

Deploying .Net Microservices w/ K8s, AKS and Azure DevOps



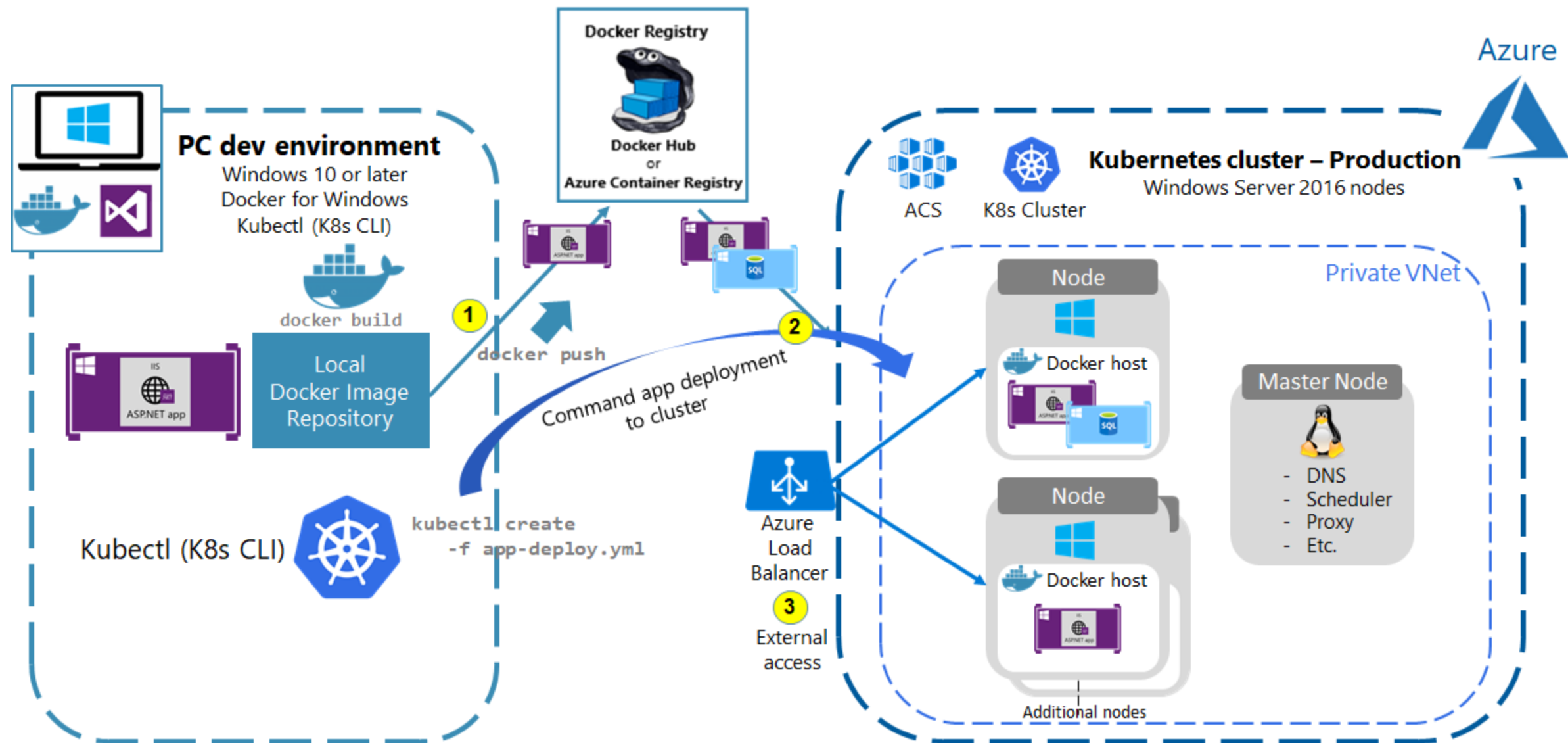
Mehmet Ozkaya

Software Architect, .NET

@ezozkme

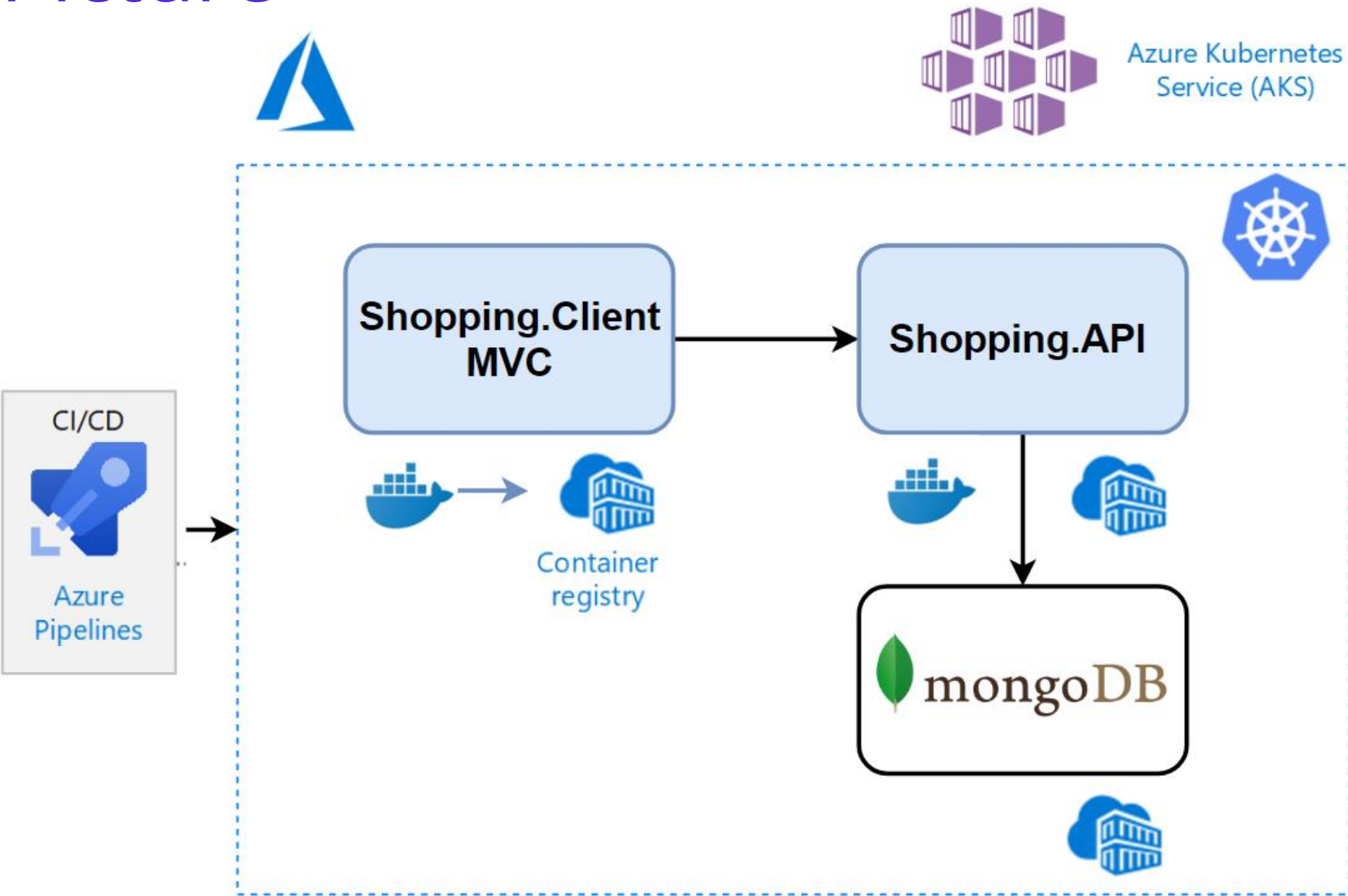


Scenario: Direct deployment to a Kubernetes cluster in Azure Container Service

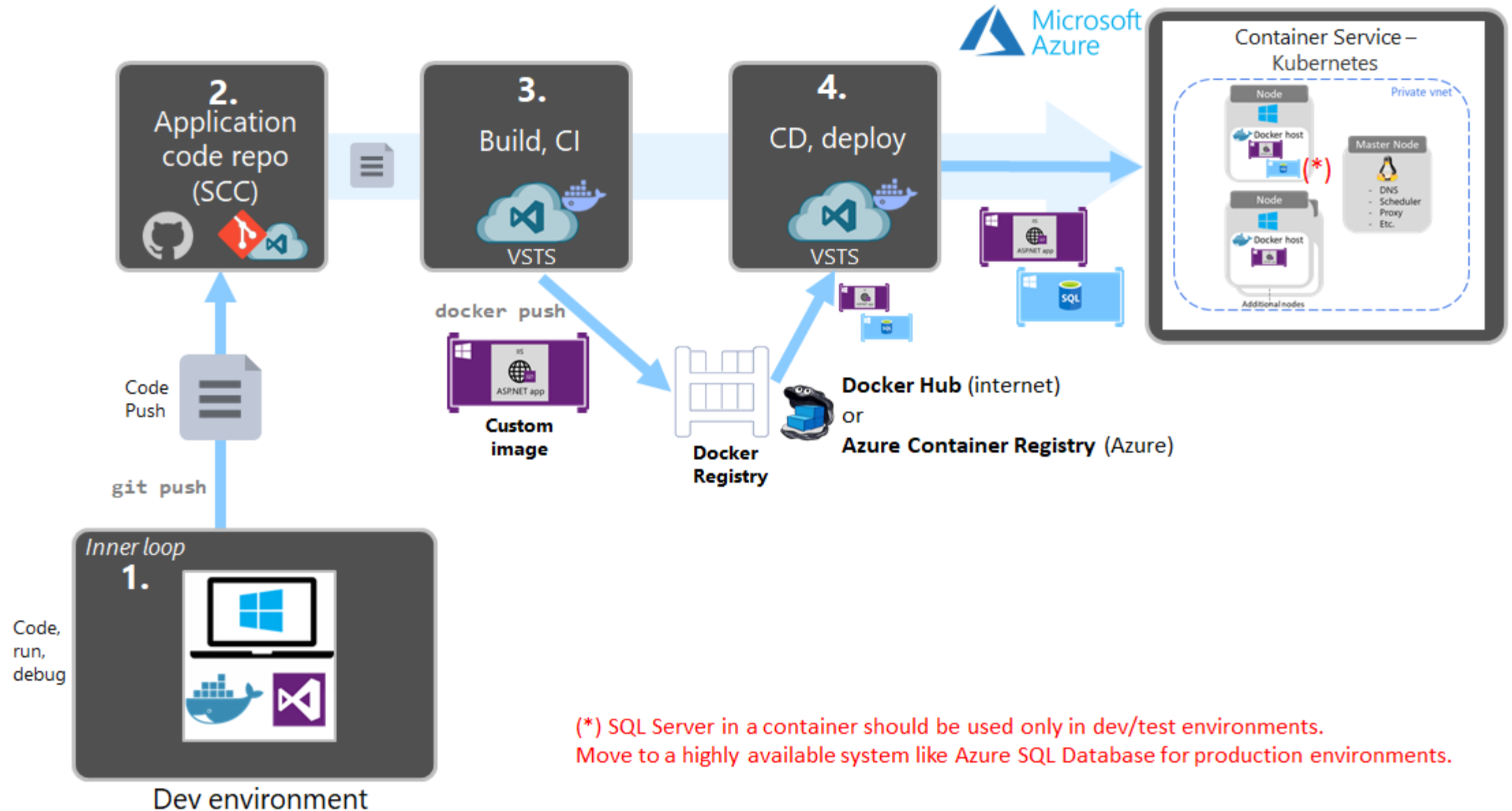


(*) SQL Server in a container should be used only in dev/test environments. Move to a highly available system like Azure SQL Database for production environments.

Big Picture



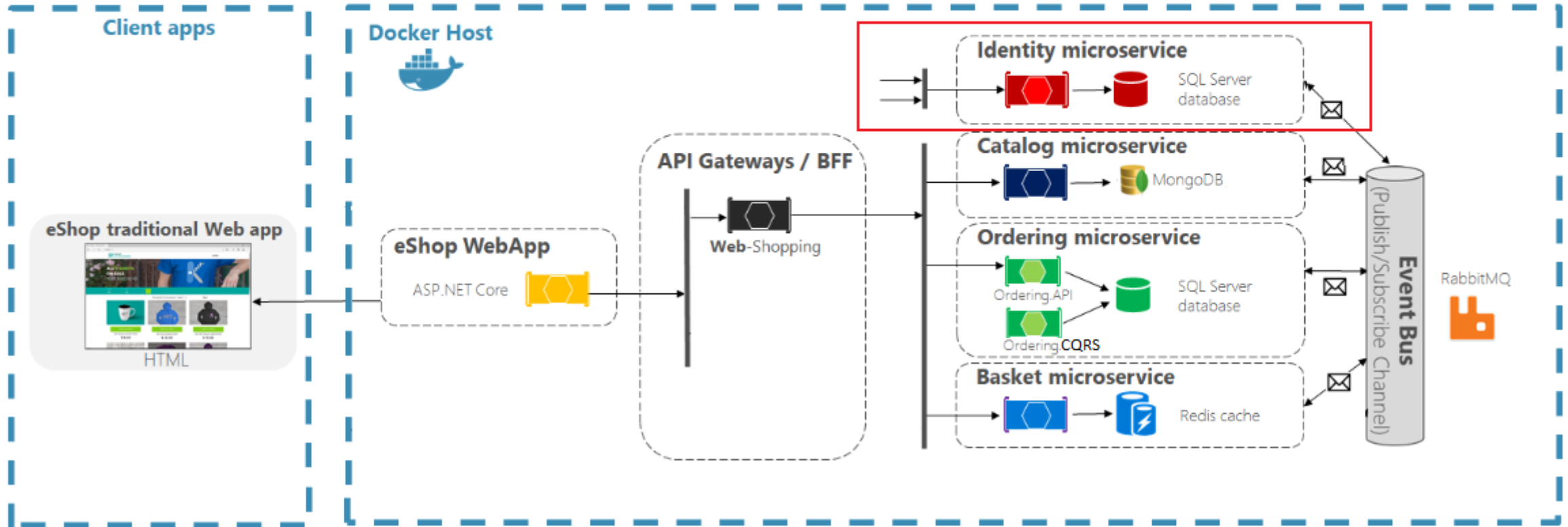
Scenario: Deploy to Kubernetes through CI/CD pipelines



