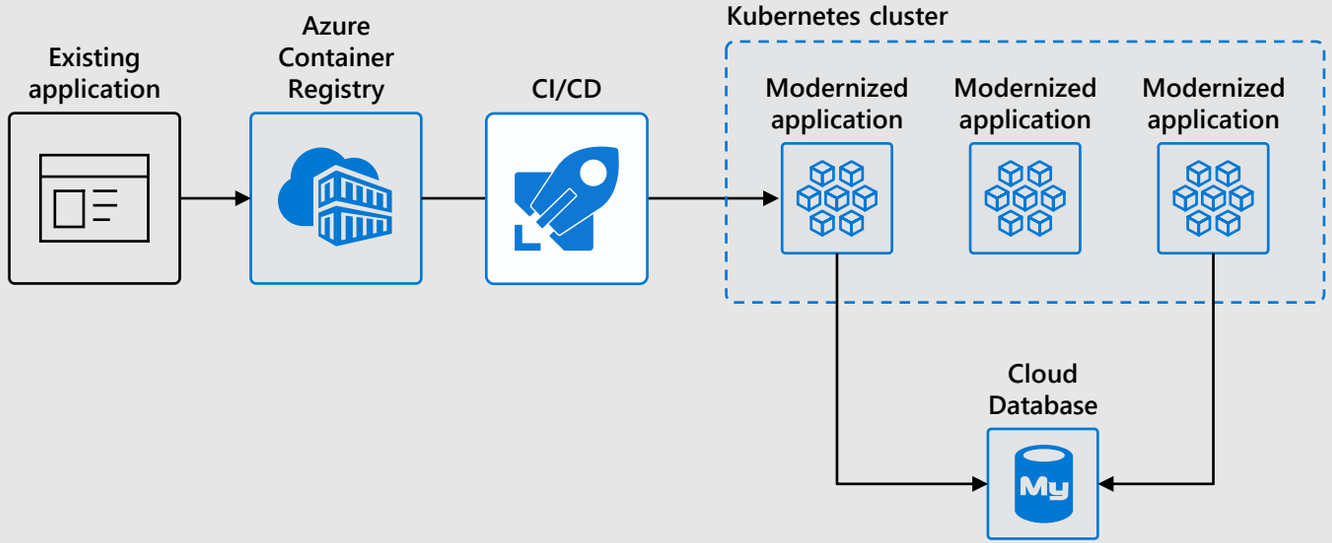
IoT

Portability

Build once, run anywhere

App modernization without code changes

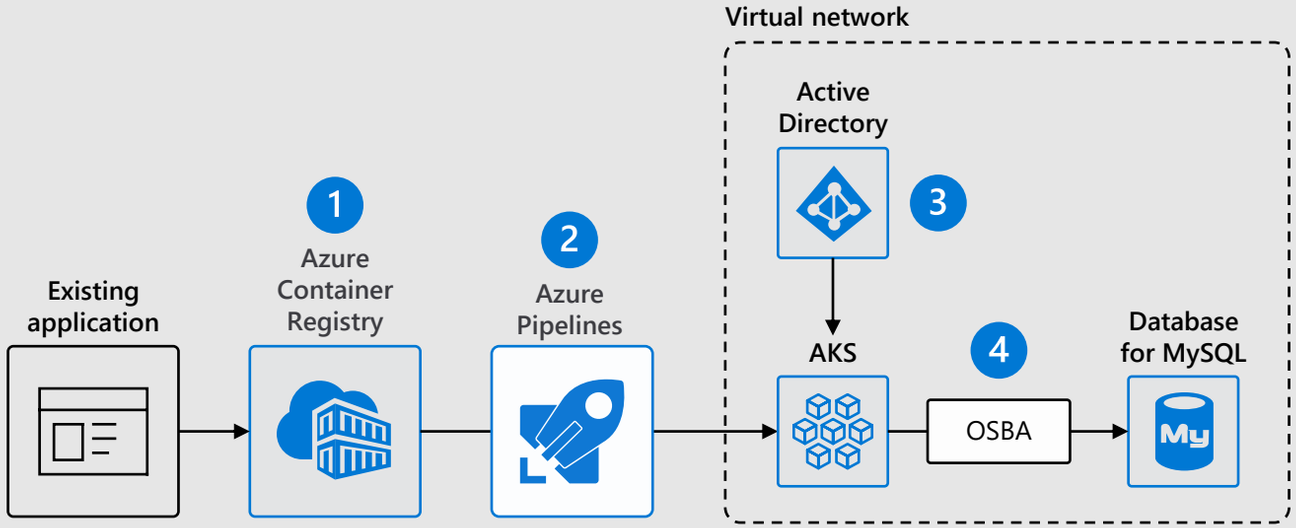
- Speed application deployments by using container technology
- Defend against infrastructure failures with container orchestration
- Increase agility with continuous integration and continuous delivery



