

Kubernetes Training

Wolfgang Ofner



kubernetes



Azure

About Me

Freelance Cloud Architect, Toronto, Canada

Focus on Azure, Kubernetes, DevOps, and .NET

<https://programmingwithwolfgang.com>

<https://www.linkedin.com/in/wolfgangofner>

[https://www.youtube.com/
@programmingwithwolfgang](https://www.youtube.com/@programmingwithwolfgang)



Examples and Code Files

[Examples and Code Files - GitHub](#)



Agenda

Day 1

- Docker
- Kubernetes Theory
- Kubernetes Exercises

Breaks can be taken at any time outside theory blocks

Questions and conversations are highly encouraged

Please leave your cameras on

Agenda

Day 2

- Kubernetes Theory and Exercises
- Helm
- Practice

Breaks can be taken at any time outside theory blocks

Questions and conversations are highly encouraged

Please leave your cameras on

Agenda

Day 3

- tbd

Agenda

Day 4

- tbd

Challenges of modern Software

Deploy 100 times a day

Versioning

Dependencies

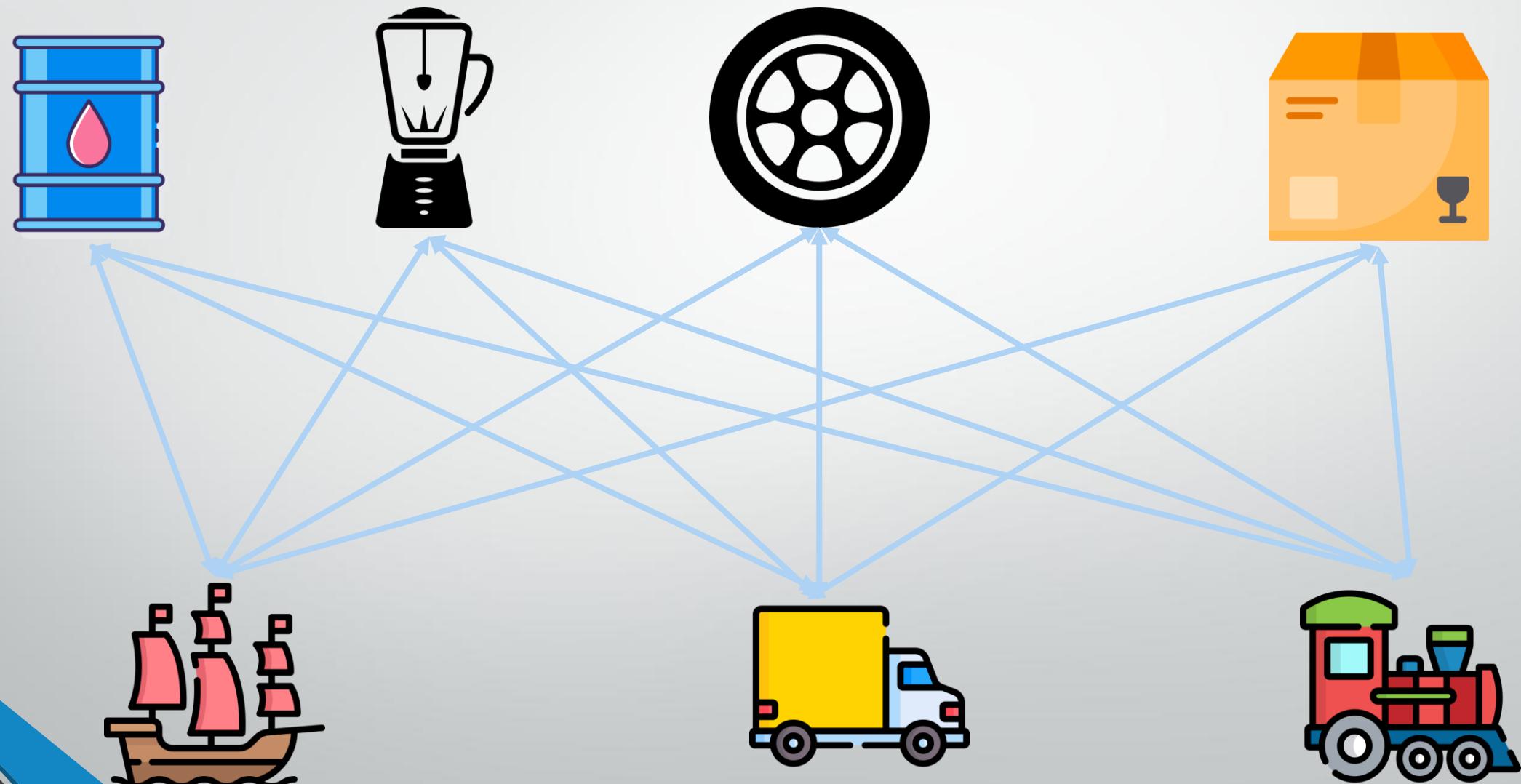
Easy to test

Fast to set up on target machine

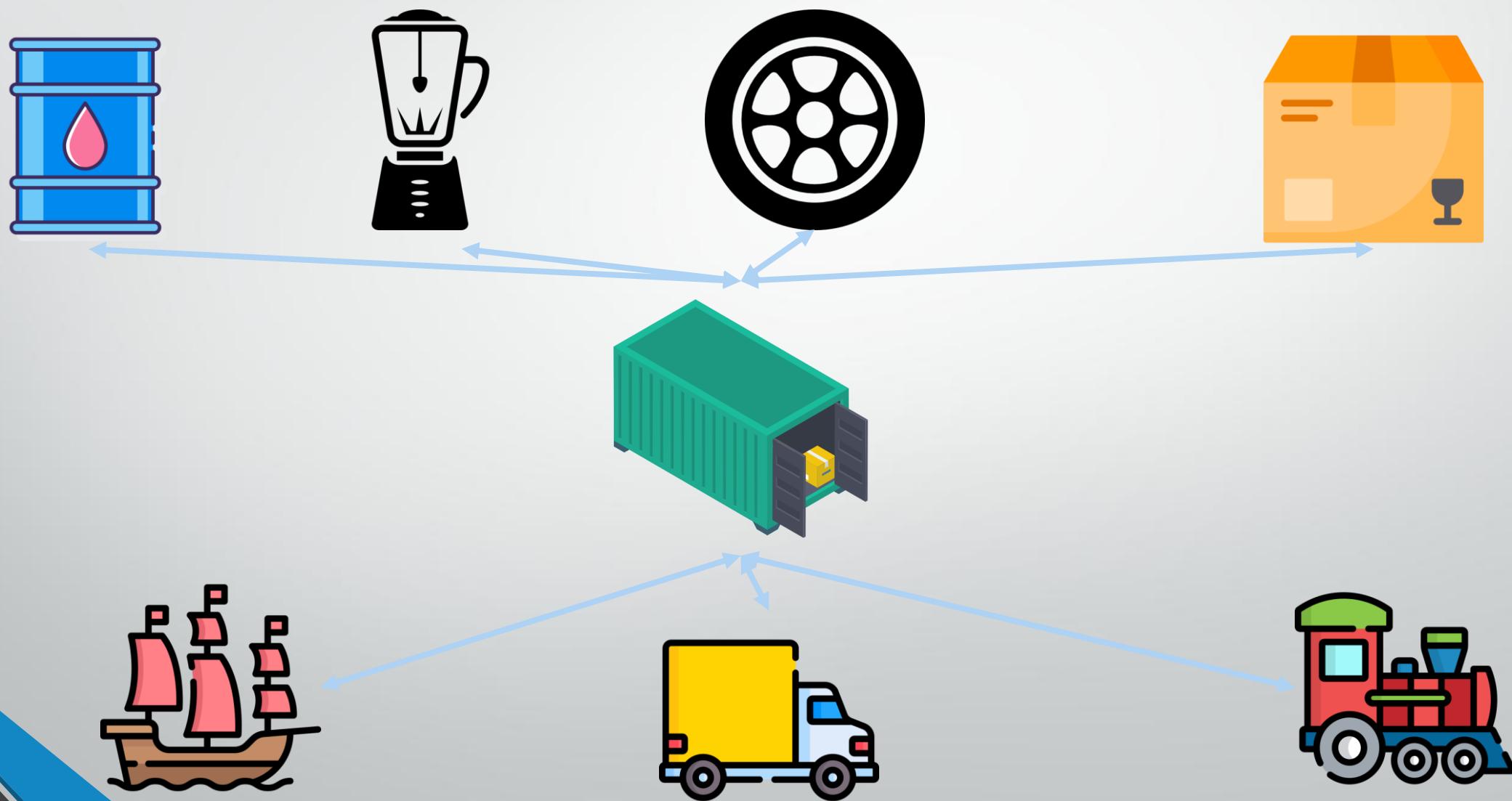
Monolithic software

Database deployments

Freighter transport before 1956



Invention of the Container (1956)