Devops Microservices Architecture



aspnetrun-microservices Environment Architecture



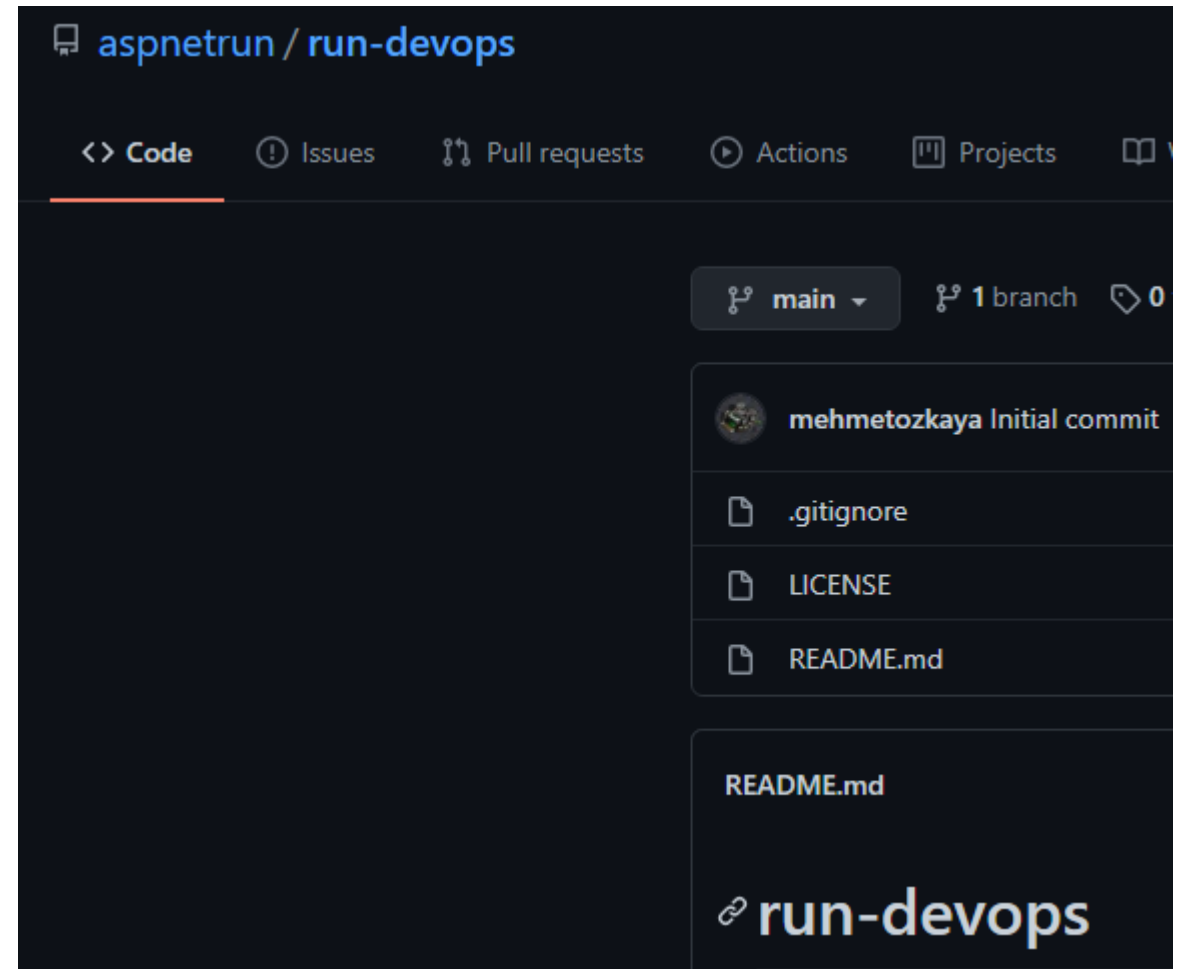
Prerequisites

- Good knowledge of C#
- Have knowledge of Asp.Net MVC and REST API's
- Docker Container knowledge
- Basics of Microservices Architecture

Source Code on Github

- Source code on Github
- Fork the repository
- Open issues
- Send Pull Requests

<https://github.com/aspnetrun/run-devops>

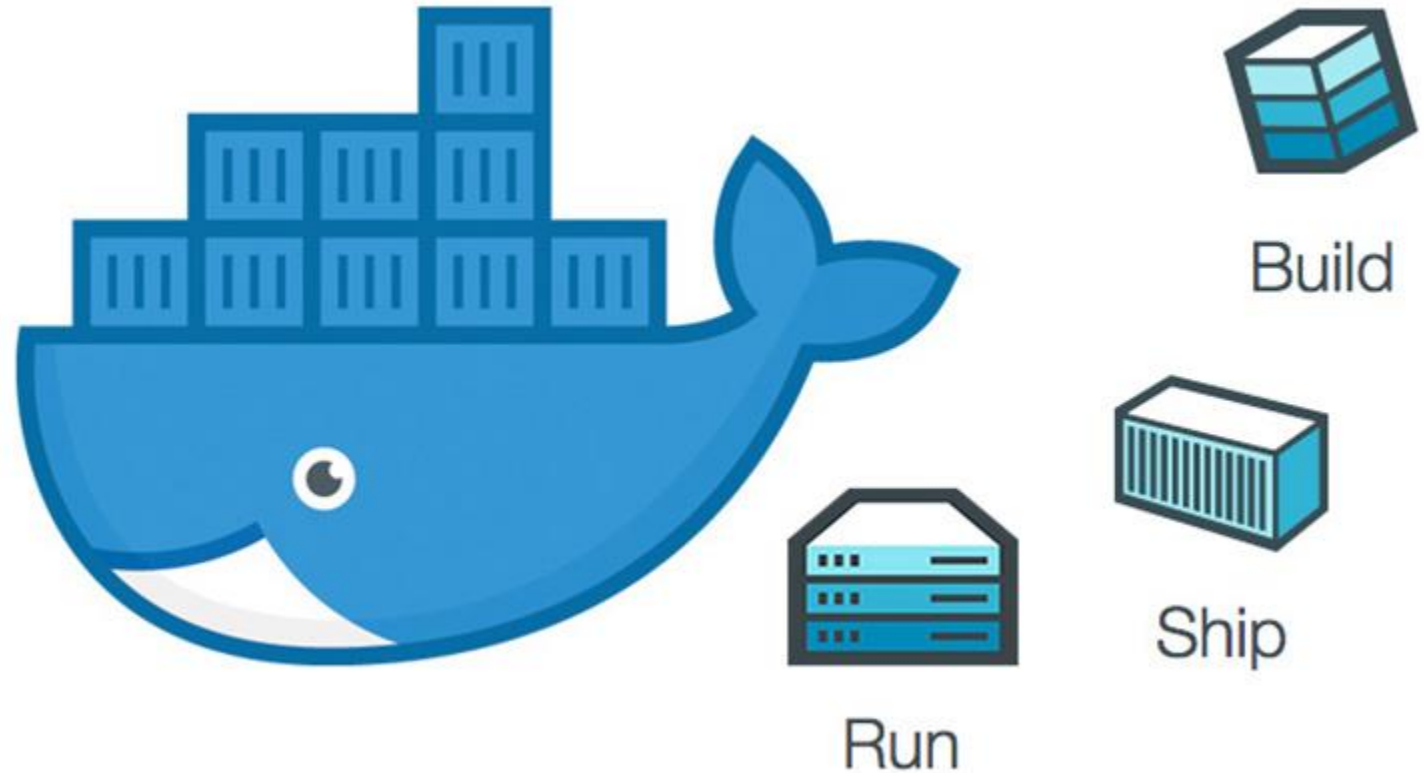


Tools that we will use

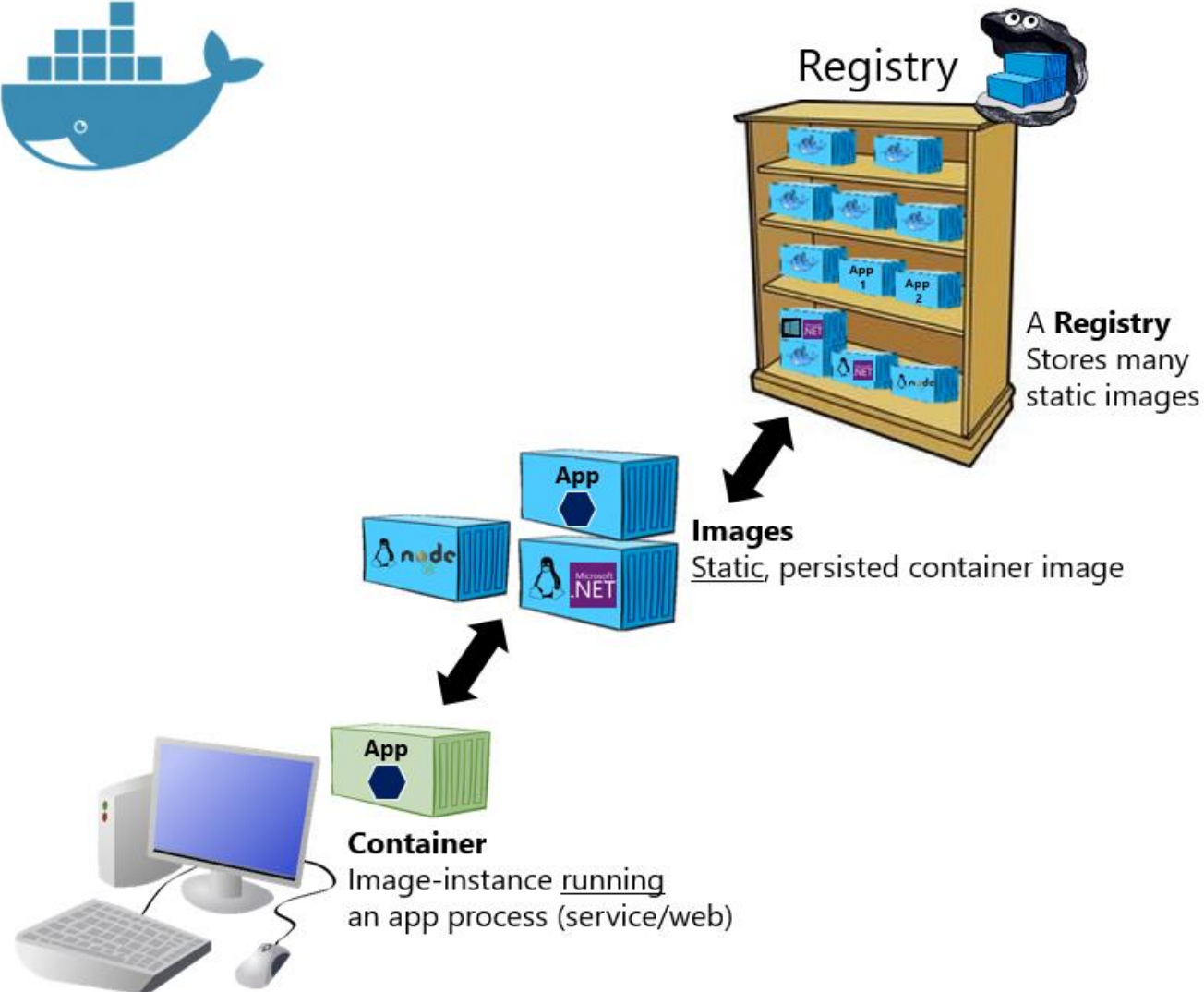
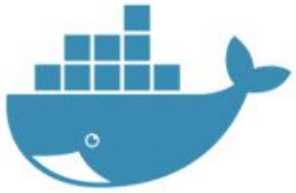
- Visual Studio 2019 and Visual Studio Code
- Docker Desktop and Docker Account for pushing images Docker Hub
- Azure Subscription for creating all azure resources
- Azure Devops Account for ci/cd devops pipelines

Docker and Container

- Developing, Shipping, and Running applications
- Reduce time to production
- Run anywhere
- Container packages code and dependencies



Basic taxonomy in Docker



Hosted Docker Registry

Docker Trusted Registry on-prem.

On-premises
(‘n’ private organizations)

Docker Hub Registry

Docker Trusted Registry on-cloud

Azure Container Registry

AWS Container Registry

Public Cloud
(specific vendors)

Google Container Registry

Quay Registry

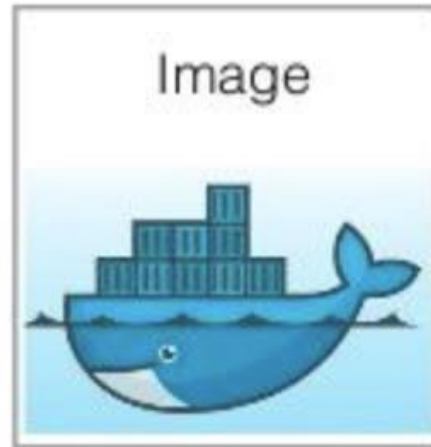
Other Cloud

Docker Application Containerization

```
FROM ubuntu:14.04
MAINTAINER Docker docker@docker.com
WORKDIR /app
COPY . /app
RUN apt-get update \
    && apt-get install -y python \
    && apt-get install -y python-dev \
    && pip install Flask
EXPOSE 5000
CMD ["python", "app.py"]
```