App modernization without code changes

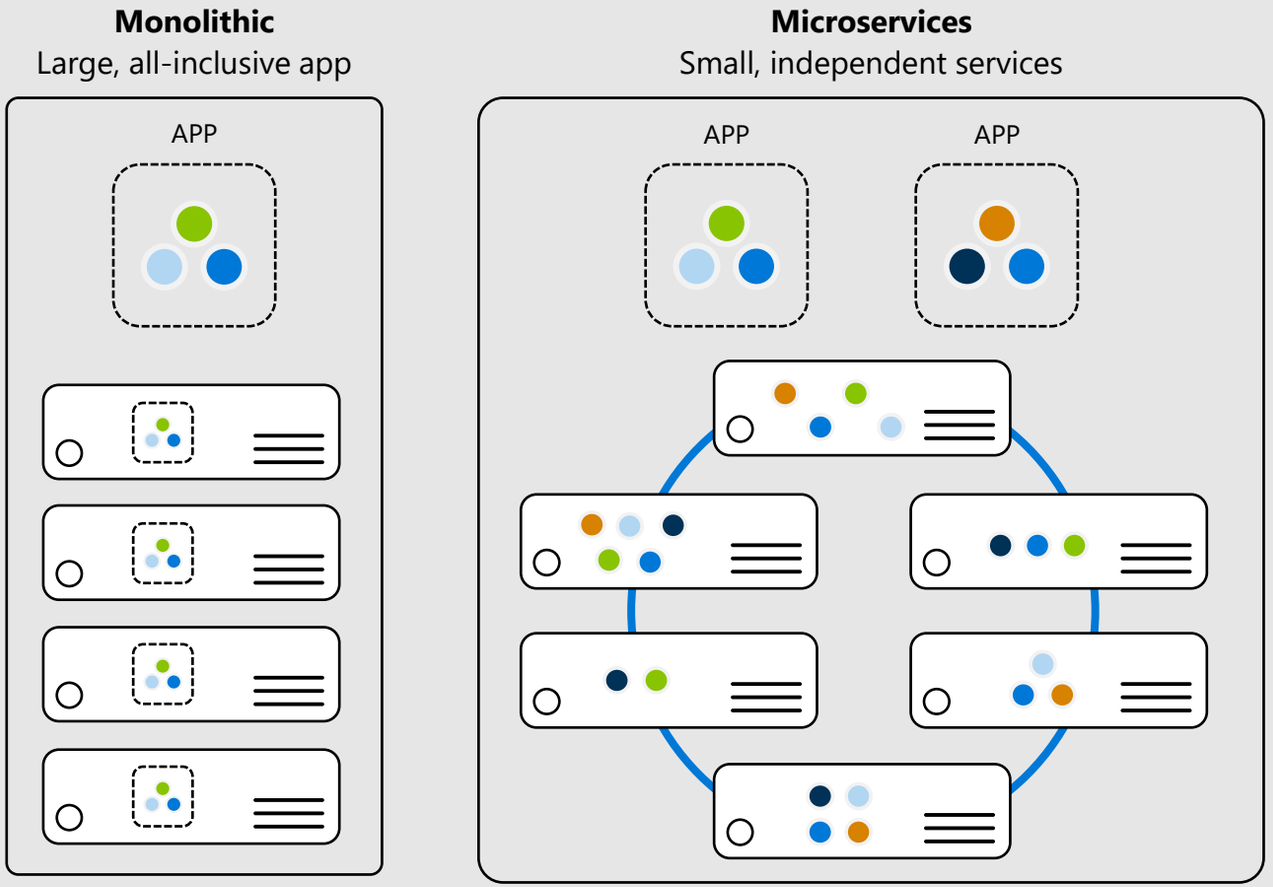
Capabilities

1. Use **Azure Container Registry** to store container images and Helm charts for your modernized applications, replicated globally for low latency image serving
2. Integrate AKS with **Azure Pipelines** or other Kubernetes ecosystem tooling to enable continuous integration/continuous delivery (CI/CD)
3. Enhance security with **Azure Active Directory** and RBAC to control access to AKS resources
4. Easily access to SLA-backed Azure Services such as Azure Database for MySQL using **Open Service Broker for Azure** (OSBA)