Container Solutions

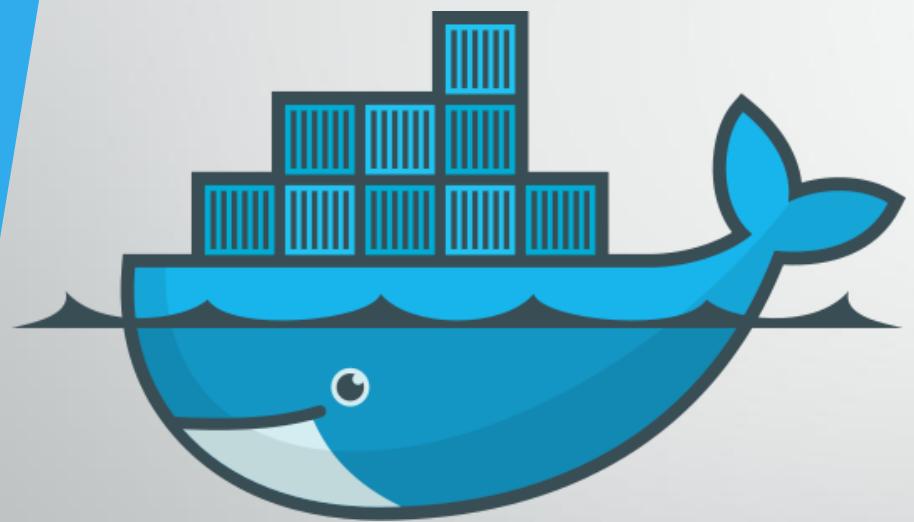
Docker

Podman

Lxc

Crio

Rancher Desktop



docker

Docker Containers

Dockerfile: blueprint

Container: Instance of this blueprint

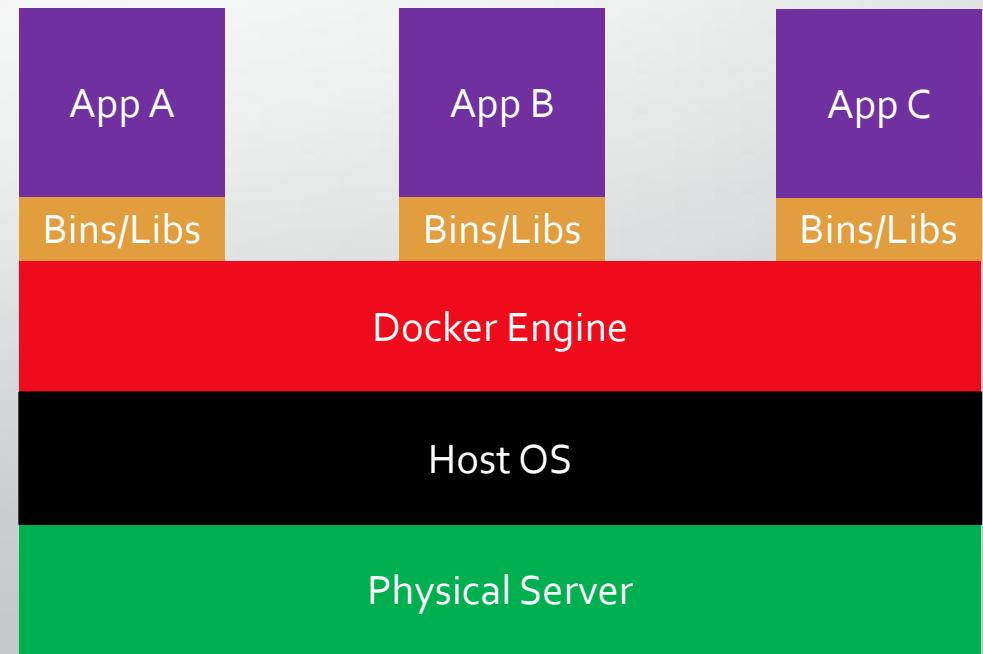
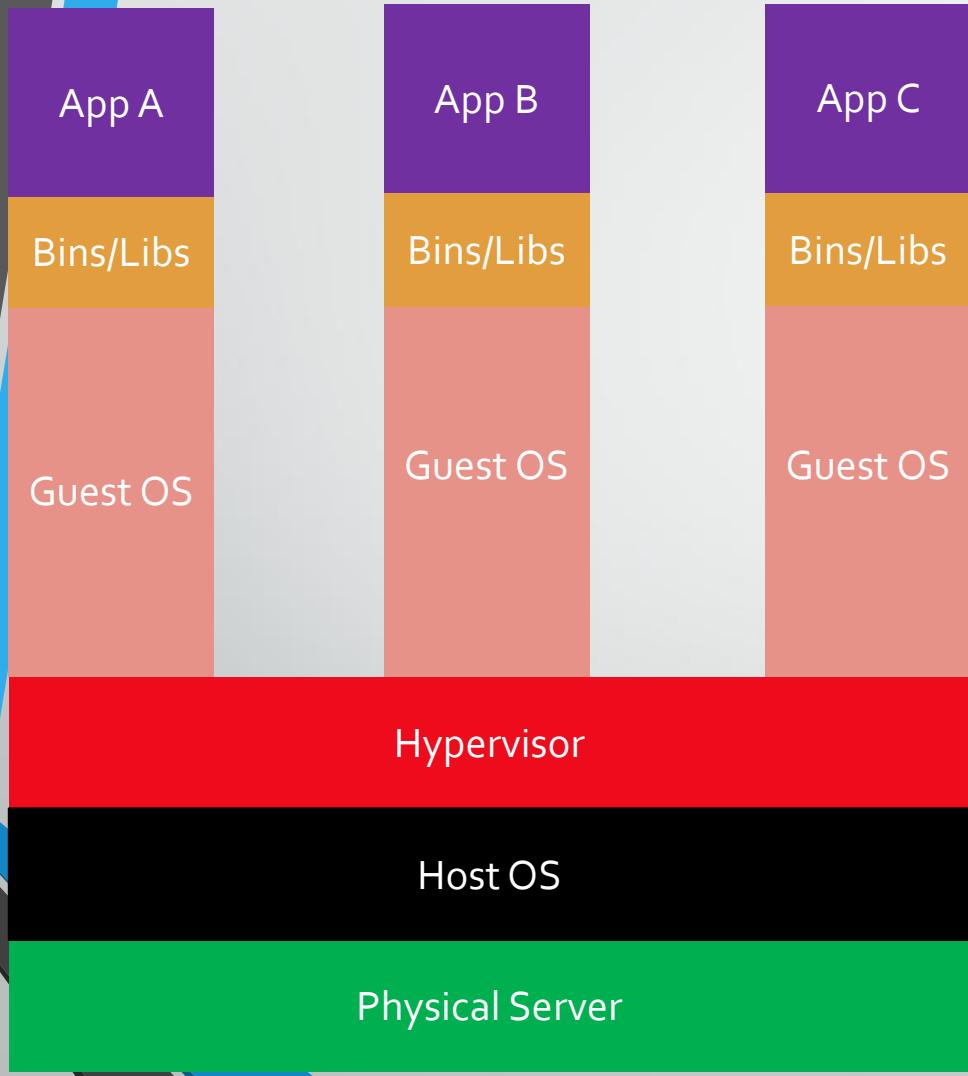
Versioned artifact

Container image is always bit by bit identical when deployed

Docker Containers

Images for different build platforms, e.g. x86, x64, ARM
OCI (Open Container Initiative) compliant
Abstracts underlying infrastructure
Fast start up times
Pet vs. Kettle

Virtual Machine vs. Container



Dockerfile

Blueprint to build Docker Image

Can be based on existing images

Commands to update the base OS and install additional software

Build artifacts to include, such as a developed application

Command to run when the container is launched

Dockerfile

```
FROM mcr.microsoft.com/dotnet/aspnet:5.0 AS base
WORKDIR /app
EXPOSE 80
EXPOSE 443

FROM mcr.microsoft.com/dotnet/sdk:5.0 AS build
WORKDIR /src
COPY ["CustomerApi/CustomerApi.csproj", "CustomerApi/"]
RUN dotnet restore "CustomerApi/CustomerApi.csproj"
COPY . .
WORKDIR "/src/CustomerApi"
RUN dotnet build "CustomerApi.csproj" -c Release -o /app/build

FROM build AS publish
RUN dotnet publish "CustomerApi.csproj" -c Release -o /app/publish

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "CustomerApi.dll"]
```

```
FROM mcr.microsoft.com/dotnet/aspnet:5.0 AS base
WORKDIR /app
EXPOSE 80
EXPOSE 443

FROM mcr.microsoft.com/dotnet/sdk:5.0 AS build
WORKDIR /src
COPY ["CustomerApi/CustomerApi.csproj", "CustomerApi/"]
COPY ["CustomerApi.Data/CustomerApi.Data.csproj", "CustomerApi.Data/"]
COPY ["CustomerApi.Domain/CustomerApi.Domain.csproj", "CustomerApi.Domain/"]
COPY ["CustomerApi.Service/CustomerApi.Service.csproj", "CustomerApi.Service/"]
COPY ["CustomerApi.Messaging.Send/CustomerApi.Messaging.Send.csproj", "CustomerApi.Messaging.Send/"]
COPY ["CustomerApi.Database.Build/CustomerApi.Database.Build.csproj", "CustomerApi.Database.Build/"]
COPY ["Tests/CustomerApi.Test/CustomerApi.Test.csproj", "Tests/CustomerApi.Test/"]
COPY ["Tests/CustomerApi.Service.Test/CustomerApi.Service.Test.csproj", "Tests/CustomerApi.Service.Test/"]
COPY ["Tests/CustomerApi.Data.Test/CustomerApi.Data.Test.csproj", "Tests/CustomerApi.Data.Test/"]
COPY ["CustomerApi/nuget.config", ""]
COPY [".props", "./"]

ARG PAT=localhost
RUN sed -i "s|</configuration>|<packageSourceCredentials><MicroserviceDemoNugets><add key=\"Username\" value=\"$PAT\" /><add key=\"ClearTextPassword\" value=\"$PAT\" /></MicroserviceDemoNugets></packageSourceCredentials>|g" ./nuget.config

RUN dotnet restore "CustomerApi/CustomerApi.csproj" --configfile "./nuget.config"
RUN dotnet restore "CustomerApi.Database.Build/CustomerApi.Database.Build.csproj" --configfile "./nuget.config"
RUN dotnet restore "Tests/CustomerApi.Test/CustomerApi.Test.csproj" --configfile "./nuget.config"
RUN dotnet restore "Tests/CustomerApi.Service.Test/CustomerApi.Service.Test.csproj" --configfile "./nuget.config"
RUN dotnet restore "Tests/CustomerApi.Data.Test/CustomerApi.Data.Test.csproj" --configfile "./nuget.config"
COPY ..

RUN dotnet build "CustomerApi/CustomerApi.csproj" -c Release -o /app/build --no-restore
RUN dotnet build "Tests/CustomerApi.Test/CustomerApi.Test.csproj" -c Release --no-restore
RUN dotnet build "Tests/CustomerApi.Service.Test/CustomerApi.Service.Test.csproj" -c Release --no-restore
RUN dotnet build "Tests/CustomerApi.Data.Test/CustomerApi.Data.Test.csproj" -c Release --no-restore

FROM build AS dacpac
ARG BuildId=localhost
LABEL dacpac=${BuildId}
WORKDIR /src
RUN dotnet build "CustomerApi.Database.Build/CustomerApi.Database.Build.csproj" -c Release -o /dacpacs --no-restore

FROM build AS test
ARG BuildId=localhost
LABEL test=${BuildId}
RUN dotnet test --no-build -c Release --results-directory /testresults --logger "trx;LogFileName=test_results.trx" /p:CollectCoverage=true /p:CoverletOutputFormat=json%2cCobertura /p:CoverletOutputFormat=json%2cCobertura
RUN dotnet test --no-build -c Release --results-directory /testresults --logger "trx;LogFileName=test_results2.trx" /p:CollectCoverage=true /p:CoverletOutputFormat=json%2cCobertura /p:CoverletOutputFormat=json%2cCobertura
RUN dotnet test --no-build -c Release --results-directory /testresults --logger "trx;LogFileName=test_results3.trx" /p:CollectCoverage=true /p:CoverletOutputFormat=json%2cCobertura /p:CoverletOutputFormat=json%2cCobertura

FROM build AS publish
RUN dotnet publish "CustomerApi/CustomerApi.csproj" --no-restore -c Release -o /app/publish

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "CustomerApi.dll"]
```

```
FROM ubuntu:20.04
RUN DEBIAN_FRONTEND=noninteractive apt-get update
RUN DEBIAN_FRONTEND=noninteractive apt-get upgrade -y

RUN DEBIAN_FRONTEND=noninteractive apt-get install -y -qq --no-install-recommends \
    apt-transport-https \
    apt-utils \
    ca-certificates \
    curl \
    git \
    iputils-ping \
    jq \
    lsb-release \
    software-properties-common \
    wget

RUN curl -sL https://aka.ms/InstallAzureCLIDeb | bash

RUN wget https://packages.microsoft.com/config/ubuntu/20.04/packages-microsoft-prod.deb -O packages-microsoft-prod.deb
RUN dpkg -i packages-microsoft-prod.deb
RUN rm packages-microsoft-prod.deb
RUN echo 'deb http://download.opensuse.org/repositories/devel:/kubic:/libcontainers:/stable/xUbuntu_20.04/ /' | tee /etc/apt/sources.list.d/kubic-libcontainers-stable.list
RUN curl -fsSL https://download.opensuse.org/repositories/devel:kubic:libcontainers:stable/xUbuntu_20.04/Release.key | apt-key add -
RUN apt-get update
RUN apt-get install -y dotnet-sdk-6.0
RUN apt-get install -y dotnet-sdk-7.0
RUN apt -y install podman fuse-overlayfs
```

Docker Demo

Onion File System

Every command
is a new layer

Layers can be
cached

Faster builds

11 Layer

```
FROM mcr.microsoft.com/dotnet/aspnet:5.0 AS base
WORKDIR /app
EXPOSE 80
EXPOSE 443

FROM mcr.microsoft.com/dotnet/sdk:5.0 AS build
WORKDIR /src
COPY ["CustomerApi/CustomerApi.csproj", "CustomerApi/"]
RUN dotnet restore "CustomerApi/CustomerApi.csproj"
COPY . .
WORKDIR "/src/CustomerApi"
RUN dotnet build "CustomerApi.csproj" -c Release -o /app/build

FROM build AS publish
RUN dotnet publish "CustomerApi.csproj" -c Release -o /app/publish

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "CustomerApi.dll"]
```