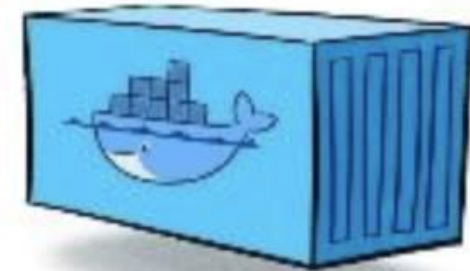
Dockerfile

build



Docker Image

run



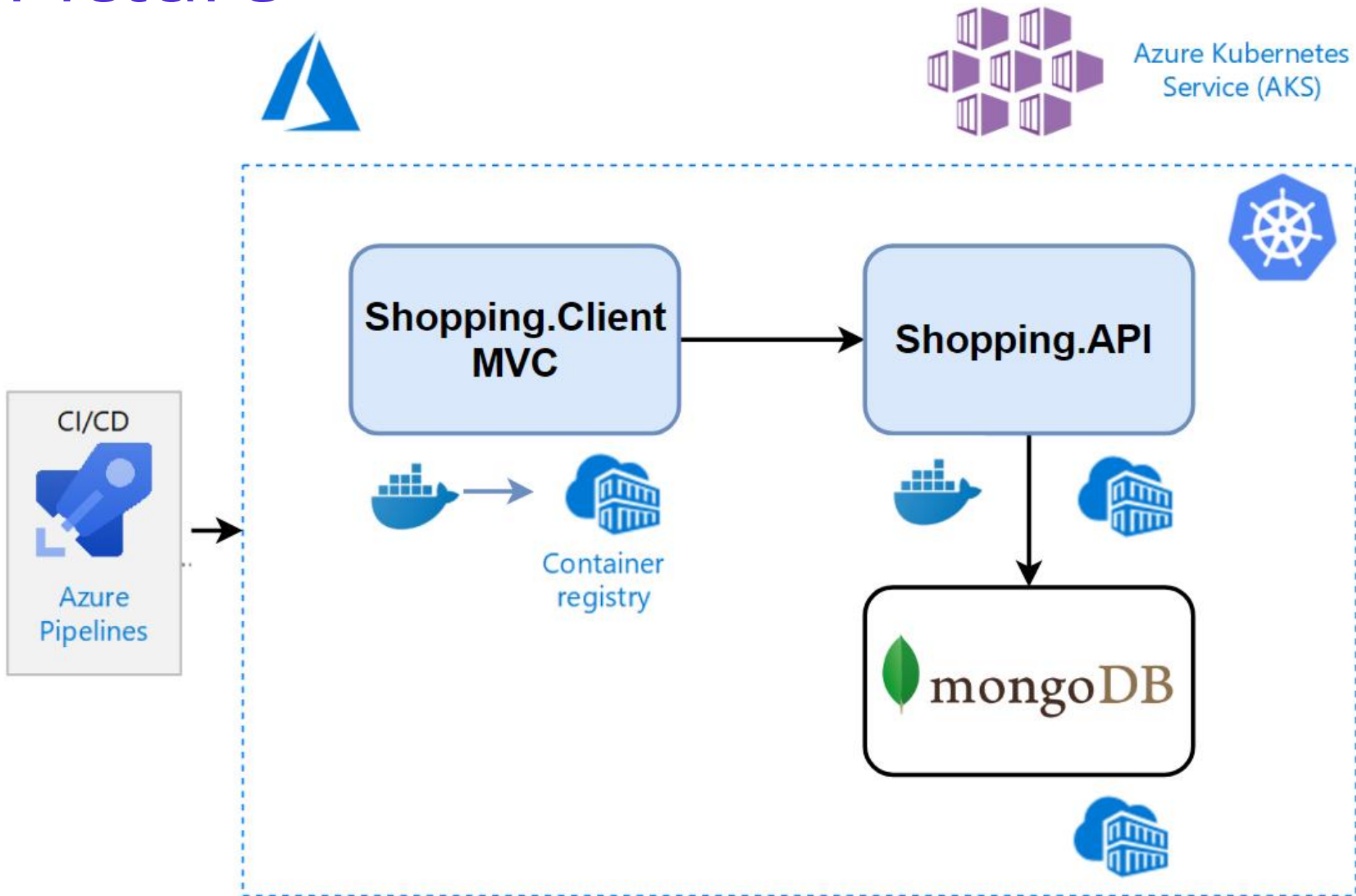
Docker Container

Section 2

Developing Your First Microservice

for deploying microservices

Big Picture

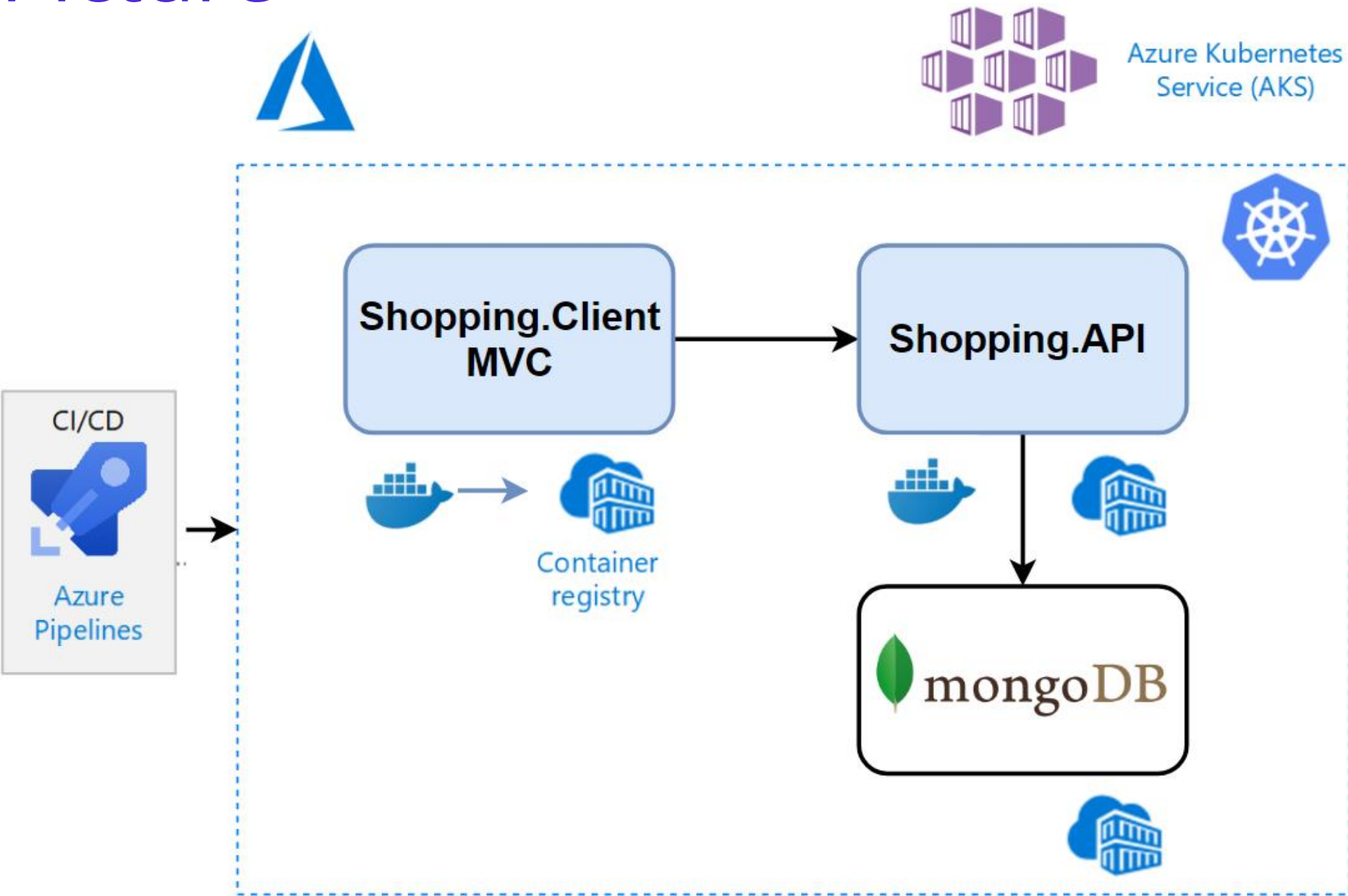


Section 3

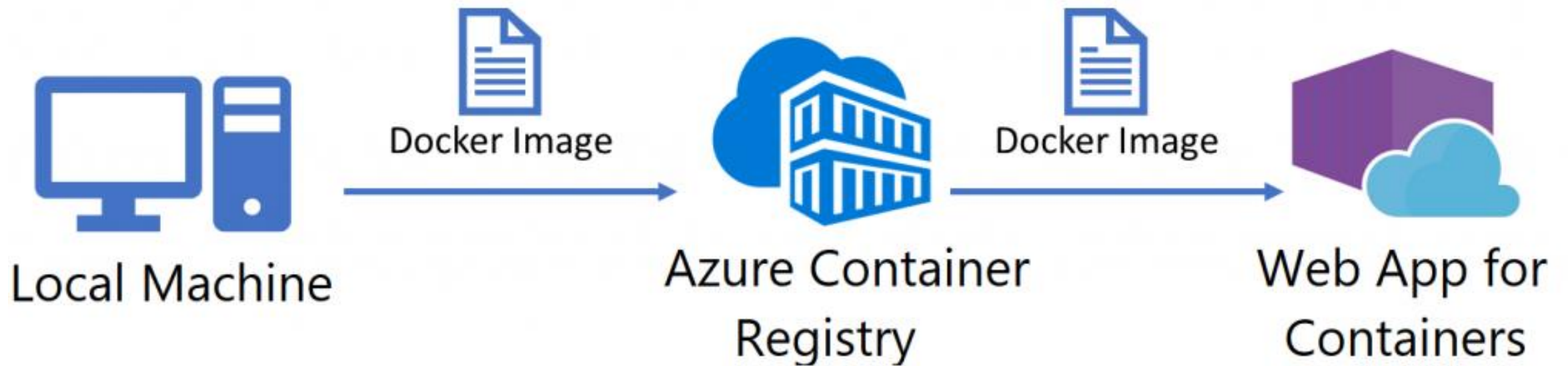
Deploy Shopping.Client Microservice

to Azure App Services - Web App for Containers

Big Picture



Deploy Shopping.Client Microservice to Azure App Services - Web App for Containers