Microservices: for faster app development

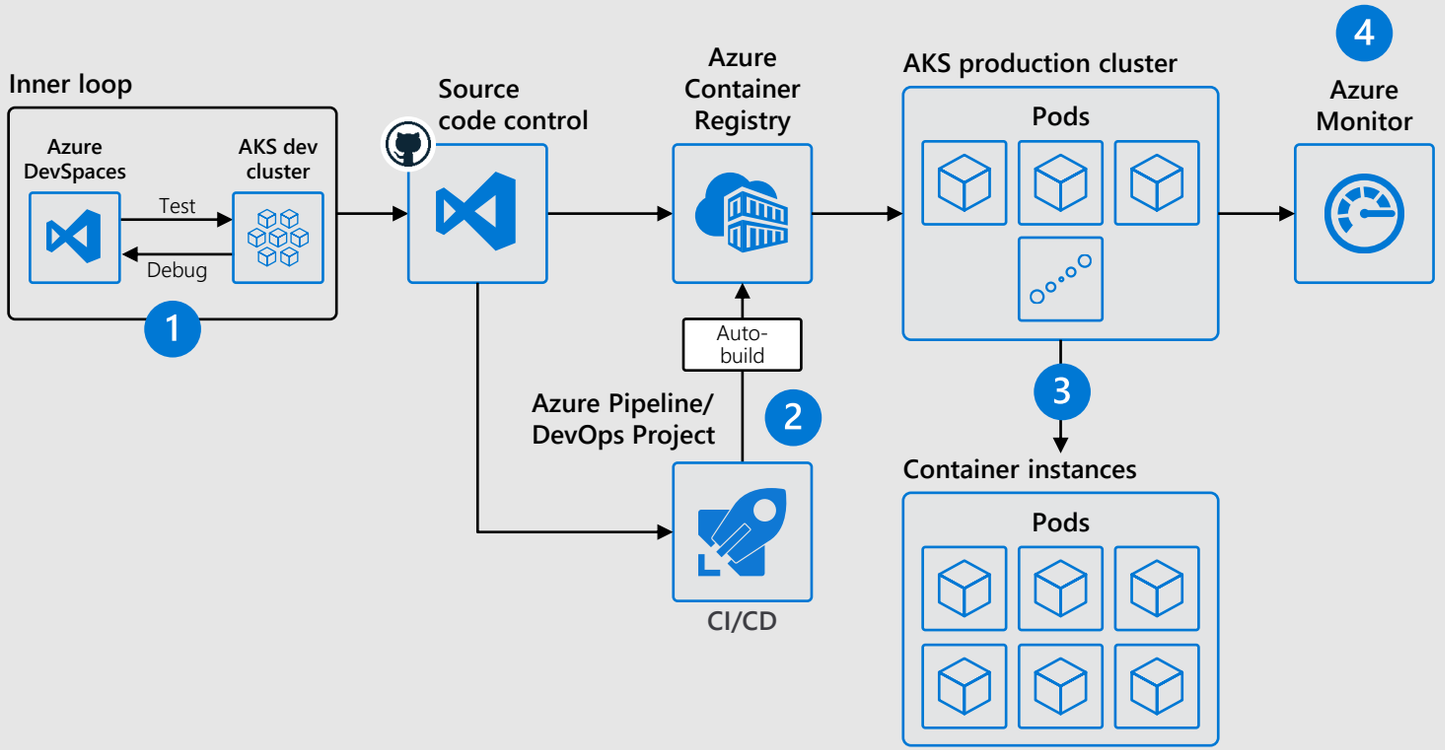
- Independent deployments
- Improved scale and resource utilization per service
- Smaller, focused teams