57 Layer

```
FROM mcr.microsoft.com/dotnet/aspnet:5.0 AS base
WORKDIR /app
EXPOSE 80
EXPOSE 443

FROM mcr.microsoft.com/dotnet/sdk:5.0 AS build
WORKDIR /src
COPY ["CustomerApi/CustomerApi.csproj", "CustomerApi/"]
COPY ["CustomerApi.Data/CustomerApi.Data.csproj", "CustomerApi.Data/"]
COPY ["CustomerApi.Domain/CustomerApi.Domain.csproj", "CustomerApi.Domain/"]
COPY ["CustomerApi.Service/CustomerApi.Service.csproj", "CustomerApi.Service/"]
COPY ["CustomerApi.Messaging.Send/CustomerApi.Messaging.Send.csproj", "CustomerApi.Messaging.Send/"]
COPY ["CustomerApi.Database.Build/CustomerApi.Database.Build.csproj", "CustomerApi.Database.Build/"]
COPY ["Tests/CustomerApi.Test/CustomerApi.Test.csproj", "Tests/CustomerApi.Test/"]
COPY ["Tests/CustomerApi.Service.Test/CustomerApi.Service.Test.csproj", "Tests/CustomerApi.Service.Test/"]
COPY ["Tests/CustomerApi.Data.Test/CustomerApi.Data.Test.csproj", "Tests/CustomerApi.Data.Test/"]
COPY ["CustomerApi/nuget.config", ""]
COPY [".props", "./"]

ARG PAT=localhost
RUN sed -i "s|</configuration>|<packageSourceCredentials><MicroserviceDemoNugets><add key=\"Username\" value=\"$PAT\" /><add key=\"ClearTextPassword\" value=\"$${PAT}\" /></MicroserviceDemoNugets></packageSourceCredentials>" ./nuget.config

RUN dotnet restore "CustomerApi/CustomerApi.csproj" --configfile "./nuget.config"
RUN dotnet restore "CustomerApi.Database.Build/CustomerApi.Database.Build.csproj" --configfile "./nuget.config"
RUN dotnet restore "Tests/CustomerApi.Test/CustomerApi.Test.csproj" --configfile "./nuget.config"
RUN dotnet restore "Tests/CustomerApi.Service.Test/CustomerApi.Service.Test.csproj" --configfile "./nuget.config"
RUN dotnet restore "Tests/CustomerApi.Data.Test/CustomerApi.Data.Test.csproj" --configfile "./nuget.config"
COPY ..

RUN dotnet build "CustomerApi/CustomerApi.csproj" -c Release -o /app/build --no-restore
RUN dotnet build "Tests/CustomerApi.Test/CustomerApi.Test.csproj" -c Release --no-restore
RUN dotnet build "Tests/CustomerApi.Service.Test/CustomerApi.Service.Test.csproj" -c Release --no-restore
RUN dotnet build "Tests/CustomerApi.Data.Test/CustomerApi.Data.Test.csproj" -c Release --no-restore

FROM build AS dacpac
ARG BuildId=localhost
LABEL dacpac=${BuildId}
WORKDIR /src
RUN dotnet build "CustomerApi.Database.Build/CustomerApi.Database.Build.csproj" -c Release -o /dacpacs --no-restore

FROM build AS test
ARG BuildId=localhost
LABEL test=${BuildId}
RUN dotnet test --no-build -c Release --results-directory /testresults --logger "trx;LogFileName=test_results.trx" /p:CollectCoverage=true /p:CoverletOutputFormat=json%2cCobertura /p:CoverletOutputFormat=json%2cCobertura
RUN dotnet test --no-build -c Release --results-directory /testresults --logger "trx;LogFileName=test_results2.trx" /p:CollectCoverage=true /p:CoverletOutputFormat=json%2cCobertura /p:CoverletOutputFormat=json%2cCobertura
RUN dotnet test --no-build -c Release --results-directory /testresults --logger "trx;LogFileName=test_results3.trx" /p:CollectCoverage=true /p:CoverletOutputFormat=json%2cCobertura /p:CoverletOutputFormat=json%2cCobertura

FROM build AS publish
RUN dotnet publish "CustomerApi/CustomerApi.csproj" --no-restore -c Release -o /app/publish

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "CustomerApi.dll"]
```

Inspect Layers

Docker history <IMAGE_ID>

Dive: <https://github.com/wagoodman/dive>

1 / 1 ▾ + ⌂ ⌂

Tilix: Default

🔍 ⌂ ✕

1: Terminal ▾

[wagoodman@kiwi dive] ↵ master \$ dive someproj:latest

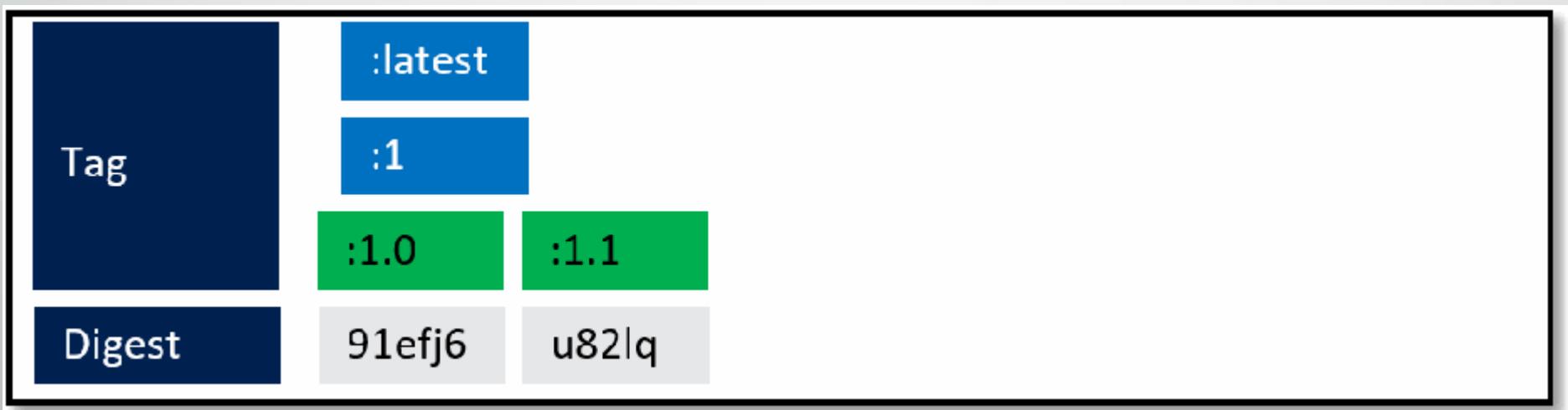
□ ×

Tags

Decide what version you run at any given time

“Latest” by default

Used for versioning



Container Registry

Repository to store container images

Docker Hub

Filters

1 - 15 of 15 results for **wolfgangofner**.

Best Match ▾

Products

- Images
- Extensions
- Plugins

Trusted Content

- Docker Official Image ⓘ
- Verified Publisher ⓘ
- Sponsored OSS ⓘ

Operating Systems

- Linux
- Windows

Architectures

- ARM
- ARM 64



wolfgangofner/microservicedemo ·  100K+ · ⭐ 0
By [wolfgangofner](#) · Updated 3 years ago
Linux x86-64



wolfgangofner/customerapi ·  3.3K · ⭐ 0
By [wolfgangofner](#) · Updated 17 days ago
Image for my NET 6 Microservice demo.
Linux x86-64



wolfgangofner/kubernetesdeploymentdemo ·  2.1K · ⭐ 0
By [wolfgangofner](#) · Updated 4 years ago
Linux x86-64



wolfgangofner/orderapi ·  1.5K · ⭐ 0

Container Registry

Repository to store container images

Docker Hub

Public vs. private registry

Azure Container Registry (ACR)

Additional functionalities like:

- Geo-replication
- Retention Policies
- Security scanning

Docker Compose

YAML file

Define container
dependencies

Run all dependent containers

Docker Compose

Advantages

- Configure dependencies between containers
- Restart policy
- Easy to start

Docker Compose

Disadvantages

- Monitoring
- Load Balancing
- Deployment
- SSL Certificate

Docker Compose

YAML file

Define container dependencies

Run all dependent containers

Advantages

- Configure dependencies between containers
- Restart policy
- Easy to start

Disadvantages

- Monitoring
- Load Balancing
- Deployment
- SSL Certificate

```
version: "3.9"

services:
  wordpress:
    image: wordpress
    restart: always
    ports:
      - 8080:80
    environment:
      WORDPRESS_DB_HOST: db
      WORDPRESS_DB_USER: exampleuser
      WORDPRESS_DB_PASSWORD: examplepass
      WORDPRESS_DB_NAME: exempledb
    volumes:
      - wordpress:/var/www/html

  db:
    image: mysql:5.7
    restart: always
    environment:
      MYSQL_DATABASE: exempledb
      MYSQL_USER: exampleuser
      MYSQL_PASSWORD: examplepass
      MYSQL_RANDOM_ROOT_PASSWORD: '1'
    volumes:
      - db:/var/lib/mysql
```

Docker Recap

Small images

Fast start up and deployment

Reusable and portable

Immutable → “Works on my machine”

Containers allow you to run your software even if your infrastructure provider does not support it

Docker Command

List running containers

```
docker ps
```

List images

```
docker image ls
```

Download an image from a registry

```
docker pull wolfgangofner/customerapi
```

Build an image from a Dockerfile

```
docker build . [-f CustomerApi/Dockerfile]
```

Tag an image

```
docker tag customerapi wolfgangofner/customerapi
```

Push an image to a registry

```
docker push wolfgangofner/customerapi
```

Start a container

```
docker run -p 32789:80 -p 32788:443 wolfgangofner/customerapi
```

Docker Compose Demo

Exercise

Docker Exercise



- Run an image from Dockerhub
- Create a new application and build it in a Dockerfile
- Upload your image to Dockerhub
- Build and run a docker-compose file
- Use different docker commands to interact with images

containers *don't* solve *all* problems



Container Orchestrator

Multi-Node Management

Resource Management

Load balancing

Monitoring and Self Healing

Zero Downtime Deployments

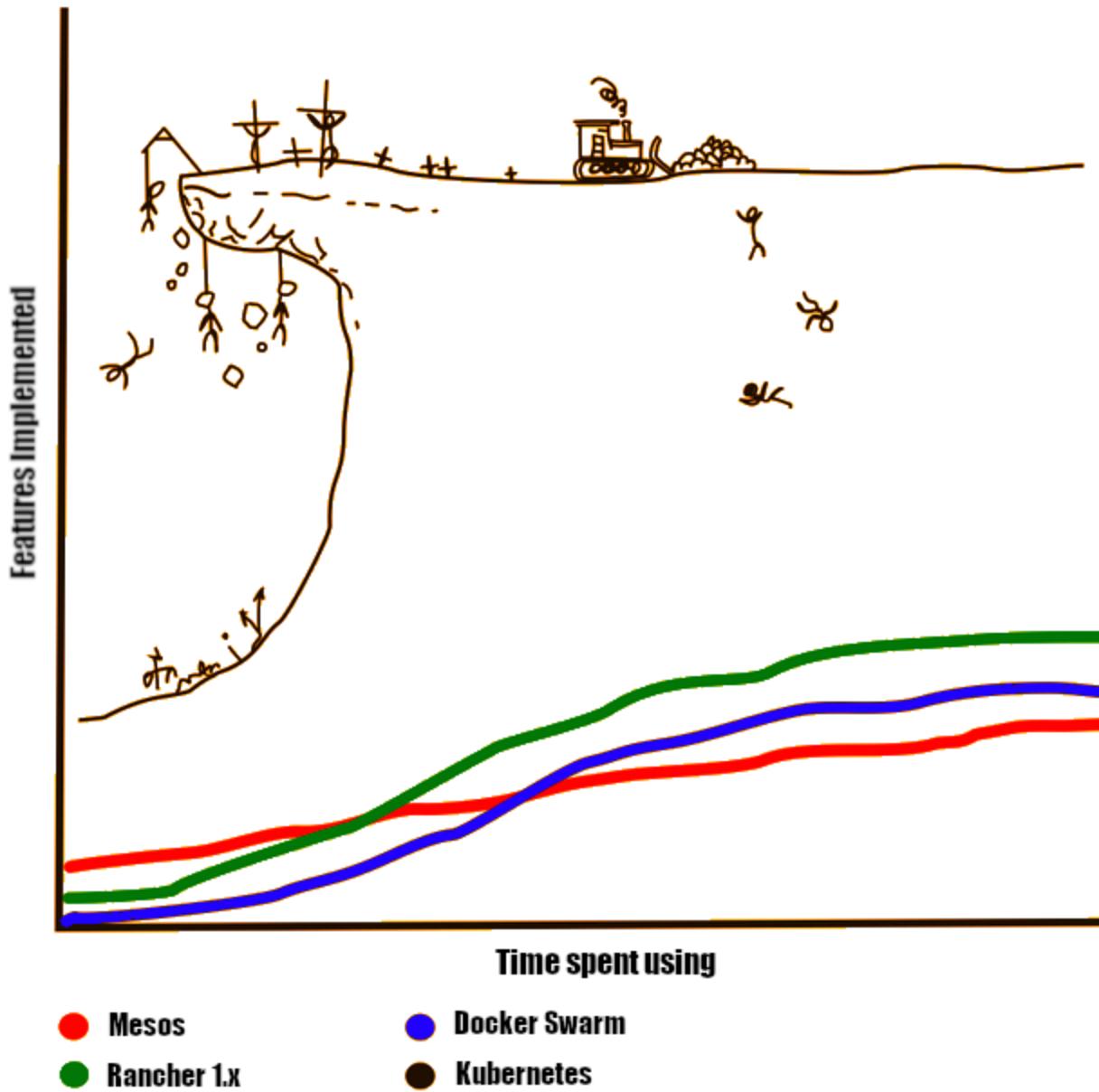
Manage TLS certificates



kubernetes

Container Orchestrator

Learning curves of some Container Orchestration Engines



Kubernetes

Kubernetes is an open-source system for automating computer application deployment, scaling, and **management of container applications**

First Release on 10 July 2015

Based on Google's Borg

Designed by Google and is now maintained by the Cloud Native Computing Foundation

Written in Go and open-source

“K8s” → K-8 character-s

Kubernetes

Multiple distributions

- K3s
- MicroK8s
- Kind
- Red Hat OpenShift
- Azure Kubernetes Service
- Amazon Elastic Kubernetes
- Google Kubernetes Engine

K8s locally

The screenshot shows the Docker Desktop application window. The left sidebar has a red box around the 'Kubernetes' item, which is currently selected. The main area displays a large blue Kubernetes logo and the text 'Enable Kubernetes'. Below it, a sub-instruction reads 'Start a Kubernetes single or multi-node cluster when starting Docker Desktop.' A prominent blue button labeled 'Create cluster' is also highlighted with a red box.

docker desktop PERSONAL

Ask Gordon BETA

Containers

Images

Volumes

Kubernetes Give feedback

Builds

Models

MCP Toolkit BETA

Docker Hub

Docker Scout

Extensions

Search Ctrl+K

?