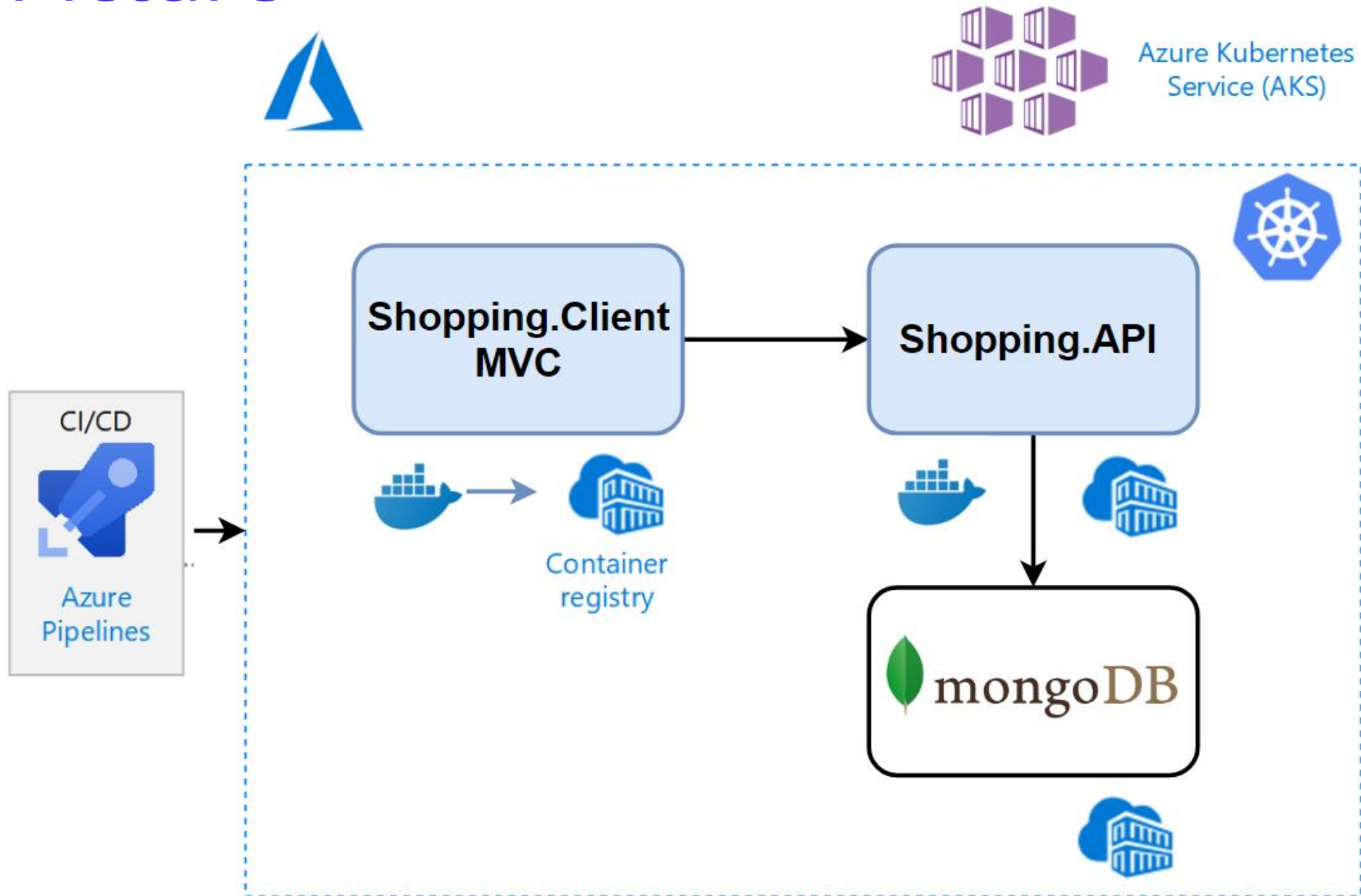
Section 4

Developing Shopping.API

Microservice

with MongoDB and Compose All Docker Containers

Big Picture

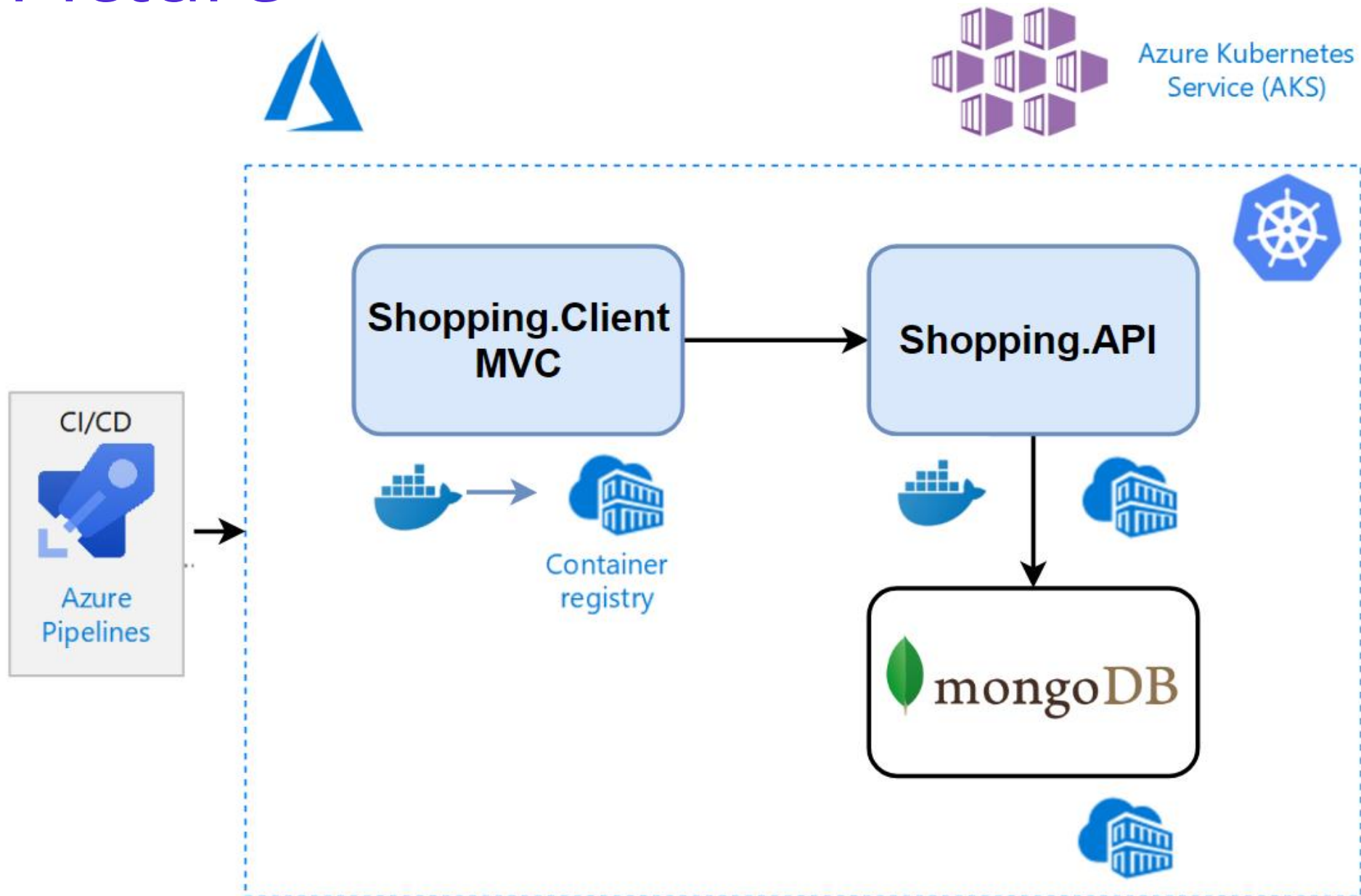


Section 5

Setup Mongo Docker Database

For connecting API

Big Picture

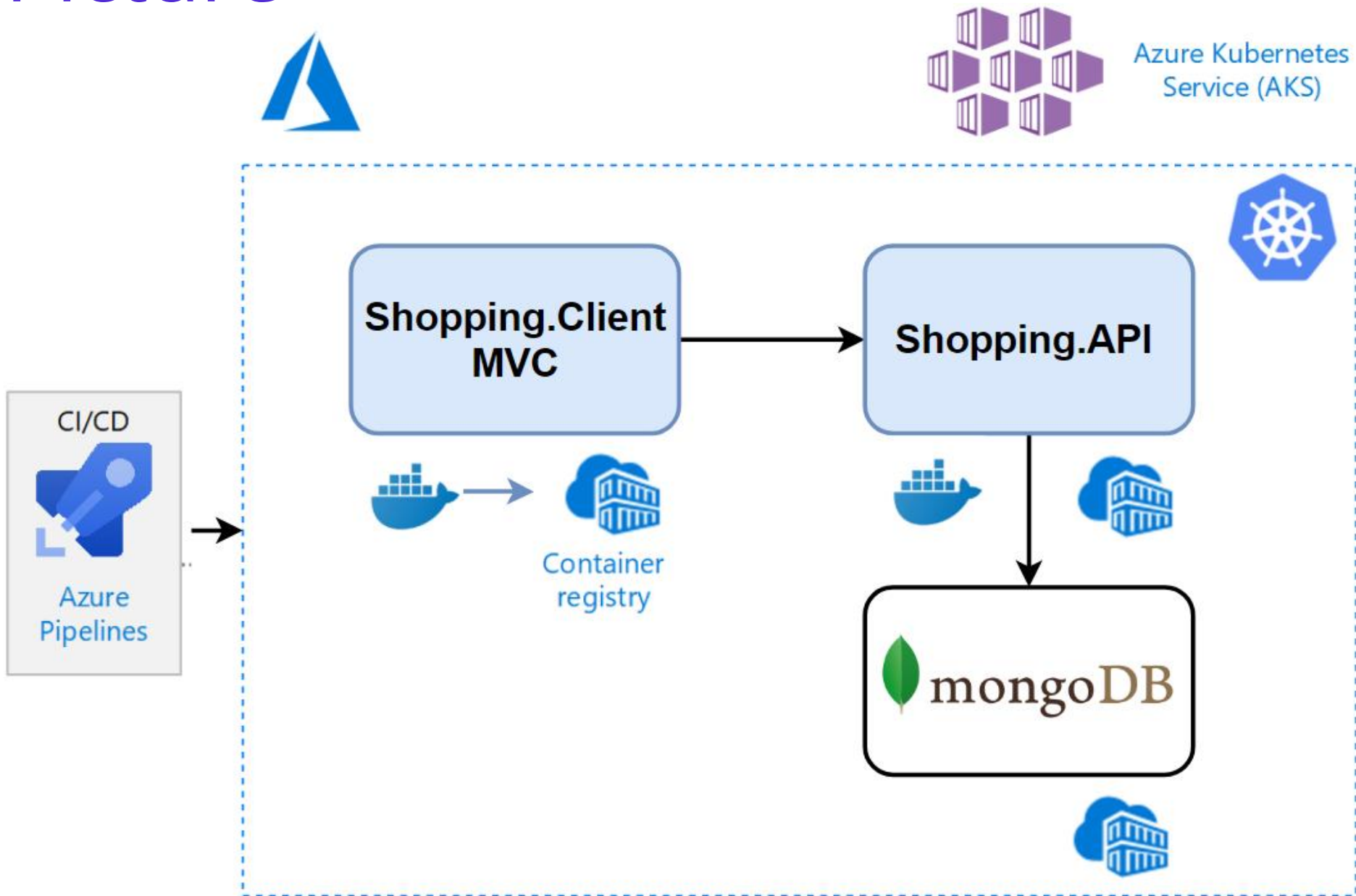


Section 6

Dockerize Microservices with Creating Multi-Container App

using Docker Compose

Big Picture

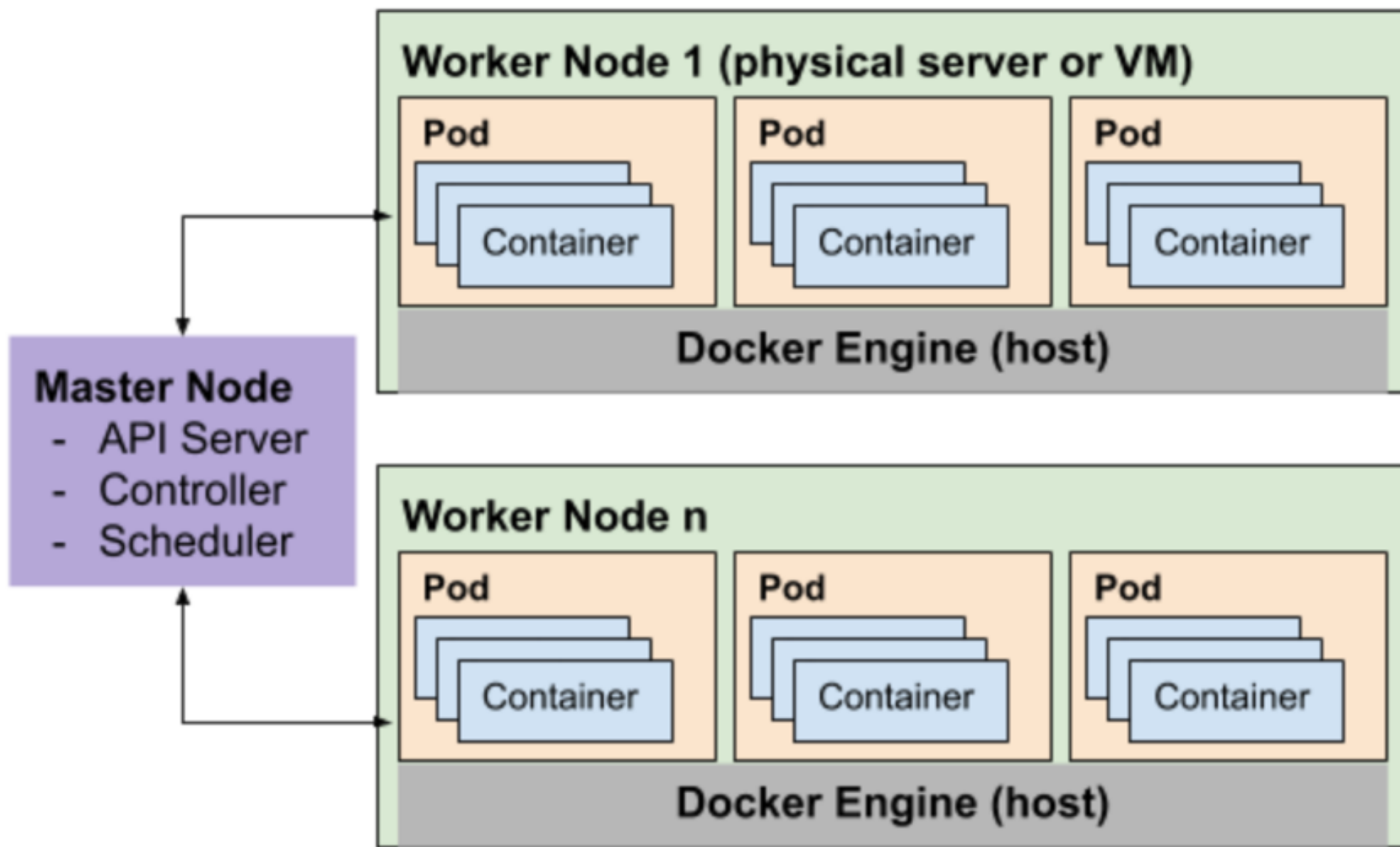


Section 7

Introduction to Kubernetes

for deploying microservices

Kubernetes Cluster



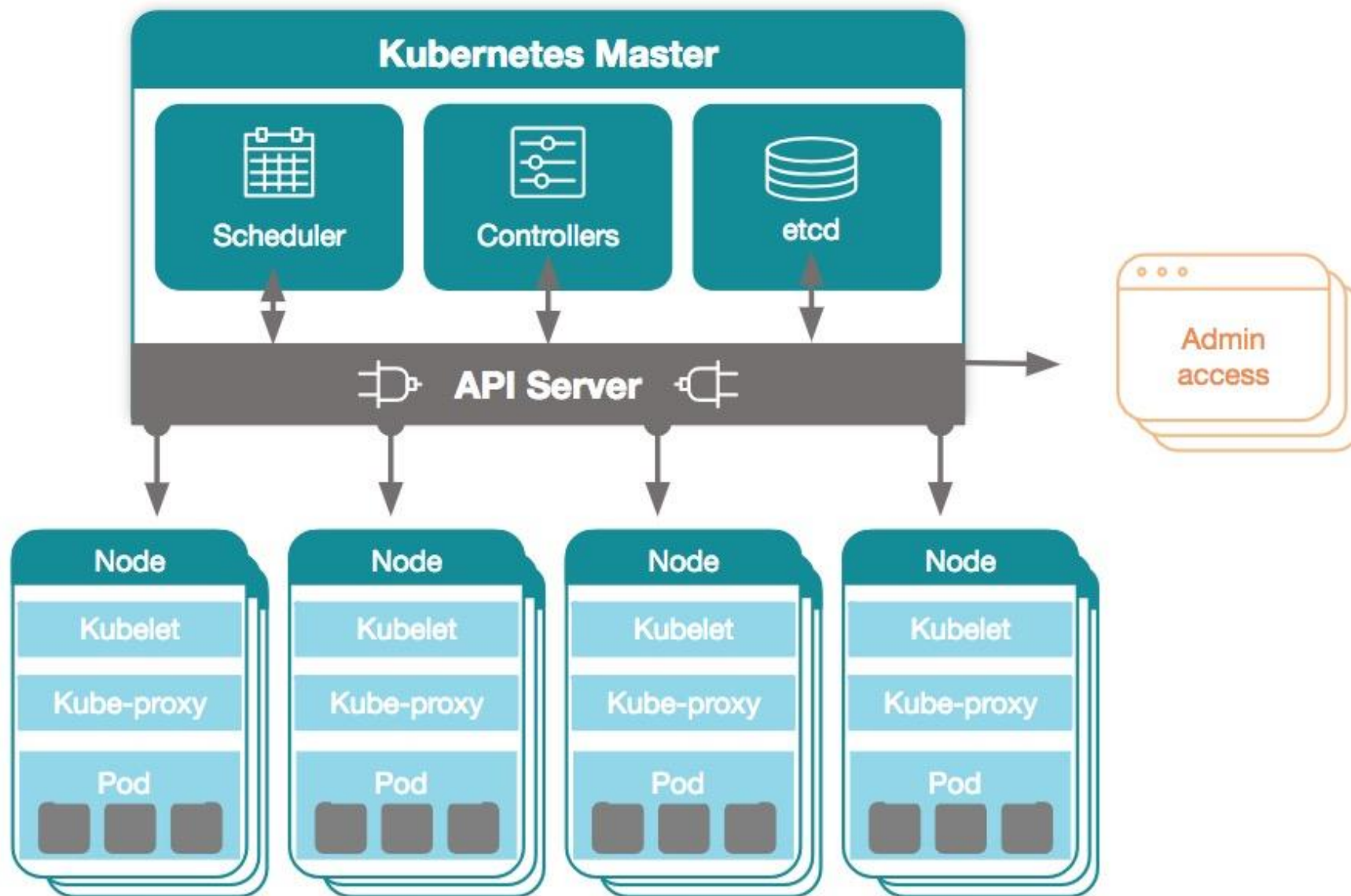
What is Kubernetes ?