Microservices: for faster app development

Capabilities

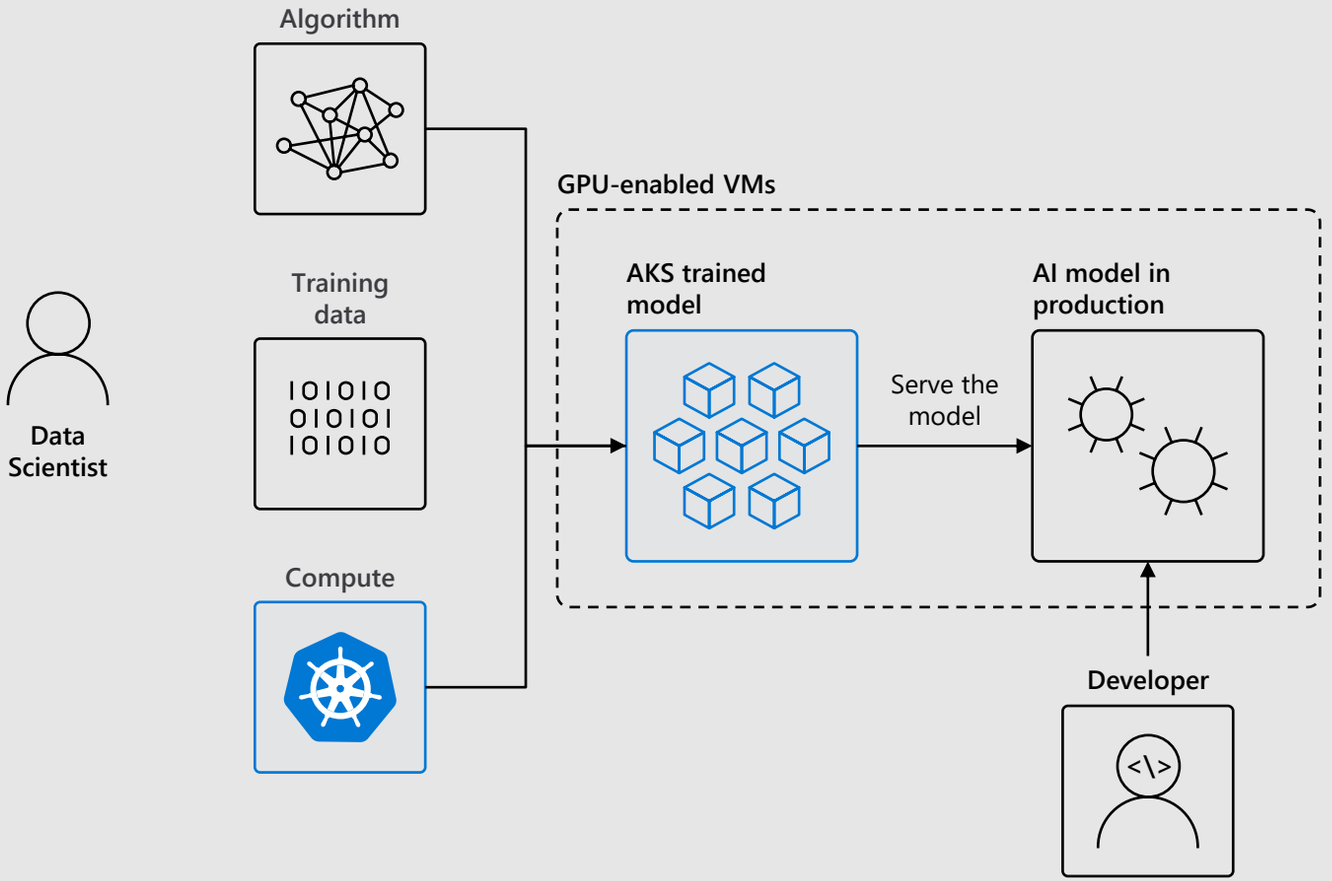
1. Use **Azure Dev Spaces** to iteratively develop, test, and debug microservices targeted for AKS clusters.
2. A**zure DevOps** has native integration with Helm and helps simplifying continuous integration/continuous delivery (CI/CD)
3. **Virtual node**—a Virtual Kubelet implementation—allows fast scaling of services for unpredictable traffic.
4. **Azure Monitor** provides a single pane of glass for monitoring over app telemetry, cluster-to-container level health analytics.



https://github.com/Microsoft/SmartHotel360-AKS-DevSpaces-Demo

Data scientist in a box

- Quick deployment and high availability
- Low latency data processing
- Consistent environment across test, control and production