!

!

W

RAM 0.75 GB CPU 0.25% Disk: 13.68 GB used (limit 1006.85 GB)

> v4.55.0

K8s locally

docker desktop PERSONAL

- Ask Gordon BETA
- Containers
- Images
- Volumes
- Kubernetes**
- Builds
- Models
- MCP Toolkit BETA
- Docker Hub
- Docker Scout
- Extensions

Create Kubernetes Cluster

Cluster Type

Kubeadm
Create a single-node cluster with kubeadm.
Version: v1.34.1

kind
Create a cluster containing one or more nodes with kind. Requires the [containerd image store](#)

Node(s): 1

Changing the number of nodes resets the cluster. All stacks and resources are deleted.

1 2 4 8 10

Version: 1.31.1

Changing the Kubernetes version resets your cluster. All stacks and resources are deleted.

Kubernetes version **1.31.1** ▾

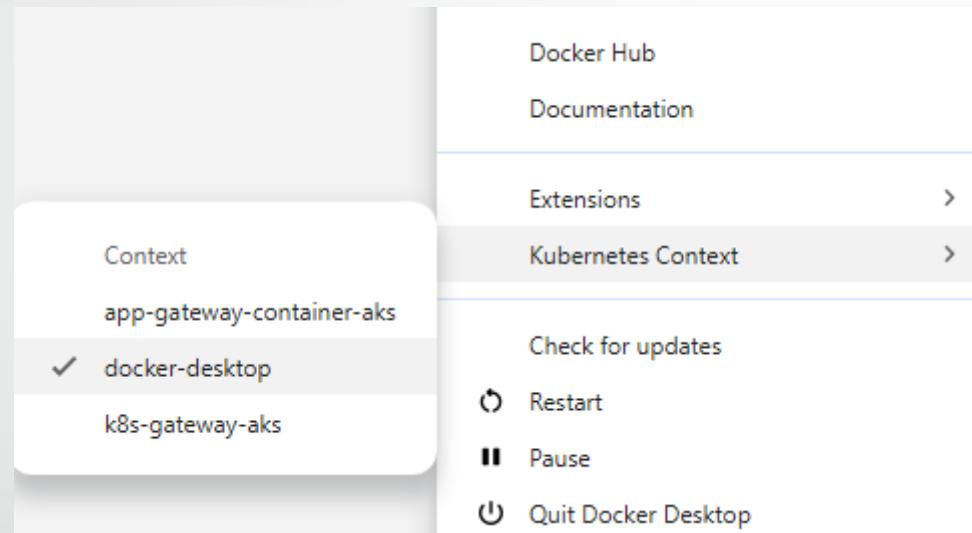
Advanced Settings

Show system containers (advanced)
Show Kubernetes internal containers when using Docker commands.

Cancel Create

Engine running | RAM 0.75 GB CPU 0.00% Disk: 13.68 GB used (limit 1006.85 GB) > ✓ v4.55.0

K8s locally



K8s locally

K3s

- Lightweight
- Great for edge and IoT
- Easy to install

```
curl -sfL https://get.k3s.io | sh -
```

Kubernetes Features

Self-healing

Service discovery and load balancing

Secret and configuration management

Horizontal scaling

Zero downtime deployments

Batch execution

Namespaces

Configuration in JSON or YAML



Self-healing Demo

Kubernetes Components

Master Node (Control Plane)

- kube-apiserver
- etcd
- kube-scheduler
- kube-control-manager
- Master Node is managed by cloud vendor

Worker Node

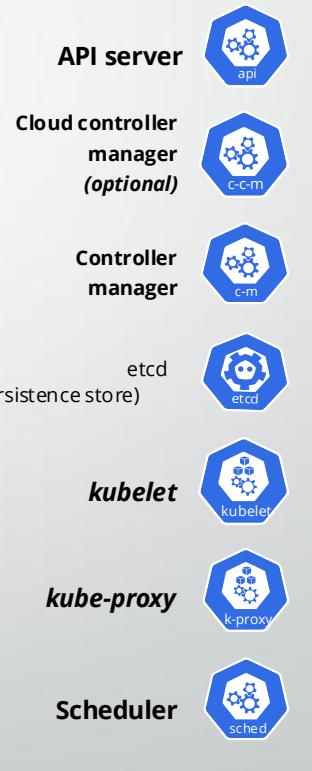
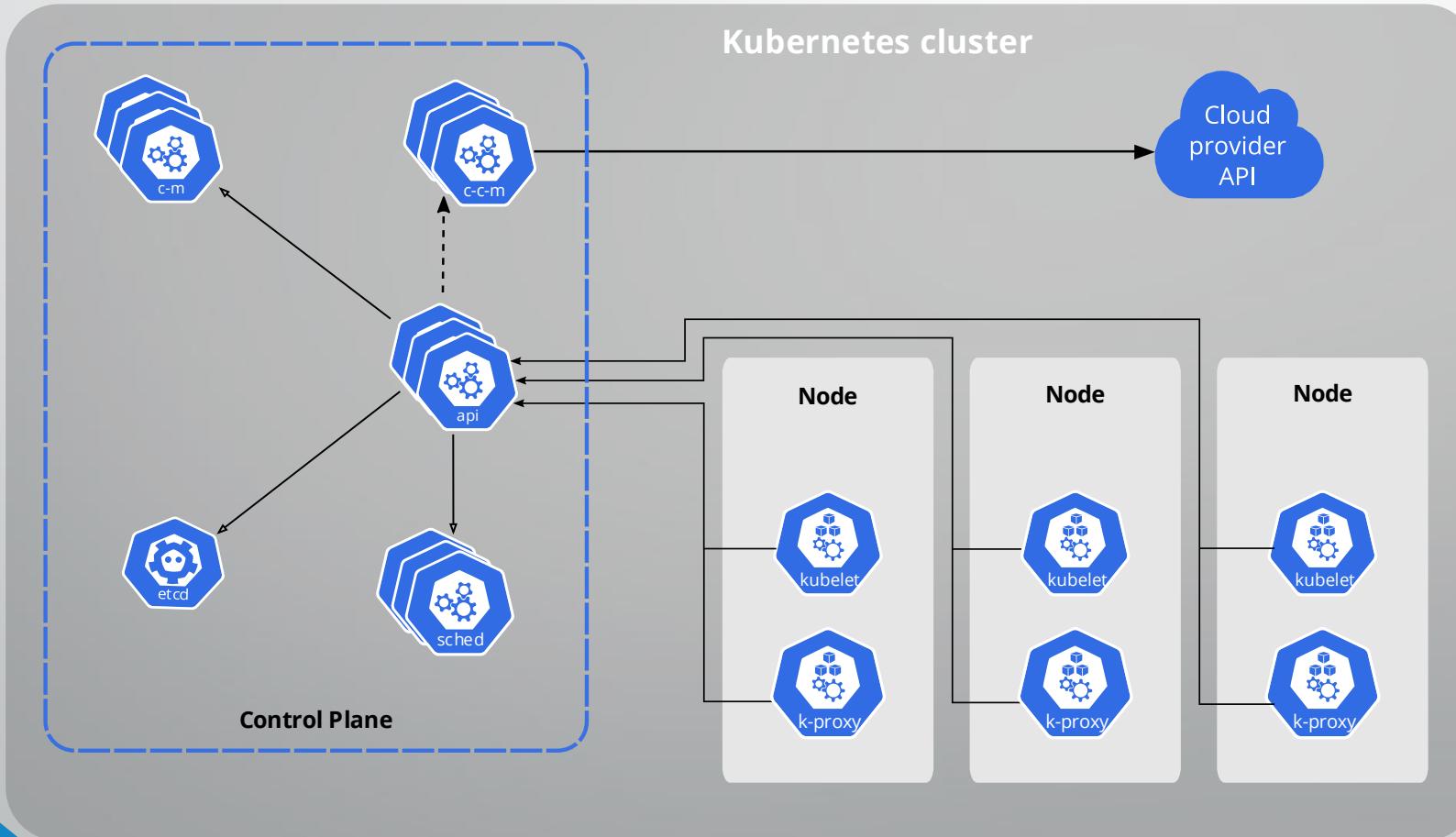
- kubelet
- kube-proxy
- Container runtime

Kubernetes Components

Addons

- DNS
- Networking
- Storage
- Dashboard
- ...

Kubernetes Components



Control plane -----

Node

53

Pod

A pod is the smallest unit in K8s

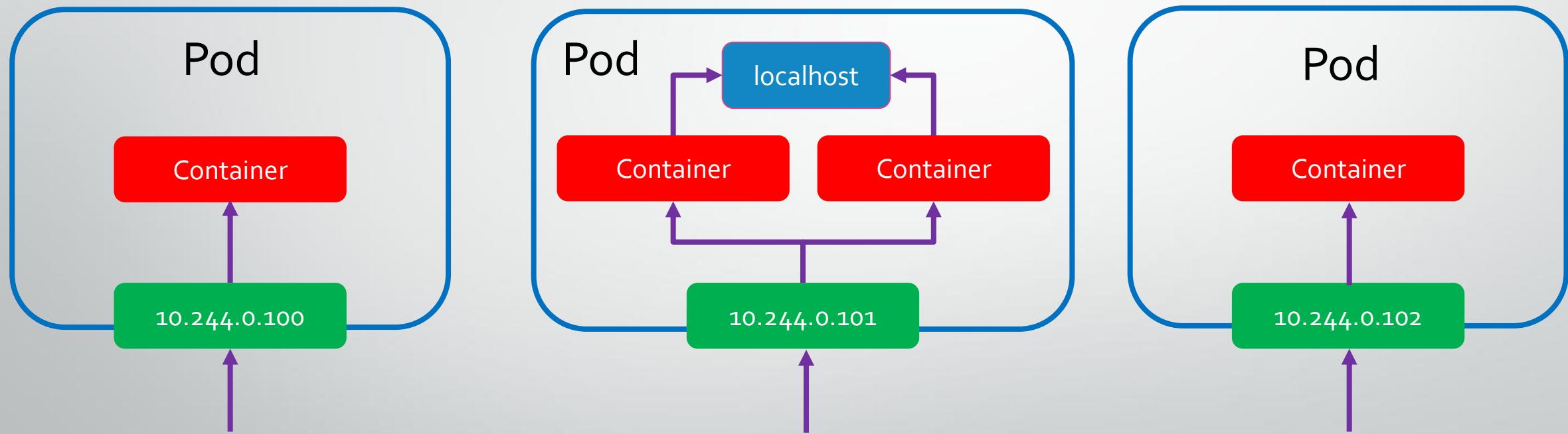
Pods wrap one or more containers

Provides a way to set environment variables and mount storage

Containers inside a pod can communicate via localhost

Multiple containers should only be combined in a pod if they are interdependent

Pods and Containers



Kubernetes Networking

The default network IP range for pods is `10.244.0.0/16`

Network CIDR range can be configured with addons:

- Cilium
- Flannel
- Calico

Namespaces

Used to create virtual cluster inside a physical cluster

- Isolation
- Resource segregation
- Multiple environments
- Resource quotas
- Access control

Kubernetes Configuration

Declarative Model and Desired State

- Tell Kubernetes what you want
- Kubernetes will figure out a way to the desired state
- etcd holds the current state component

Hey Kubernetes, run 3 pods of
wolfgangofner/customerapi



Let me check if I am already
running your pods



Currently there is 1 pod of
wolfgangofner/customerapi
running

Starting 2 more pods of
wolfgangofner/customerapi

Kubernetes Configuration

Declarative Model and Desired State

- Tell Kubernetes what you want
- Kubernetes will figure out a way to the desired state
- etcd holds the current state component

Configuration Handling

- YAML or JSON files
- Kubernetes CLI called kubectl
- kubectl communicates with Kubernetes API