- Kubernetes (also known as k8s or "kube")
- Open-source container orchestration platform
- Automates many of the manual processes
- Deploying, managing, and scaling containerized applications



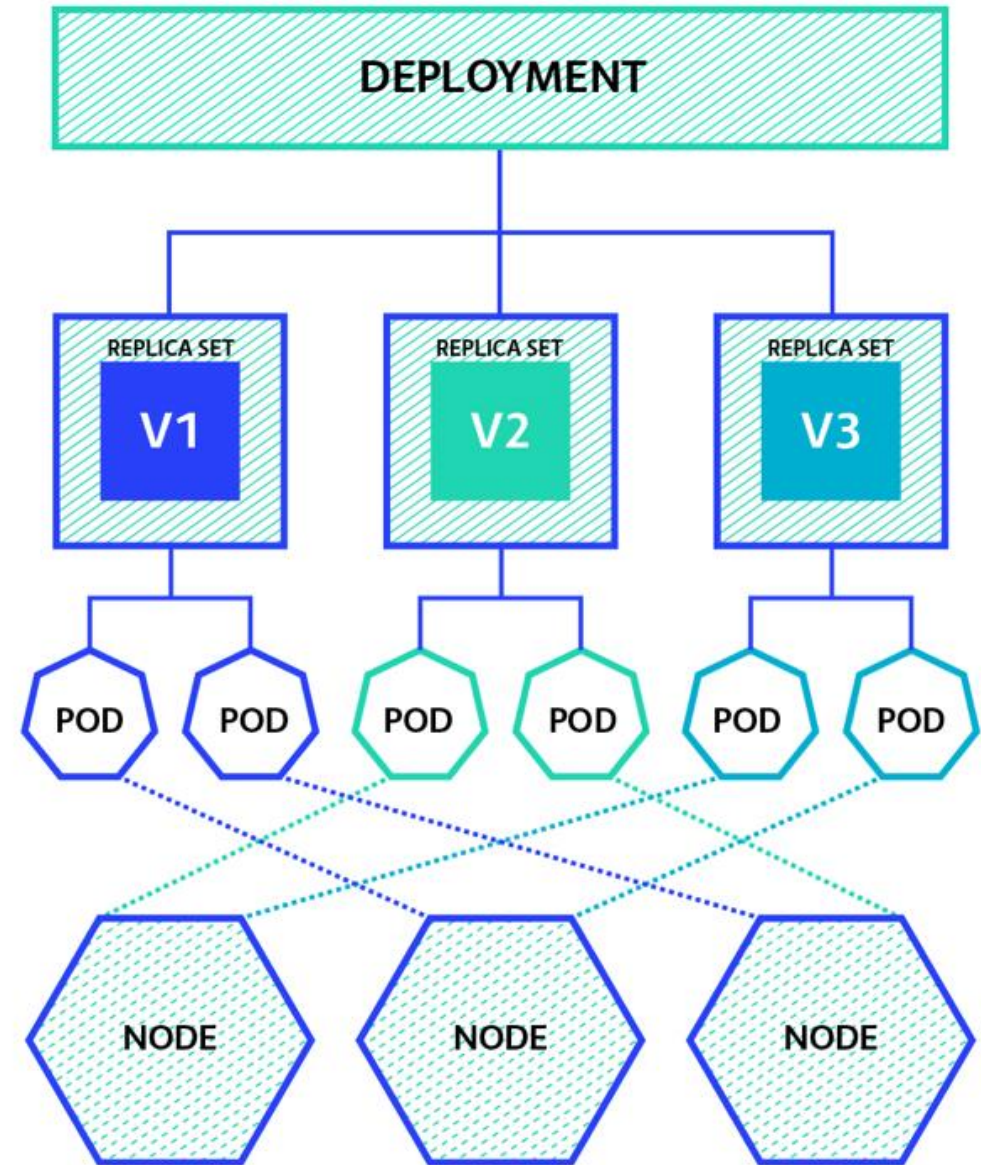
kubernetes

Kubernetes Architecture



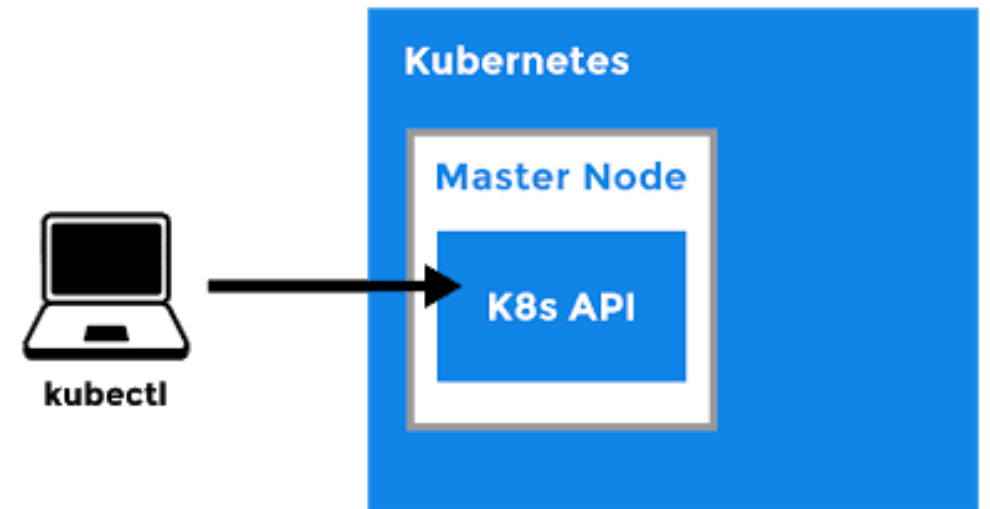
K8s Components

- Pods
- ReplicaSet
- Deployments
- Deployment > ReplicaSet > Pod
- Service
- ConfigMap - Secret



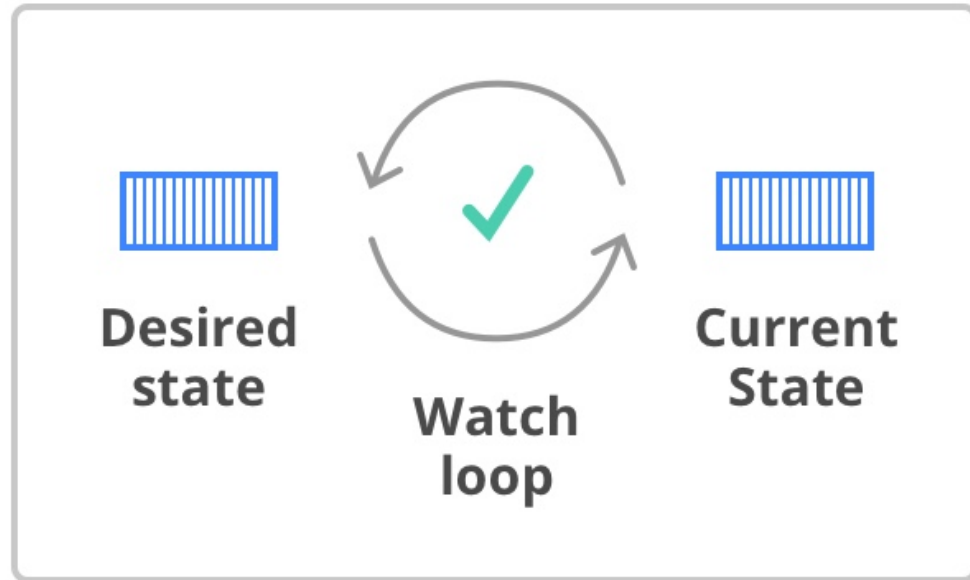
Kubectl Commands

- kubectl get nodes / pod / service / all
- kubectl create/edit/delete CRUD commands
- Abstraction : deployment -> replicaset -> pod
- Debugging Pods, Troubleshooting
- kubectl Cheat Sheet