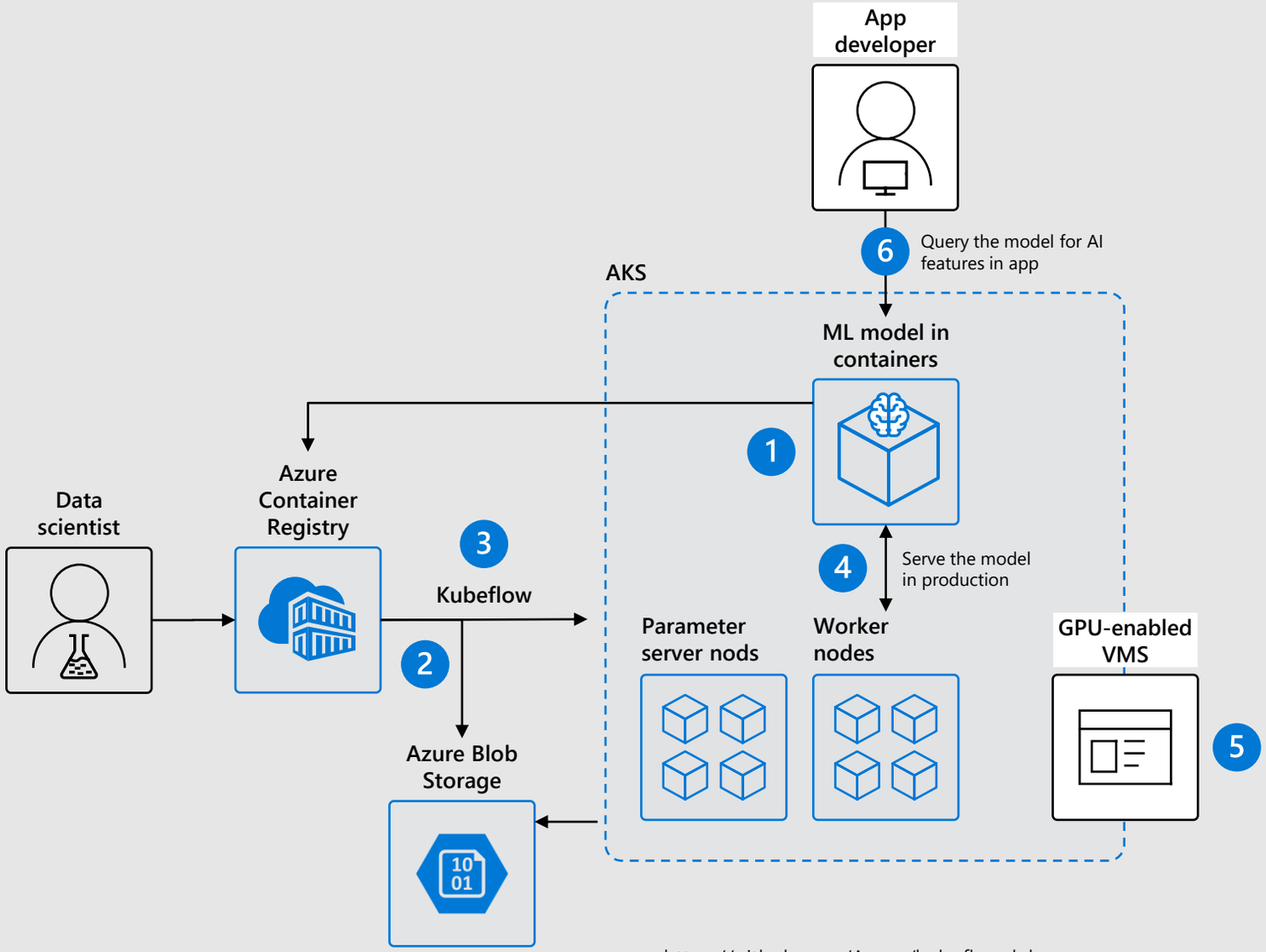


https://github.com/Azure/kubeflow-labs

Data scientist in a box

Capabilities

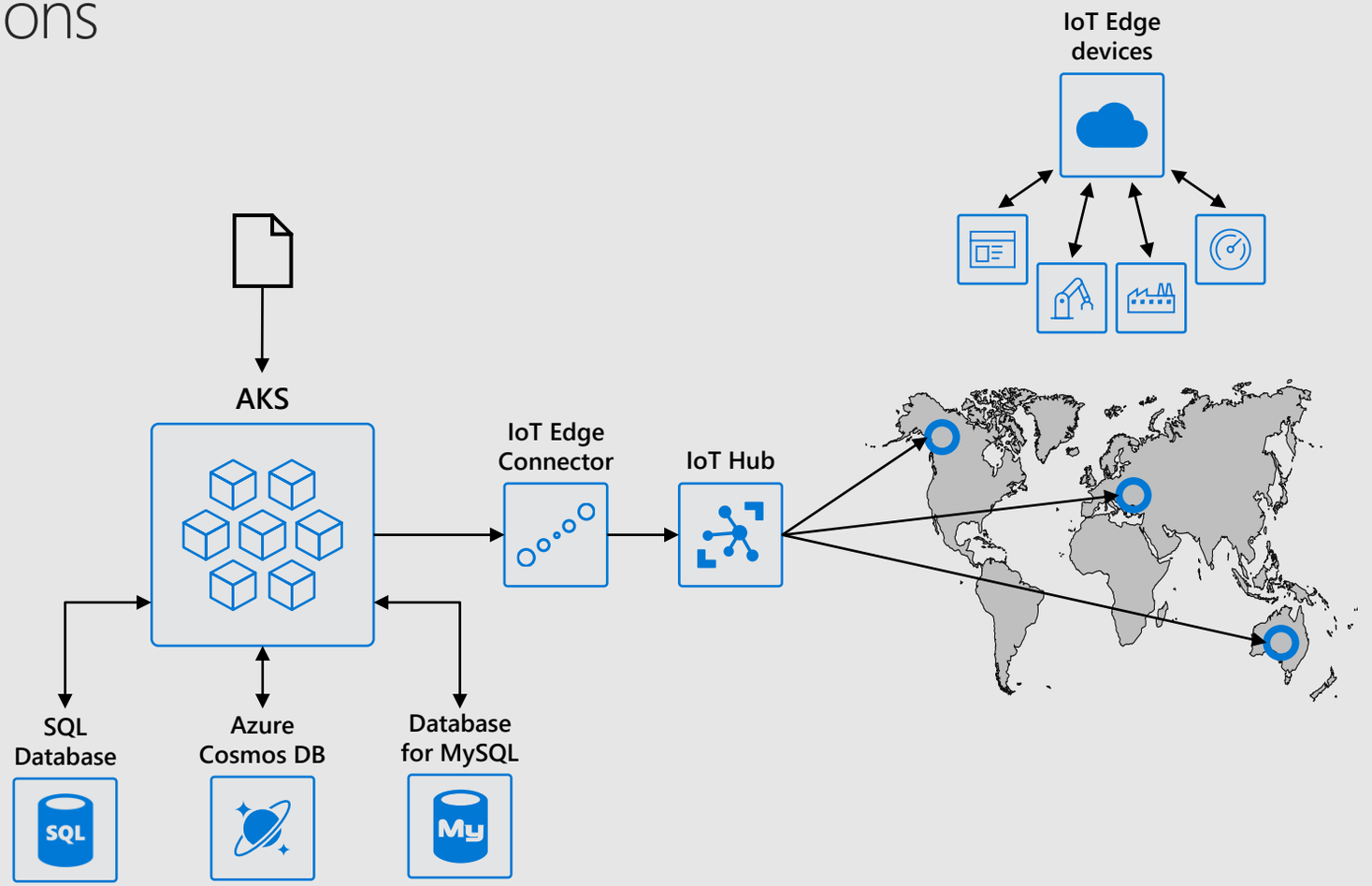
1. Package ML model into a container and publish to **Azure Container Registry**
2. **Azure Blob Storage** hosts training data sets and trained model
3. Use **Kubeflow** to deploy training job to AKS, distributed training job to AKS includes Parameter servers and Worker nodes
4. Serve production model using **Kubeflow**, promoting a consistent environment across test, control and production
5. AKS supports **GPU enabled VM**
6. Developer can build features querying the model running in AKS cluster



https://github.com/Azure/kubeflow-labs

Scalable Internet of Things solutions

- Portable code, runs anywhere
- Elastic scalability and manageability
- Quick deployment and high availability