Kubernetes Configuration

Declarative Model and Desired State

- Tell Kubernetes what you want
- Kubernetes will figure out a way to the desired state
- etcd holds the current state component

Configuration Handling

- YAML or JSON files
- Kubernetes CLI called kubectl
- kubectl communicates with Kubernetes API Server

kubectl



Kube Control

Kube Cuddle

YAML File

```
apiVersion: v1
kind: Service
metadata:
  name: kubernetesdemo-service
spec:
  type: LoadBalancer
  selector:
    app: kubernetesdemo
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: kubernetesdemo-deployment
  labels:
    app: kubernetesdemo
spec:
  replicas: 1
  selector:
    matchLabels:
      app: kubernetesdemo
  template:
    metadata:
      labels:
        app: kubernetesdemo
    spec:
      containers:
        - name: kubernetesdemo
          image: wolfgangfner/kubernetesdeploymentdemo:start
          ports:
            - containerPort: 80
```

Labels and Annotations

Labels

- Key valuepairs that are bound to objects like deployments or pods with a maximum of 63 character
- app:MyAppName
- Used to filter or select objects
- Can be changed or deleted at any times

Labels and Annotations

Annotations

- Also key value pairs but without the character limitation
- Can not be used for filtering or selecting objects

```
metadata:  
  creationTimestamp: "2021-10-17T11:58:22Z"  
labels:  
  component: apiserver  
  provider: kubernetes
```

Metadata
Age 51m
Labels component:apiserver provider:kubernetes

Services

Pods come and go

IP addresses will change

Services stay for the entire lifetime of the application

Persistent entry point

Fixed IP address

Load Balancing



Load Balancer Demo

Services

Pods come and go

IP addresses will change

Services stay for the entire lifetime of the application

Persistent entry point

Fixed IP address

Load Balancing

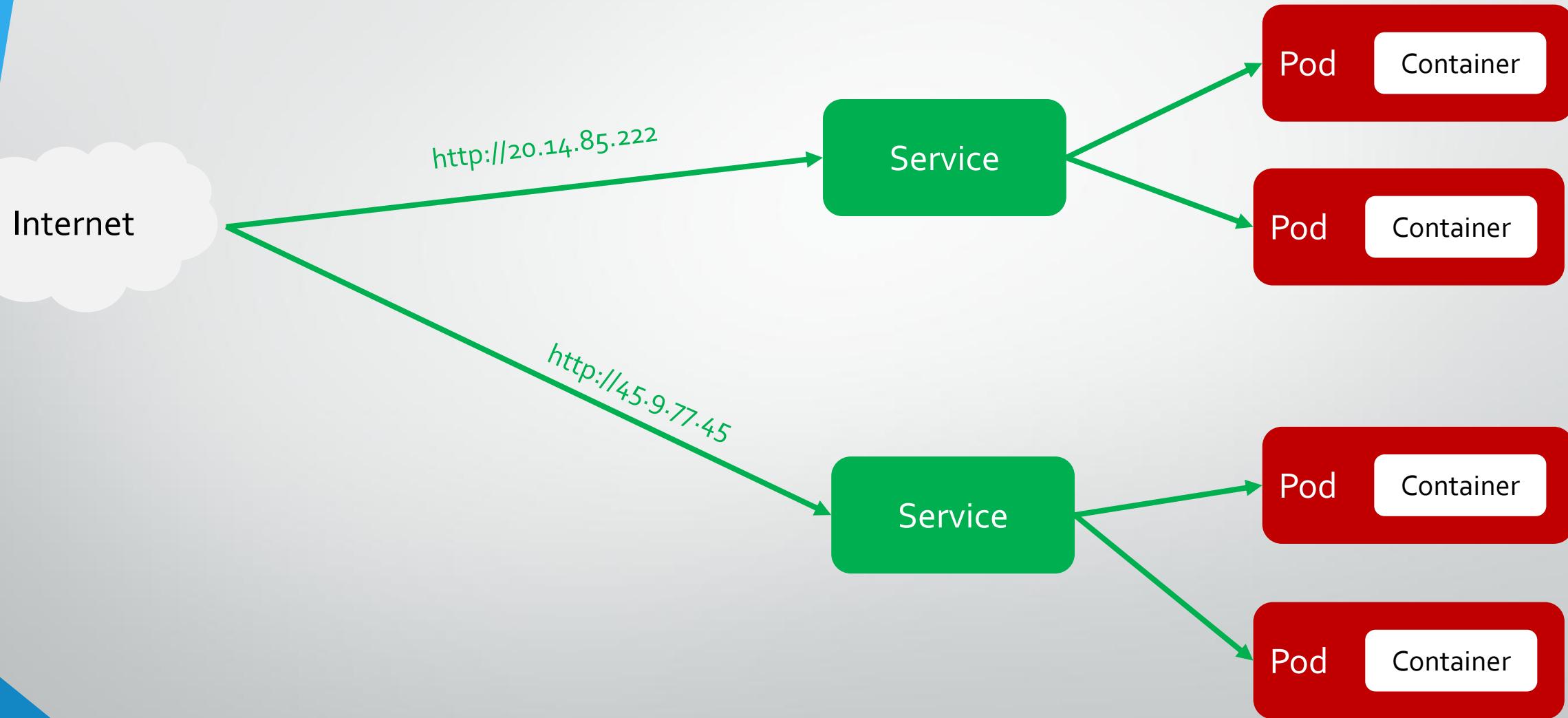
Pods and Services are matched using Labels

Services

Four types of Services:

- **LoadBalancer**: exposes the Service using an external load balancer
- **ClusterIP**: makes the Service accessible only from within the cluster
- **NodePort**: exposes the Service at each node's IP at a static port
- **ExternalName**: maps the service to an existing DNS FQDN

Service



Swagger UI x +

← → ⌂ Not secure | 20.103.233.94/index.html

{-} **swagger** Select a spec My API V1

kubernetesdemo-deployment-599d8f48c-gl4t9 v1

/swagger/v1/swagger.json

A collection of Web APIs

Values

GET /api/Values

POST /api/Values

GET /api/Values/{id}

PUT /api/Values/{id}

DELETE /api/Values/{id}

Kubectl Commands

Get resource

kubectl get pods/service/deployment

Delete resource

kubectl delete pod/service/deployment

Display information about resource

kubectl describe pod/node/service resource-name

Add/update new resource

kubectl apply --f myfile.yaml [namespace=my-namespace]

Set current namespace

kubectl config set-context --current --namespace=my-namespace

Kubernetes Cheat Sheet:

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

Exercise

Exercise

Connect to the Kubernetes cluster

Create a new namespace

- `kubectl create ns my-namespace`

Apply provided YAML file to your namespace

- `kubectl apply -f filepath --namespace=my-namespace`
- `kubectl get all -n my-namespace`

Set Replicas to 3 and check what happens

Replace image in Deployment with your Docker image

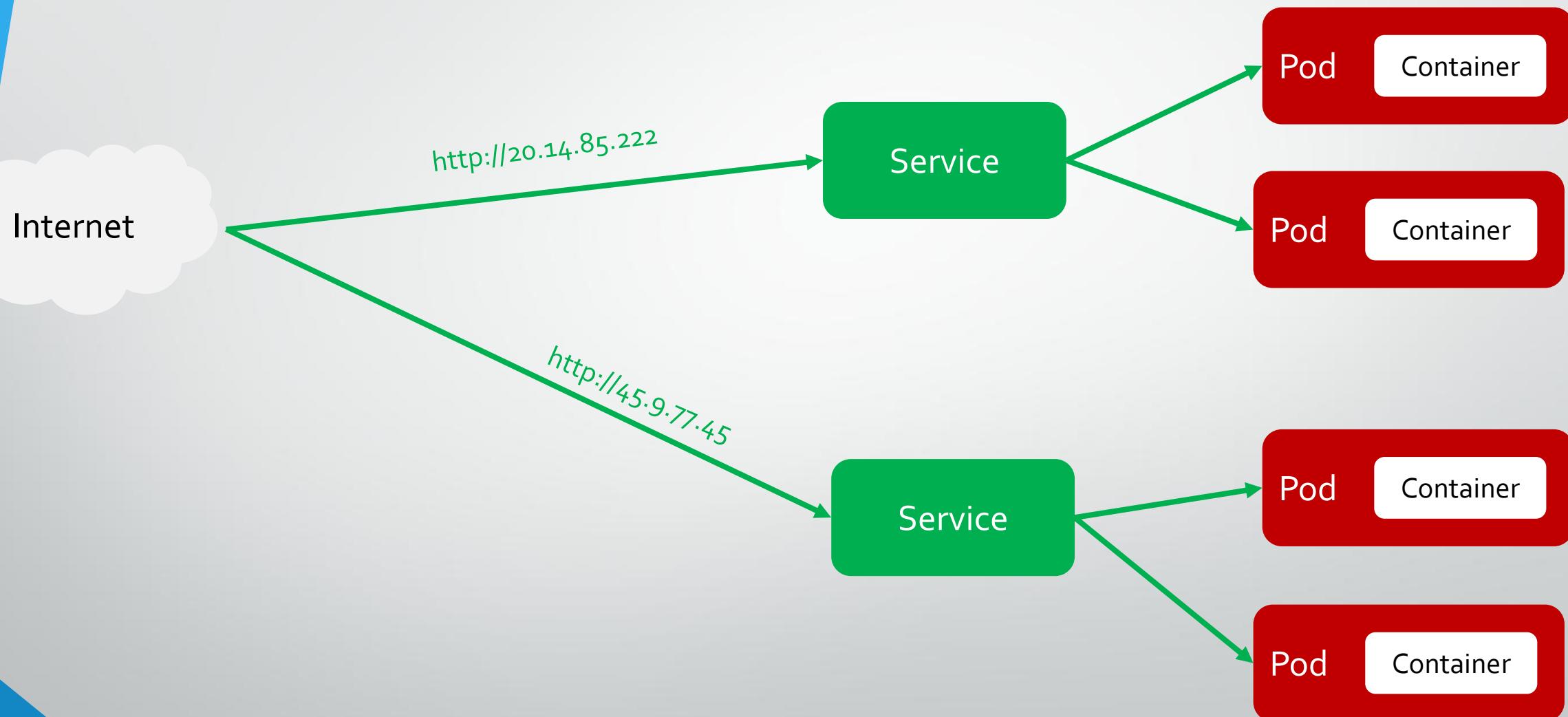
Recap

Containers are immutable objects

Pods run containers in Kubernetes

Services are an entry point and route requests to pods

Service



kubernetesdemo-deployment-599d8f48c-gl4t9 v1

/swagger/v1/swagger.json

A collection of Web APIs

Values

GET /api/Values

POST /api/Values

GET /api/Values/{id}

PUT /api/Values/{id}

DELETE /api/Values/{id}

Recap

Needed for production environment:

- Secret Management
- Health Checks
- TLS Certificates
- Deployments
- Resource Management

Secrets

Base64 encoded

Automatically decrypted when attached to a pod

Can be used in config file or environment variable

[Config and Storage](#) > [Secrets](#) > kedademoapi-tls

kedademoapi-tls

Summary

Metadata

Resource Viewer

YAML

```
1  ---
2  apiVersion: v1
3  data:
4    |   tls.crt: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tck1JSUZXVENDQkVHZ0F3SUJBZ01T
5    |   tls.key: LS0tLS1CRUdJTiBSU0EgUFJJVkJURSBLRVktLS0tLQpNSU1Fb2dJQkFBS0NBUVB
6  kind: Secret
```

Cert-Issuer

Kubernetes resource

Handles certificate requests

Supported issuers

- Let's Encrypt
- HashiCorp Vault
- Venafi
- Private PKI (Public Key Infrastructure)

```
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: letsencrypt
spec:
  acme:
    server: https://acme-v02.api.letsencrypt.org/directory
    email: <Your Email>
    privateKeySecretRef:
      name: letsencrypt
    solvers:
    - http01:
        ingress:
          class: nginx
        podTemplate:
          spec:
            nodeSelector:
              "kubernetes.io/os": linux
```

Cert-Manager

Manages obtaining and renewing of certificates

Can use variety of CAs like Let's Encrypt, HashiCorp Vault, and Venafi

Updates certificates at a configured time before expiry

Uses Cert Issuer to issue certificates

Cert-Manager

Issuers

letsencrypt

venafi-tpp

hashicorp-vault

Cert-Manager

Certificates

example.com
Issuer: letsencrypt

foo.bar.com
Issuer: hashicorp-vault

Kubernetes
Secrets

Signed keypair

Signed keypair

kedademoapi-tls

[Summary](#)[Metadata](#)[Resource Viewer](#)[YAML](#)

```
1  ---
2  apiVersion: v1
3  data:
4    tls.crt: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUZXVENDQkVHZ0F3SUJBZ01TQS
5    tls.key: LS0tLS1CRUdJTiBSU0EgUFJJVkJURSBLRVktLS0tLQpNSU1Fb2dJQkFBS0NBUVBNG
6  kind: Secret
7  metadata:
8    annotations:
9      cert-manager.io/alt-names: test.kedademo.programmingwithwolfgang.com
10     cert-manager.io/certificate-name: kedademoapi-tls
11     cert-manager.io/common-name: test.kedademo.programmingwithwolfgang.com
12     cert-manager.io/ip-sans: ""
13     cert-manager.io/issuer-group: cert-manager.io
14     cert-manager.io/issuer-kind: ClusterIssuer
15     cert-manager.io/issuer-name: letsencrypt
16     cert-manager.io/uri-sans: ""
17   creationTimestamp: "2021-10-17T12:07:46Z"
```



Connection is secure

Your information (for example, passwords or credit card numbers) is private when it is sent to this site.