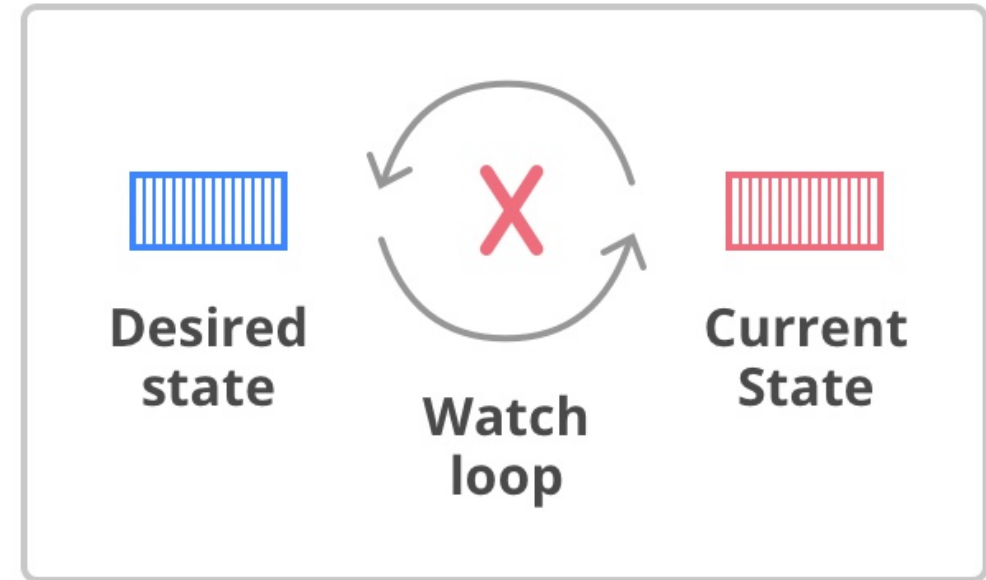


<https://kubernetes.io/docs/reference/kubectl/cheatsheet/>

Declarative vs Imperative



`kubectl run myapp --image myrepo:mytag
--replicas 2`



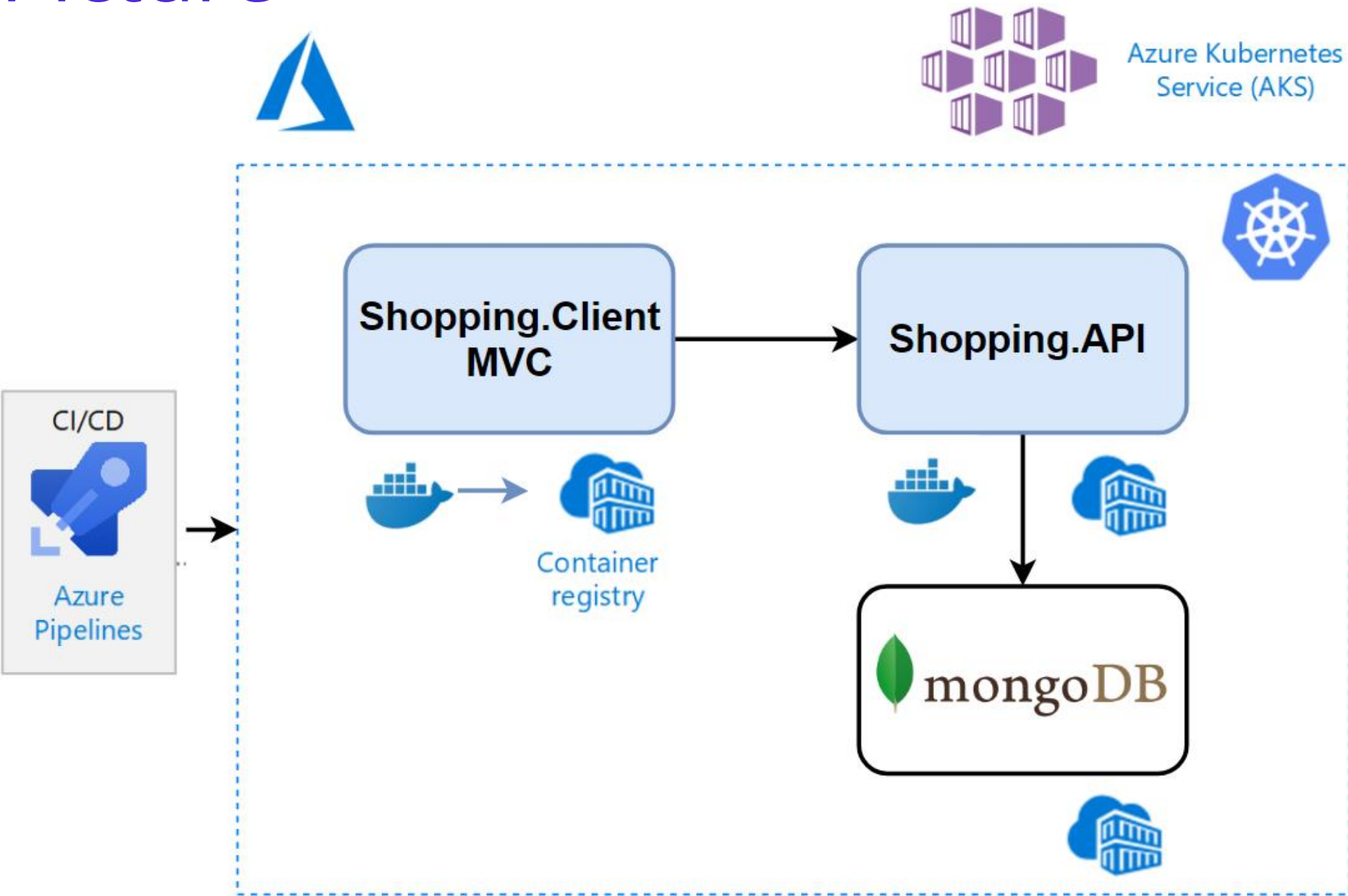
`kubectl apply -f app.yaml`

Section 8

Deploying Microservices on Kubernetes

for Shopping microservices

Big Picture



Planning to Shopping Microservices yaml files

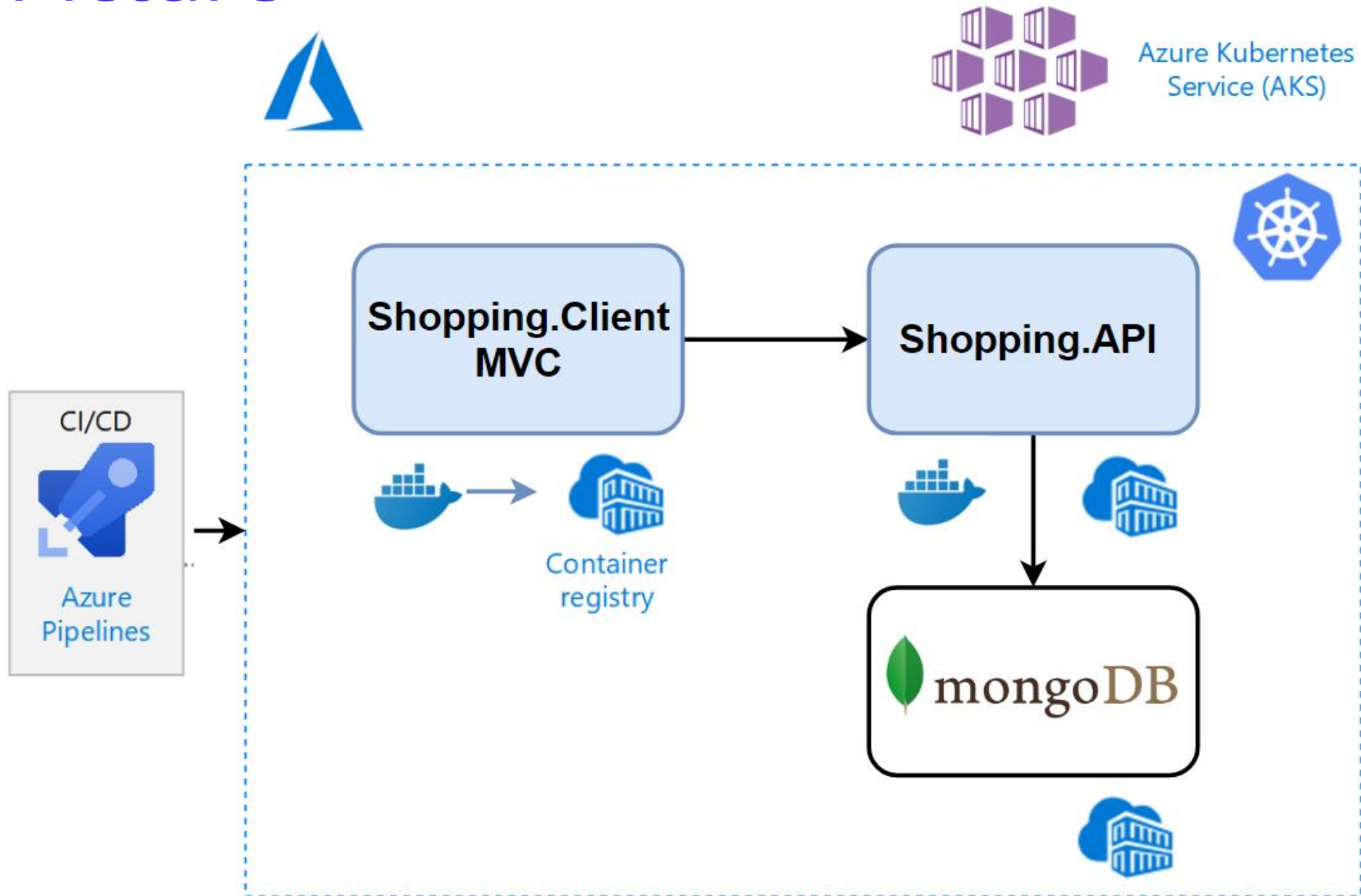
- **MongoDb** -> Deployment, Service, ConfigMap(db url), Secret (Db username-password)
- **Shopping.API** -> Deployment, Service, set env variables config map
- **Shopping.Client** -> Deployment, Service (LoadBalancer for External calls)
- External -> Shopping.Client -> Shopping.API ->MongoDb

Section 9

Deploying Microservices on AKS with ACR

into Cloud Azure Kubernetes Service (AKS) with using Azure Container Registry (ACR)

Big Picture



Azure Container Registry (ACR)



Local development PC
(Windows 10 + Docker for Windows)

Your app's bits



docker build

Compile &
Publish bits



Local Image Repository



Your custom image



.NET base image



Windows Server Core
base image



SQL Server image

docker push



Docker Registry

(Docker Hub or Azure Container Registry)



Your custom image



.NET base image



Windows Server Core
base image



SQL Server image

Other base images



docker pull

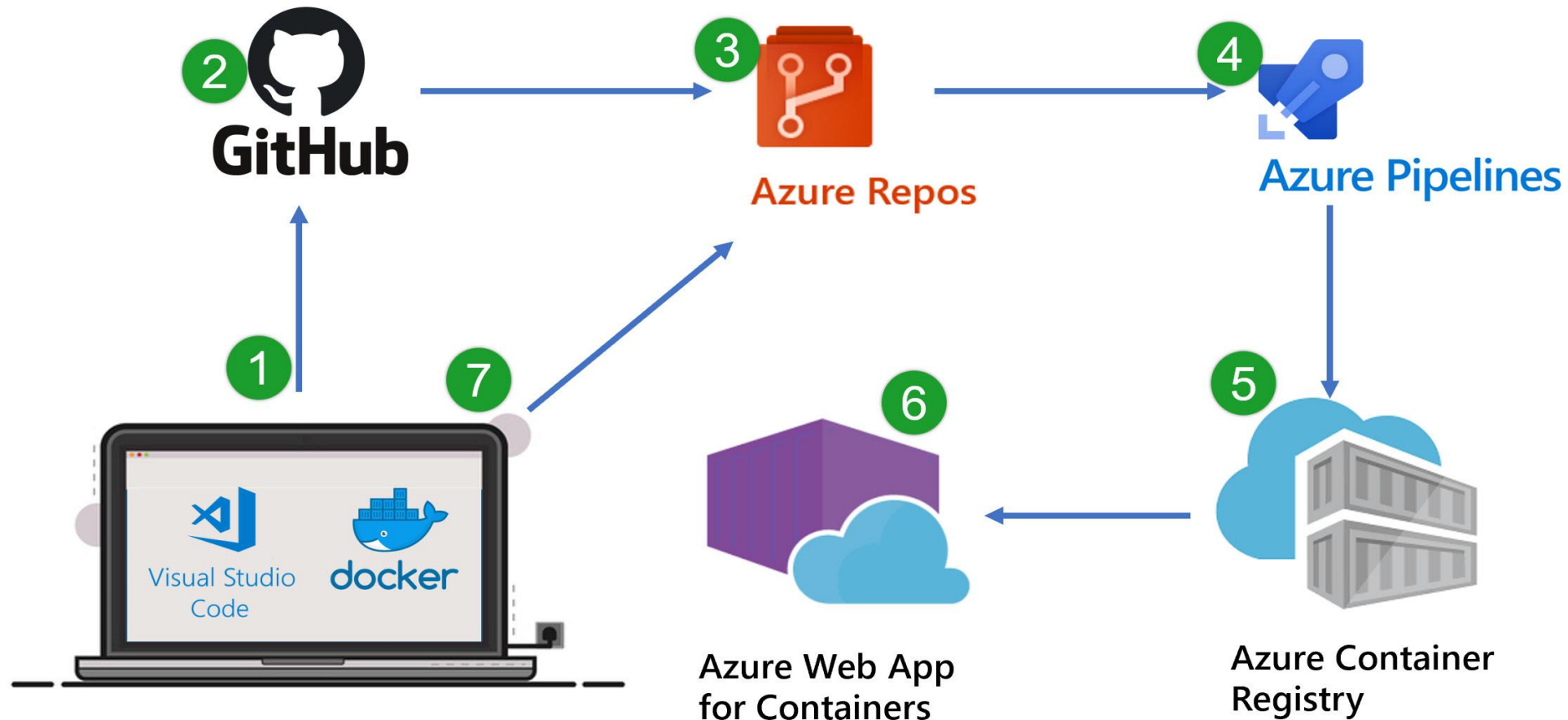
Production
Environments
(Cloud or On-Prem.)



Your app running in
a container



Example use of ACR in Devops Pipeline

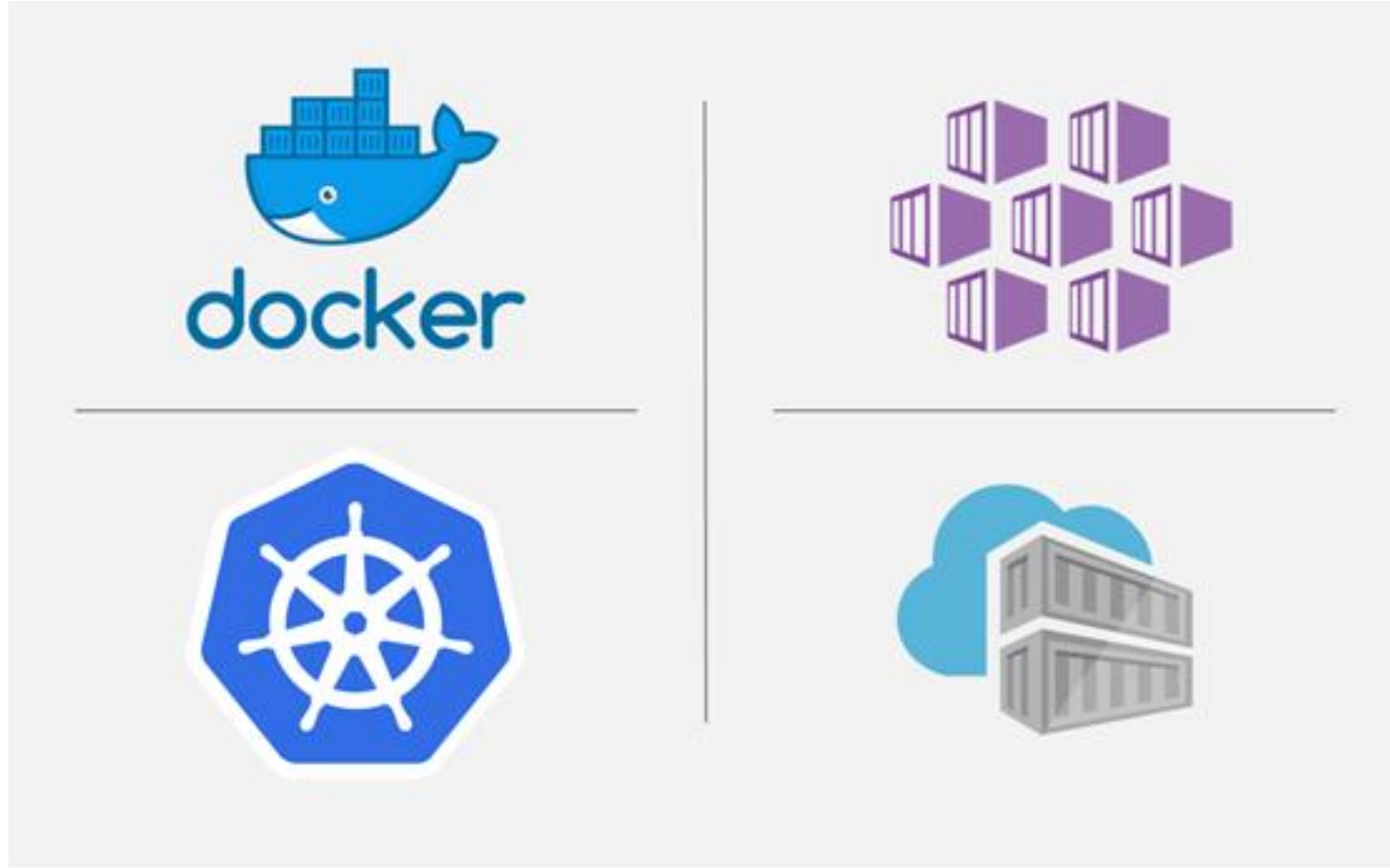


Azure Kubernetes Services (AKS)

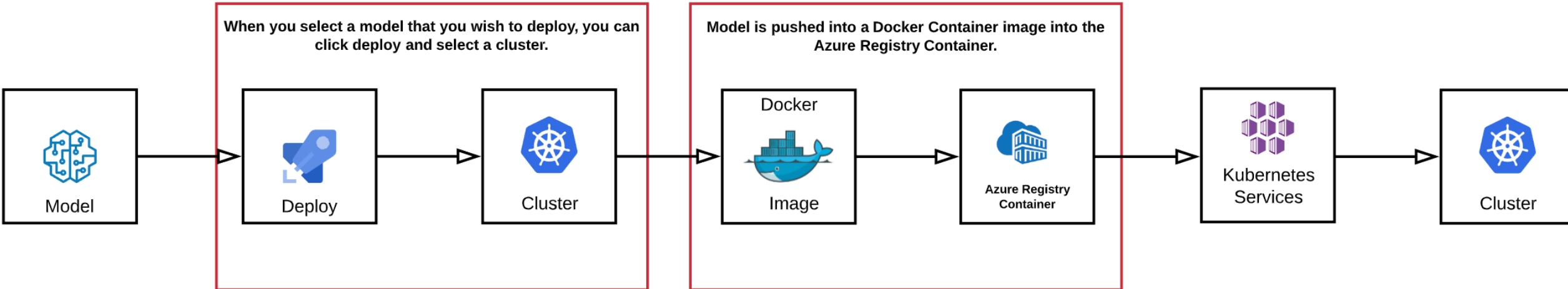
- Deploy a managed Kubernetes
- Reduces the complexity and operational overhead
- Azure handles critical tasks
- Maintenance for you
- Free plan