Scalable Internet of Things solutions

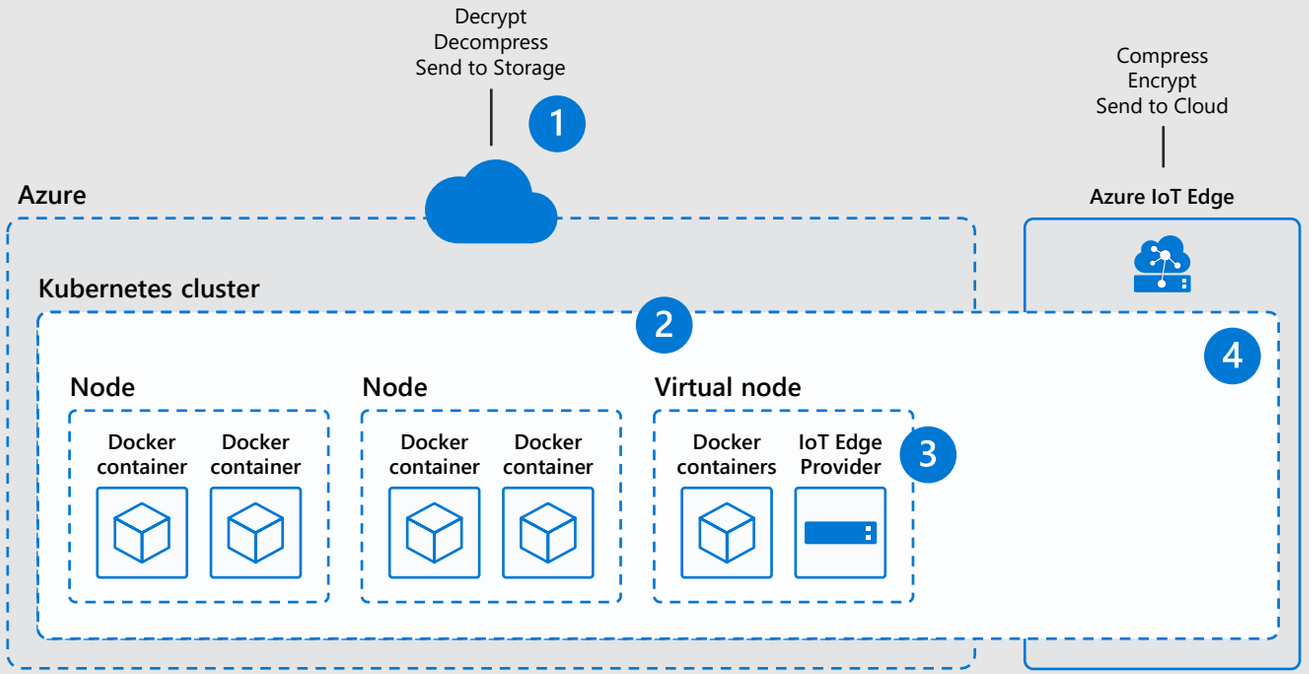
Capabilities

1. **Azure IoT Edge** encrypts data and send to Azure, which then decrypts the data and send to storage

2. **Virtual node**, an implementation of Virtual Kubelet, serves as the translator between cloud and Edge

3. **IoT Edge Provider in virtual node** redirects containers to IoT Edge and extend AKS cluster to target millions of edge devices

4. Consistent update, manage, and monitoring as one unit in AKS using single pod definition



Resources

Best support for your enterprise need

AKS WORKSHOP – Georgia, December 2019 - GitHub site:

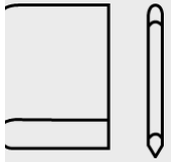
https://github.com/matjazperpar/AKS-Workshop-Georgia

Kubernetes 101 Docs
aka.ms/LearnAKS

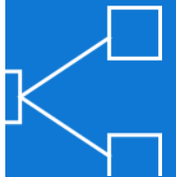
Best practices
aka.ms/aks/bestpractices

Hear from experts
aka.ms/k8s/lightboard

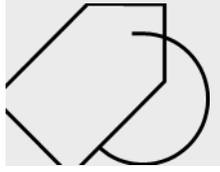
Case studies
aka.ms/aks/casestudy



Microservices architecture
aka.ms/aks/microservices



Try for free
aka.ms/aks/trial



Feedback on the roadmap? Tell us at https://aka.ms/aks/feedback

Microsoft Azure

Thank you.