[Learn more](#)

Select a definition

Customer API V1

Cu

/swagg

Certificate (Valid)

A simp

Cookies (1 in use)

Wolfga
Send e

Site settings

Customer

GET**/v1/Customer** Action to see all existing customers.**POST****/v1/Customer** Action to create a new customer in the database.**PUT****/v1/Customer** Action to update an existing customer

Ingress Controller

Entry point into the cluster

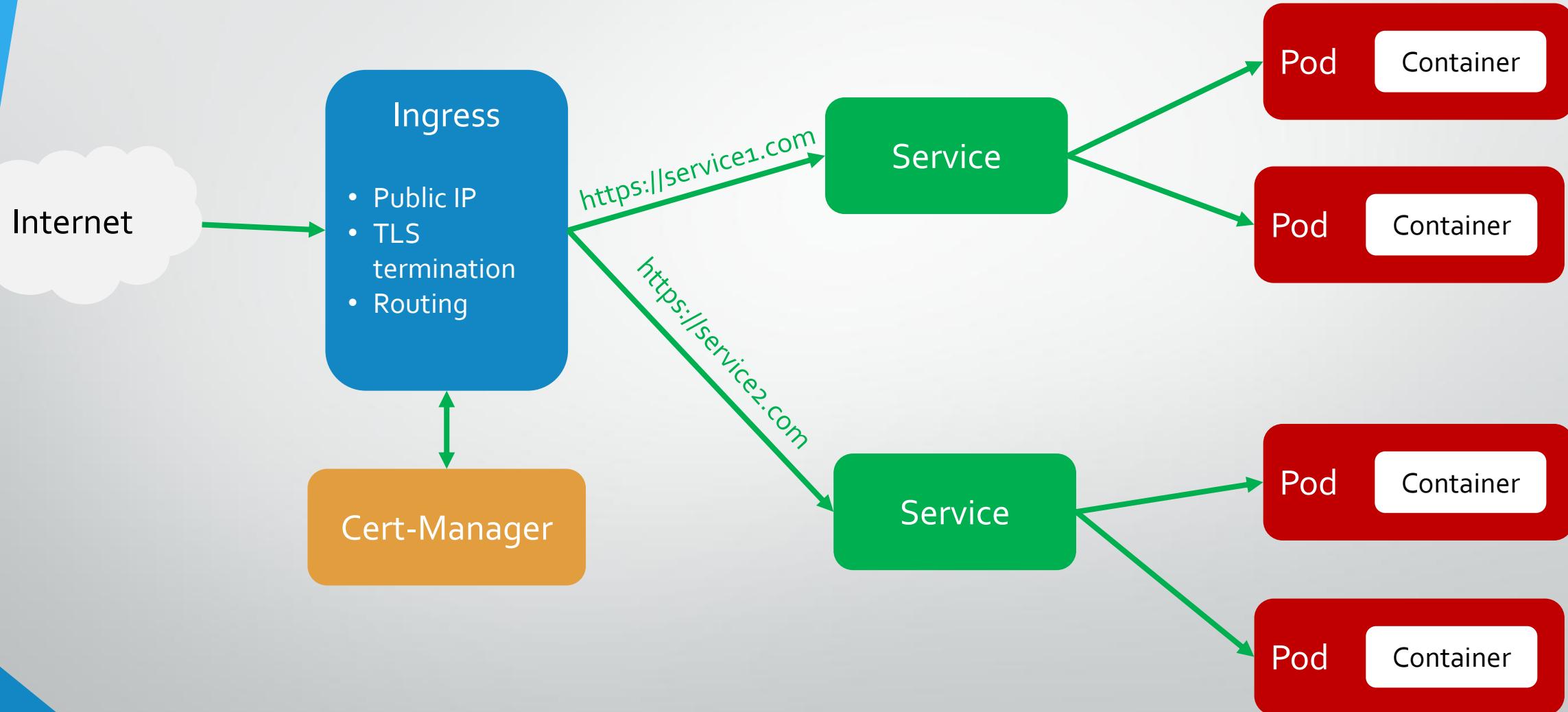
Processes HTTPS traffic

Reverse proxy redirects requests to Application

Popular Ingress Controller:

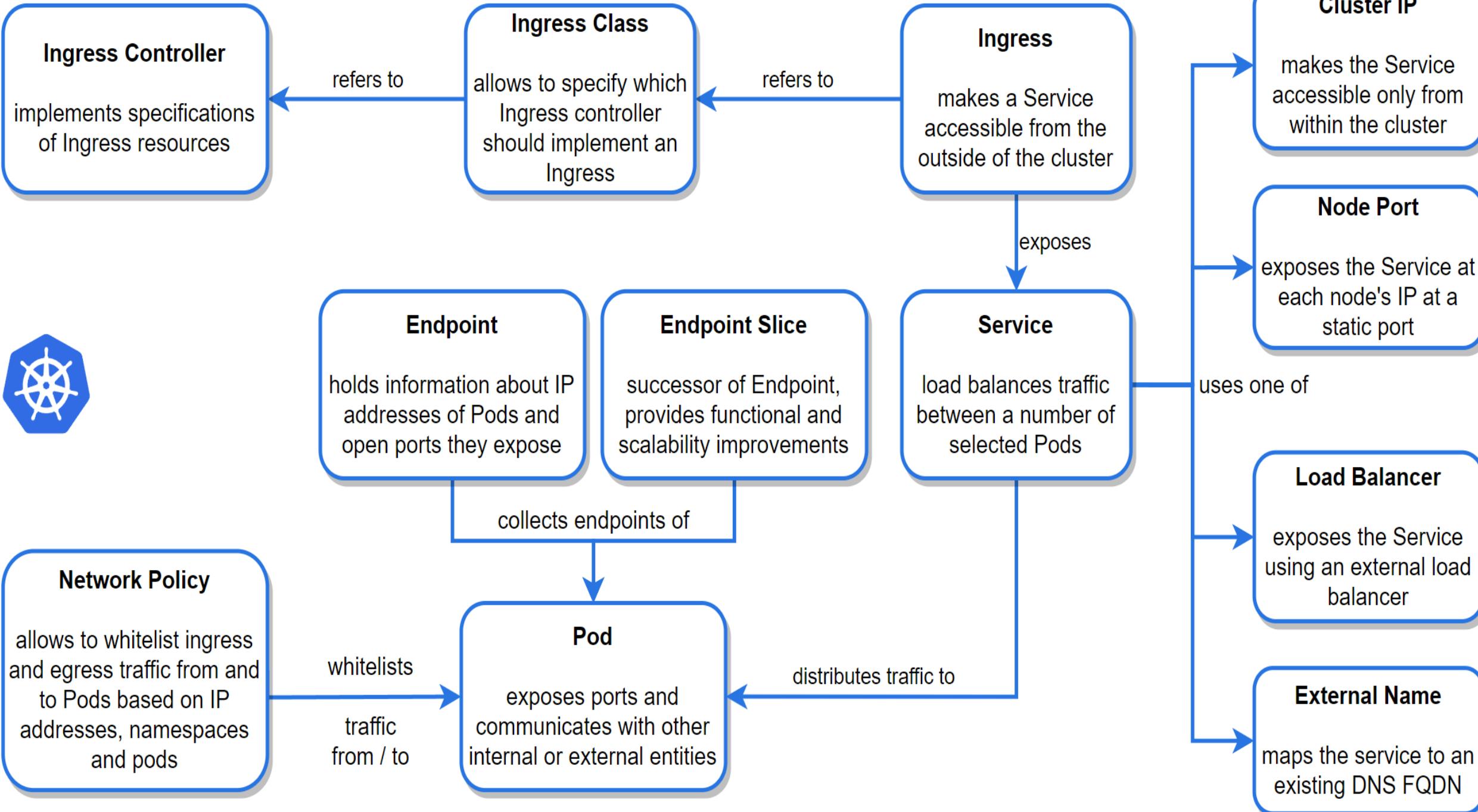
- NGINX
- Traefik
- HAProxy

Ingress Controller





Pull Request Demo



Pod Deployment

Pods are not directly deployed

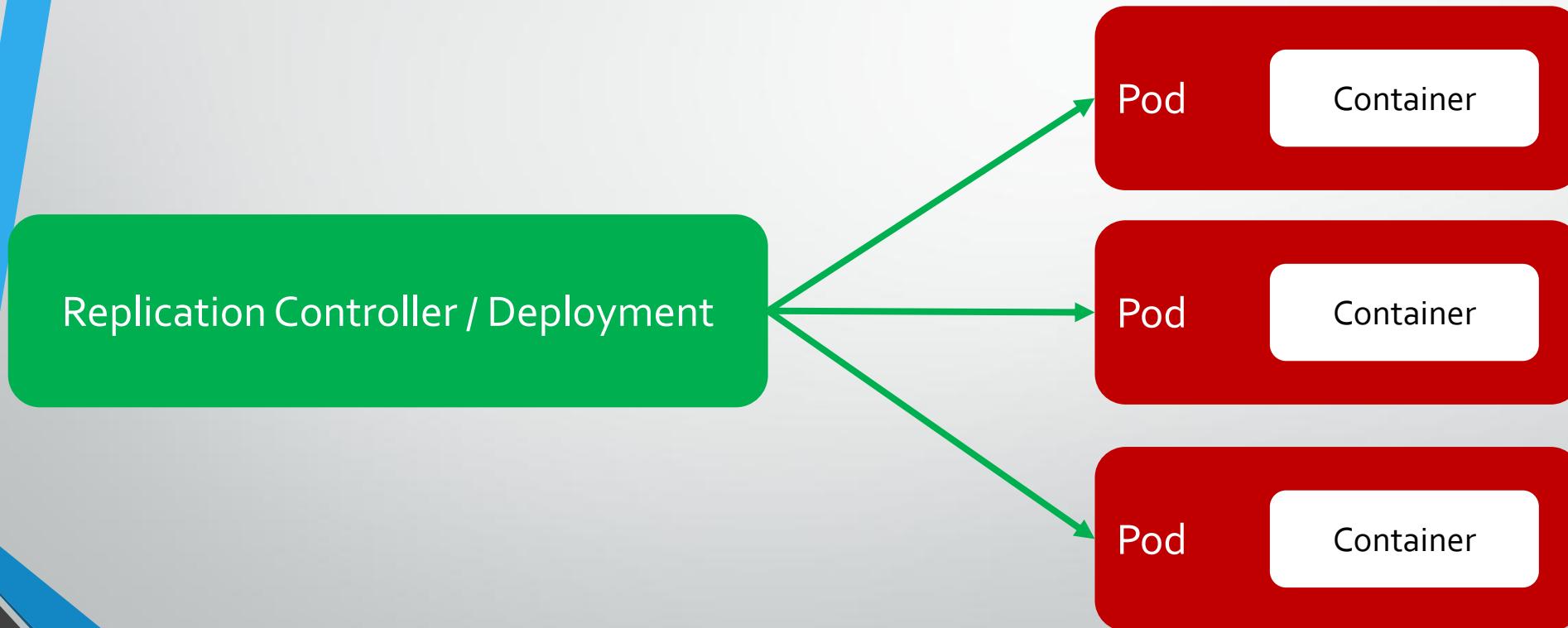
Deployment / Replicate Set create pods



Pod Deployment

Pods are not directly deployed

Deployment / Replicate Set create pods



Pod Deployment

Pods are not directly deployed

Deployment / Replicate Set create pods

Deployments manage ReplicaSets

Manages stateless applications

DaemonSet, CronJob, StatefulSet

Alternatively to Deployments, pods can be run using DaemonSets, CronJobs, and StatefulSets