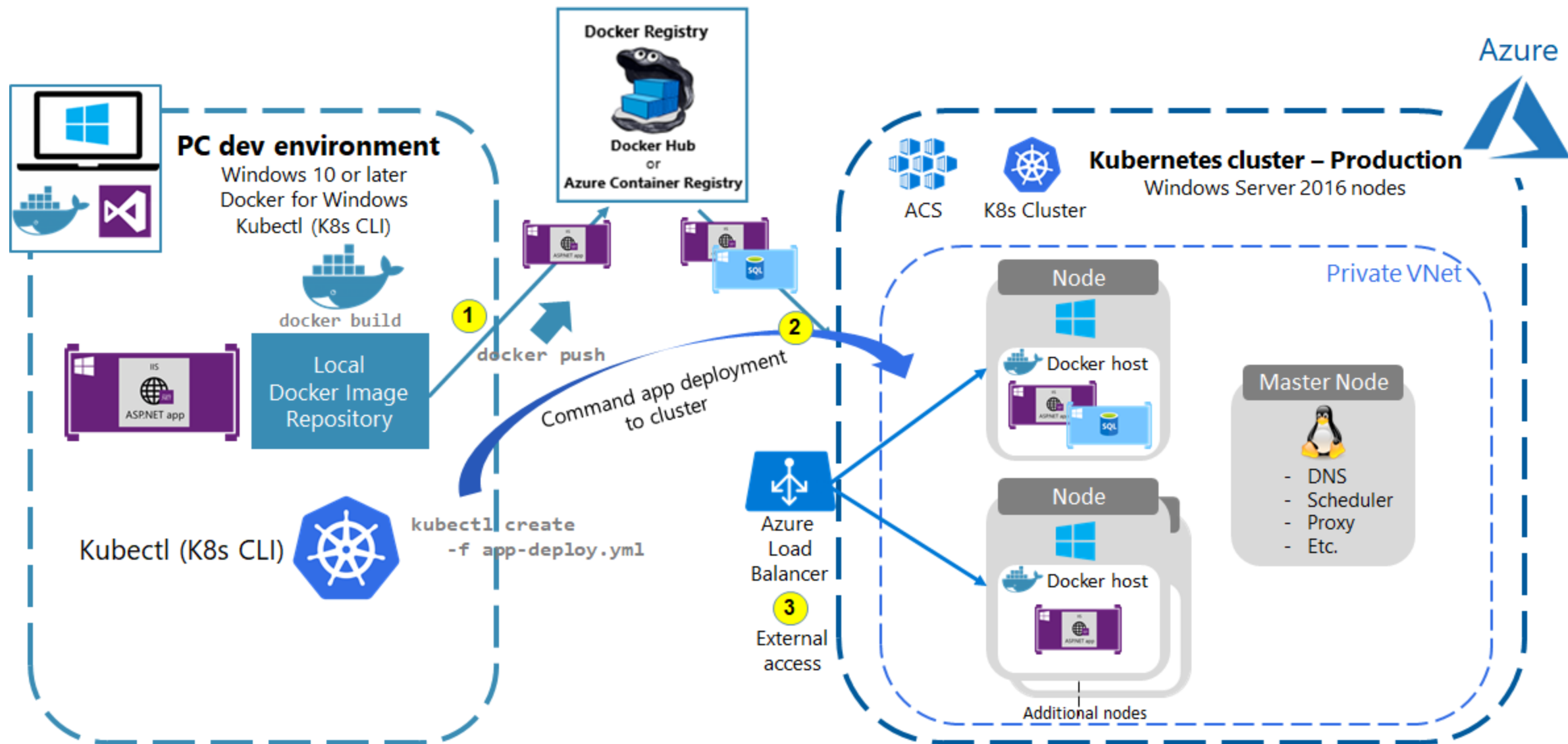
Shifting Local to Cloud



Steps to the AKS Deployment



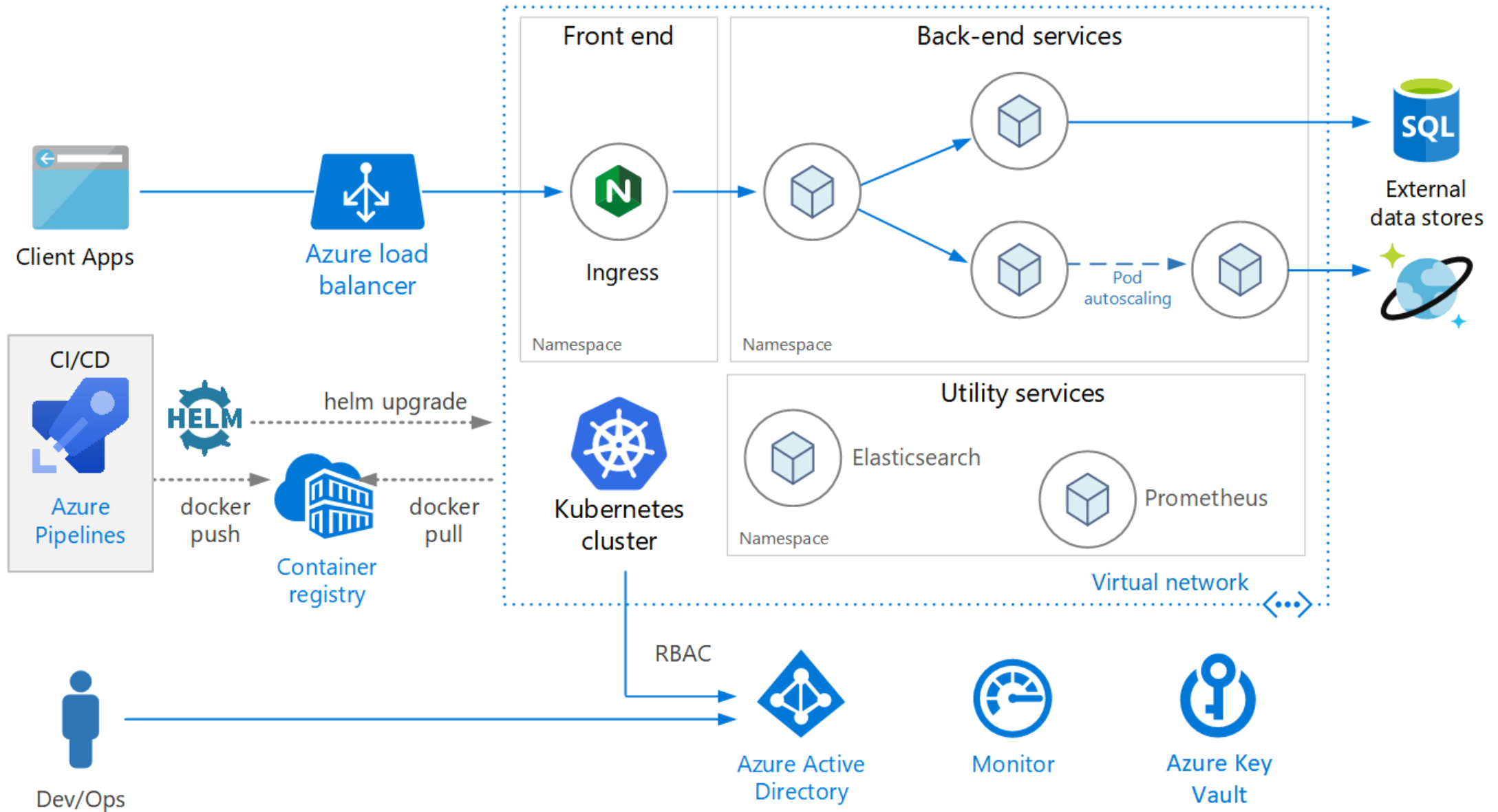
Scenario: Direct deployment to a Kubernetes cluster in Azure Container Service



(*) SQL Server in a container should be used only in dev/test environments. Move to a highly available system like Azure SQL Database for production environments.



Azure Kubernetes
Service (AKS)

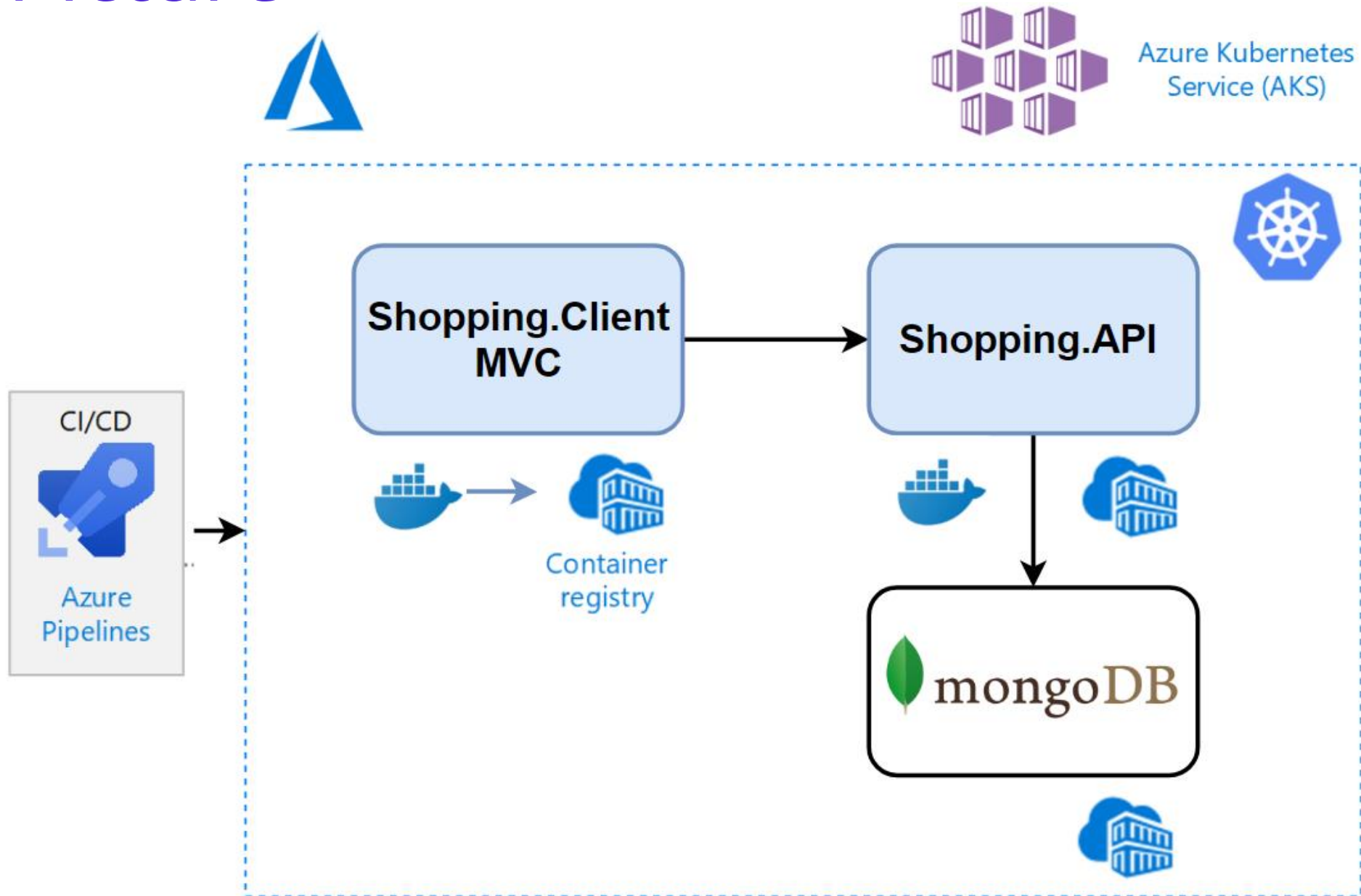


Section 10

Azure Devops

into Cloud Azure Kubernetes Service (AKS) with using Azure Container Registry (ACR)

Big Picture



Azure DevOps



Azure Boards



Azure Repos



Azure Pipelines



Azure Test Plans



Azure Artifacts

Introducing Azure DevOps



Azure
Boards

Plan, track, and discuss work across teams, deliver value to your users faster.



Azure
Repos

Unlimited cloud-hosted private Git repos. Collaborative pull requests, advanced file management, and more.



Azure
Pipelines

CI/CD that works with any language, platform, and cloud. Connect to GitHub or any Git provider and deploy continuously to any cloud.



Azure
Test Plans

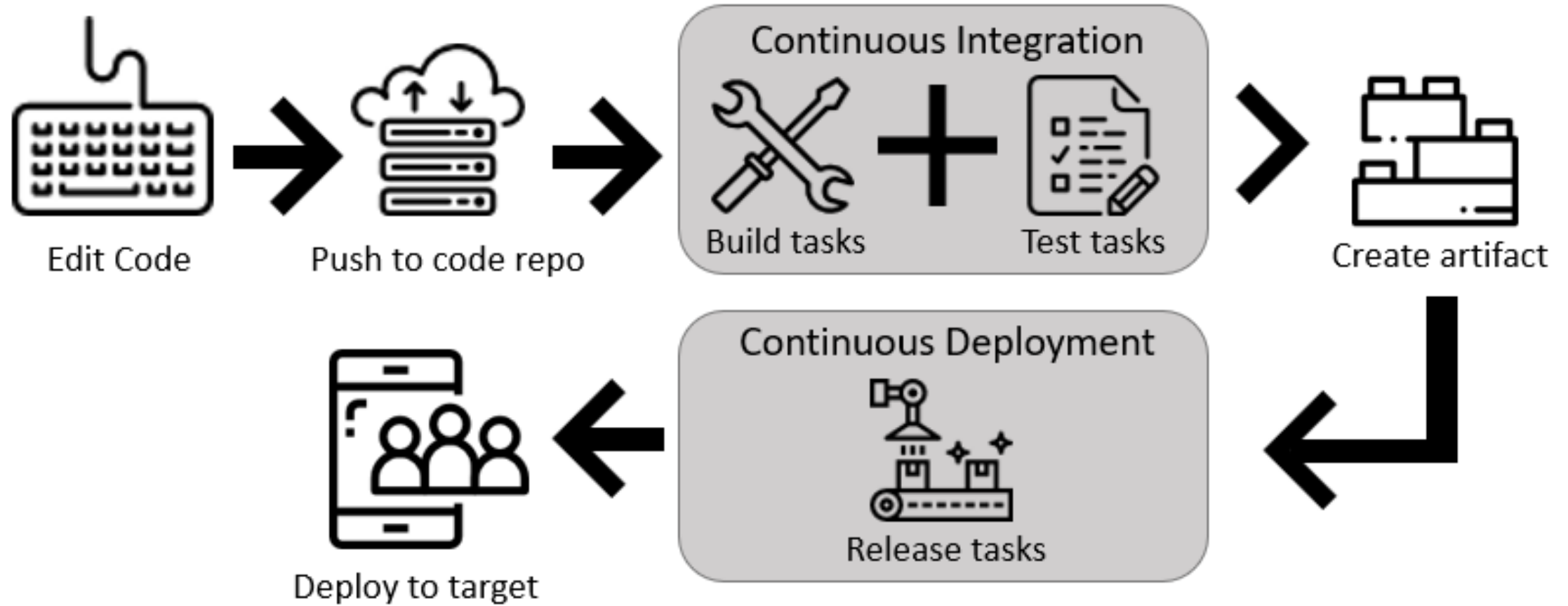
The test management and exploratory testing toolkit that lets you ship with confidence.



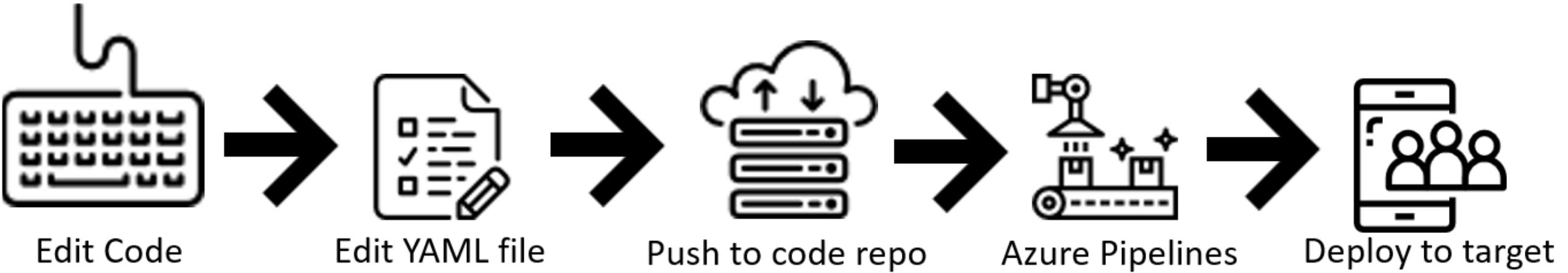
Azure
Artifacts

Create, host, and share packages. Easily add artifacts to CI/CD pipelines.