CronJobs can be scheduled to start pods

StatefulSets manage stateful applications

DaemonSet, CronJob, StatefulSet

DeamonSets run pods on every node in the cluster

- Logging
- Monitoring
- Backup
- Reports
- Automated testing
- Databases

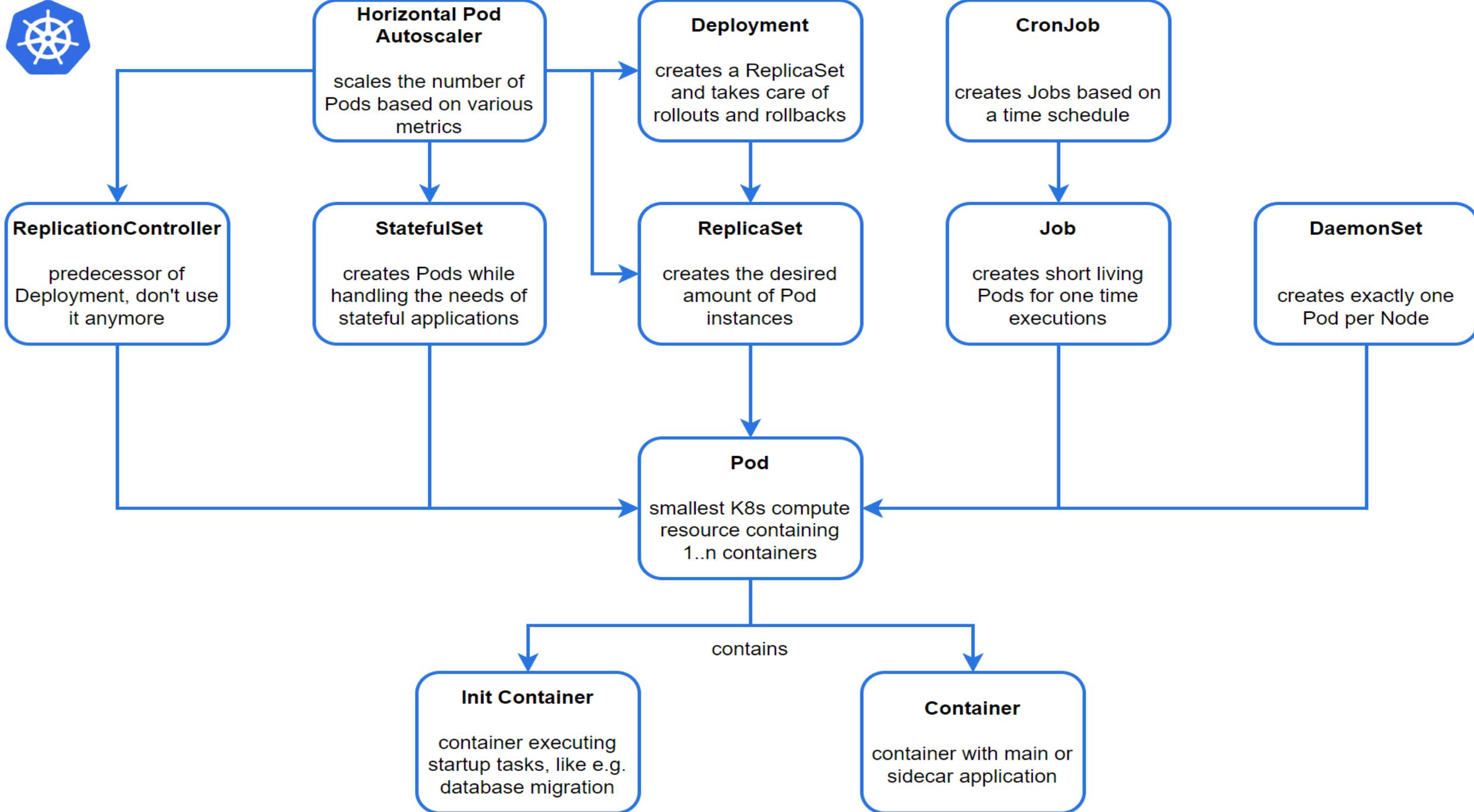
Horizontal Pod Autoscaler (HPA)

Queries resource utilization, e.g. CPU and RAM usage

Instructs ReplicationSet to scale out or scale in

Configures minimum and maximum number of pods

```
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: customerapi
spec:
  maxReplicas: 10
  minReplicas: 1
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: customerapi
  targetCPUUtilizationPercentage: 50
```





Auto Scaling Demo

Docker Health Checks

Docker provides a simple health check

--interval=DURATION

--timeout=DURATION

--start-period=DURATION

--retries should be set to N

Docker Health Checks

Problems with simple health checks

- Application startup may be longer than expected
- Different startup and health checks
- Specific port for checks

Liveness Probe

Checks if pod is alive

Sends HTTP request to check pod

Alive if answer \geq HTTP 200 & $<$ HTTP 400

- Pod will be restarted if dead

Configuration is part of the Deployment

```
livenessProbe:  
  httpGet:  
    path: /health  
    port: http  
  initialDelaySeconds: 15
```

Readiness Probe

Checks if pod is ready to receive traffic

Sends HTTP request to check pod

Alive if answer \geq HTTP 200 & $<$ HTTP 400

- Traffic will be routed to the pod when ready

Configuration is part of the Deployment

```
readinessProbe:  
  httpGet:  
    path: /health  
    port: http  
  initialDelaySeconds: 15
```

Resource Requests & Resource Limits

1000 Millicores = 1 Core

Memory is defined in bytes

Mebibyte = ~1MB

Configured in Deployment

Resource Requests & Resource Limits

Resource Requests

- Describe how many free resources a node has to have
- CPU and/or RAM

Resource Limits

- Maximum resources a pod is allowed to use
- Pod gets throttled when it uses too many resources
- If throttling is not successful → pod will be evicted
- CPU and/or RAM

```
resources:  
  limits:  
    cpu: 0.3  
    memory: 128Mi  
  requests:  
    cpu: 100m  
    memory: 64Mi
```