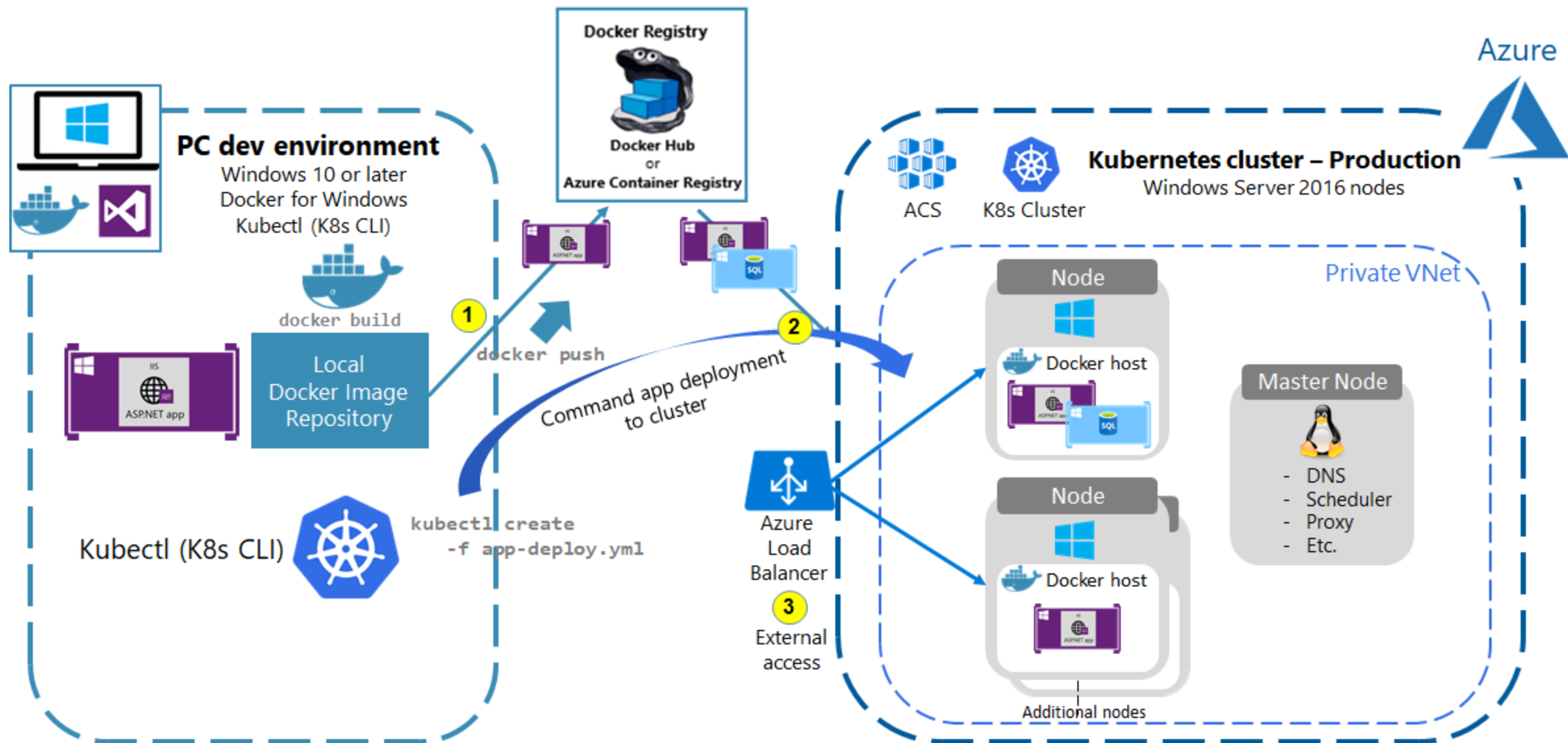
CI/CD



Azure Pipelines

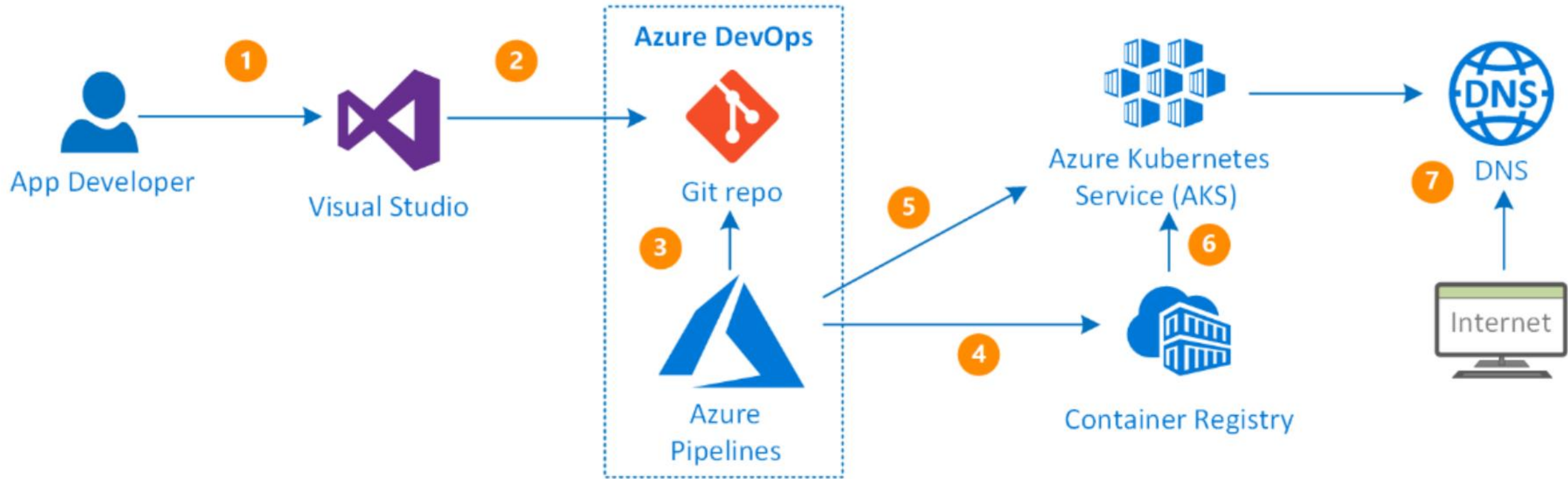


Scenario: Direct deployment to a Kubernetes cluster in Azure Container Service

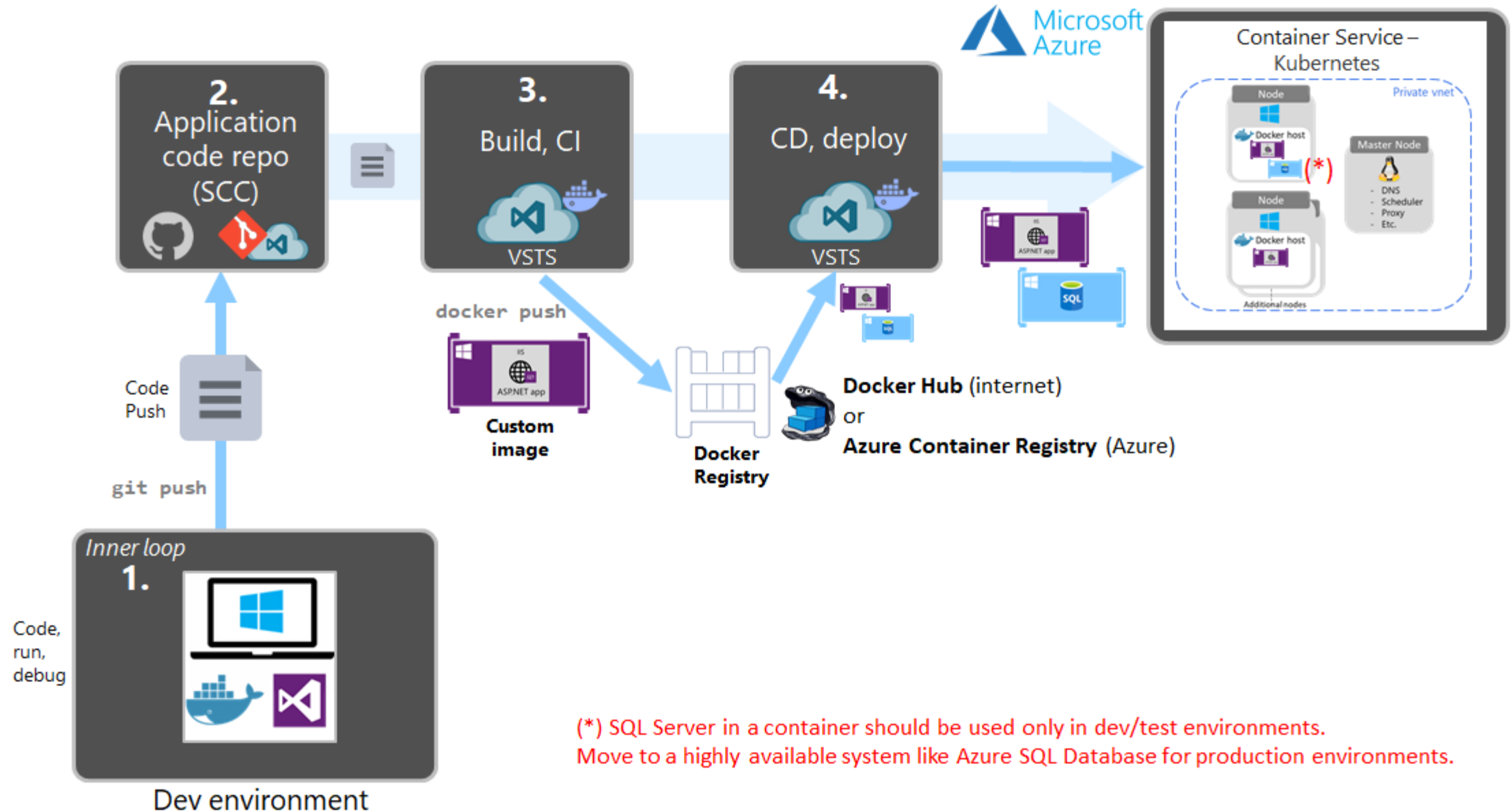


(*) SQL Server in a container should be used only in dev/test environments. Move to a highly available system like Azure SQL Database for production environments.

Steps to Azure DevOps Deployment



Scenario: Deploy to Kubernetes through CI/CD pipelines



Section 11

Automate Existing Microservices

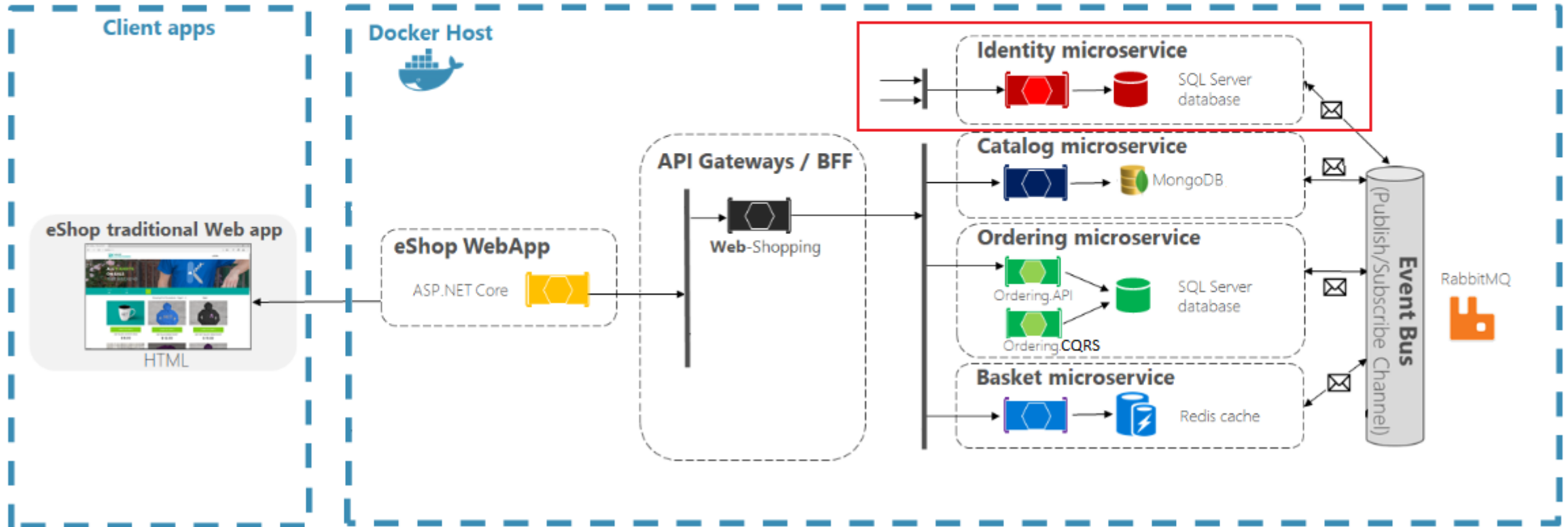
Reference Application

with AKS and Azure Devops

Devops Microservices Architecture



aspnetrun-microservices Environment Architecture





Existing Microservices Project

Microservices Architecture and Implementation on .NET

Building Microservices on .Net which used Asp.Net Web API, Docker, RabbitMQ, Ocelot API Gateway, MongoDB, Redis, SqlServer

★★★★★ 0.0 (0 ratings) 0 students enrolled

Created by Mehmet Özkaya Published 6/2020  English  English [Auto]

- Github Repository -> <https://github.com/aspnetrun/run-aspnetcore-microservices>

Thanks!

Follow me on github

<https://github.com/mehmetozkaya>

<https://github.com/aspnetrun>

Follow me on twitter

<https://twitter.com/ezozkme>

Follow me on medium

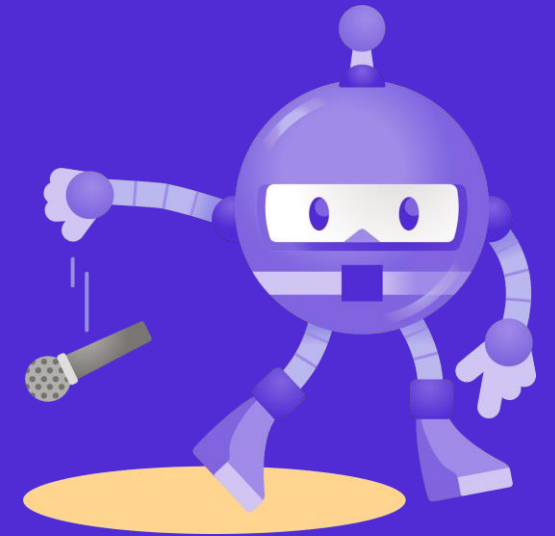
<https://mehmetozkaya.medium.com/>

Send mail me anything you can ask

ezozkme@gmail.com

Check my other courses on udemy:

<https://www.udemy.com/user/ff0e5c8c-dd71-443e-be0a-e73ba821f7d7/>



Advanced Microservices Deployment