Resource Quotas

Limit resource usage within a namespace

Configured in the ResourceQuota object

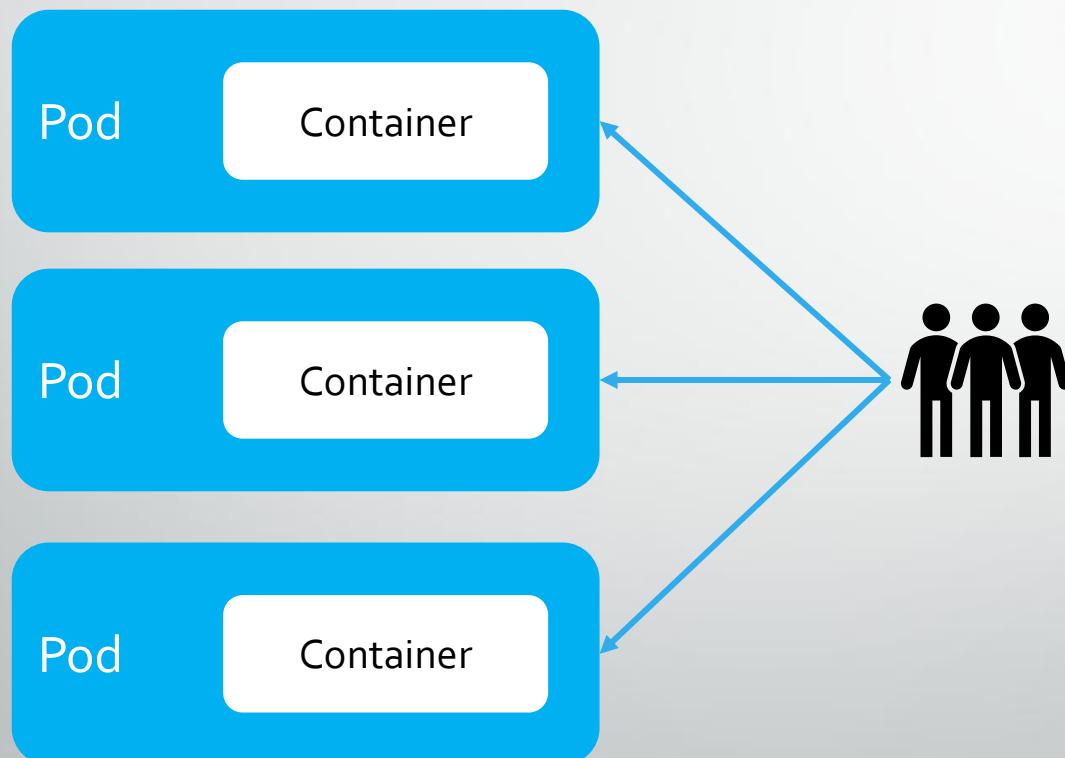
Restrict the following resources

- Number of pods
- CPU and RAM per request
- Total CPU and RAM usage

Deployment Strategies

Blue Green deployment

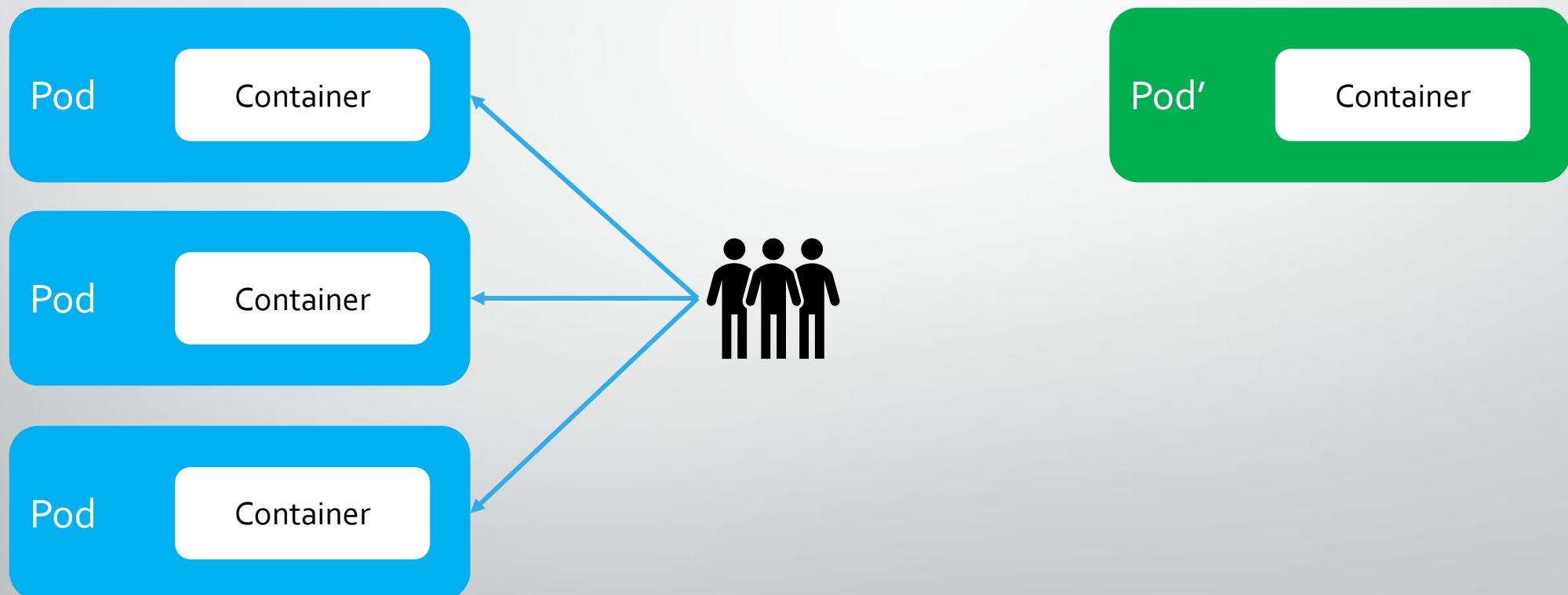
Start all new pods and then switch



Deployment Strategies

Blue Green deployment

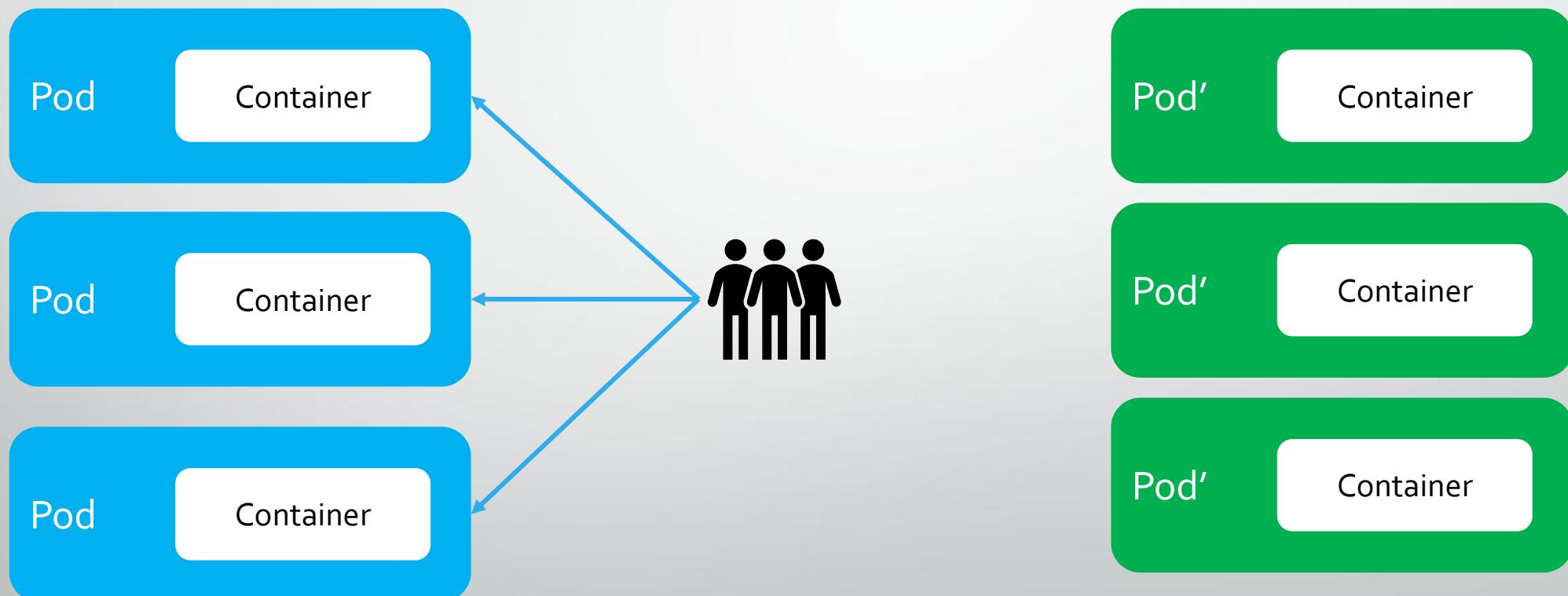
Start all new pods and then switch



Deployment Strategies

Blue Green deployment

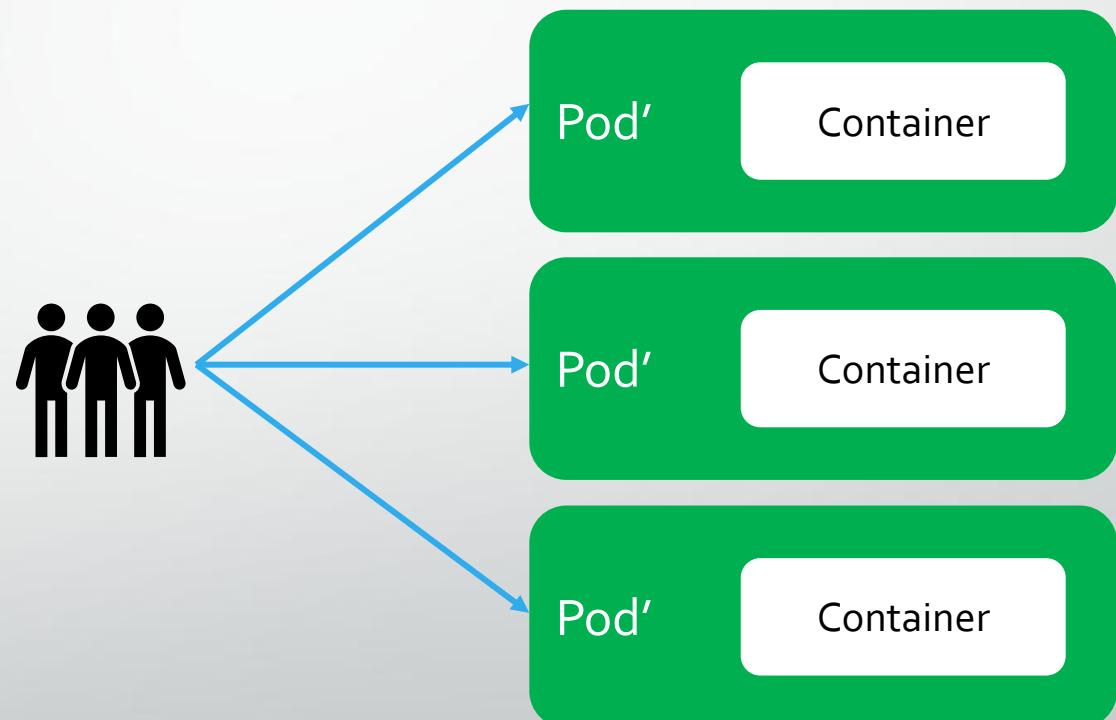
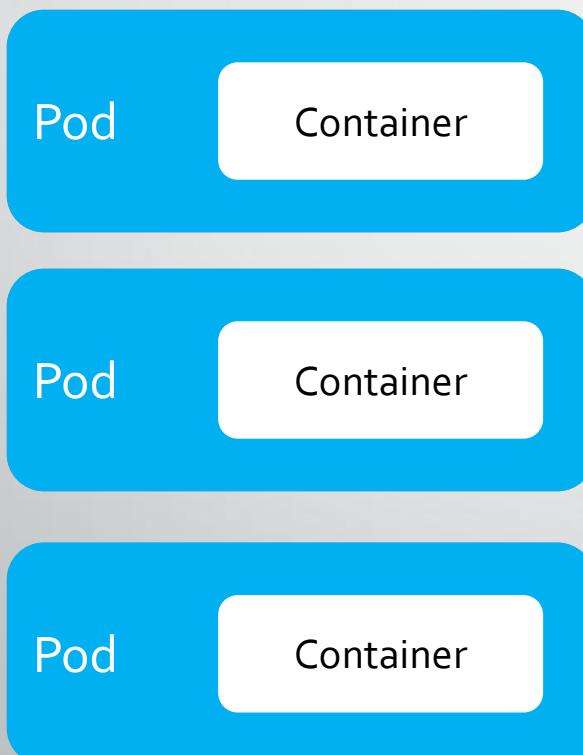
Start all new pods and then switch



Deployment Strategies

Blue Green deployment

Start all new pods and then switch



Deployment Strategies

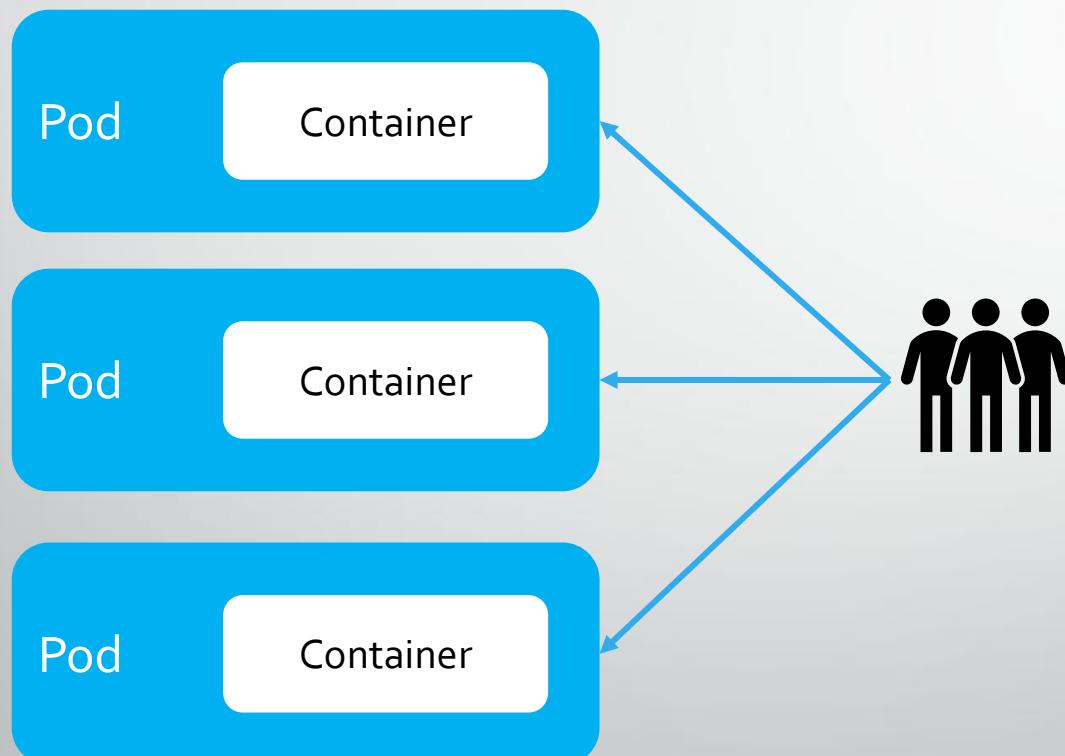
Rolling deployment

Replace old pods with new ones until all are replaced

Deployment Strategies

Rolling deployment

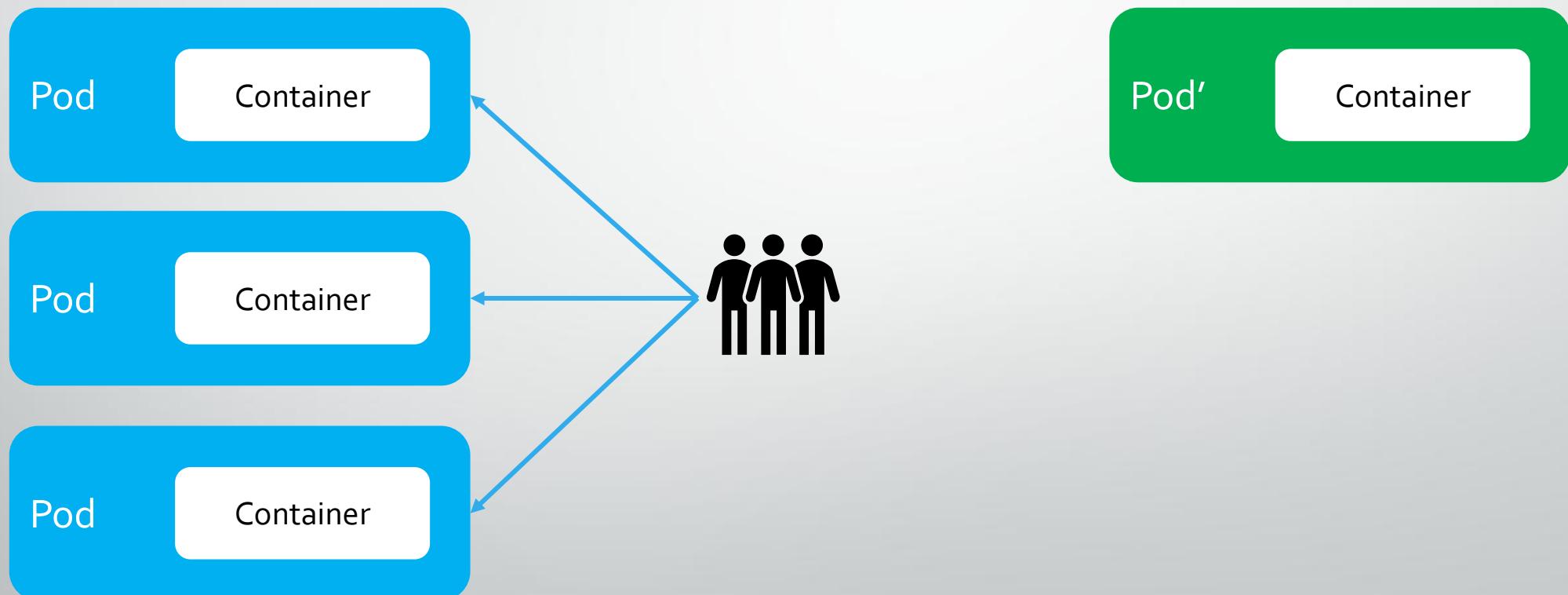
Replace old pods with new ones until all are replaced



Deployment Strategies

Rolling deployment

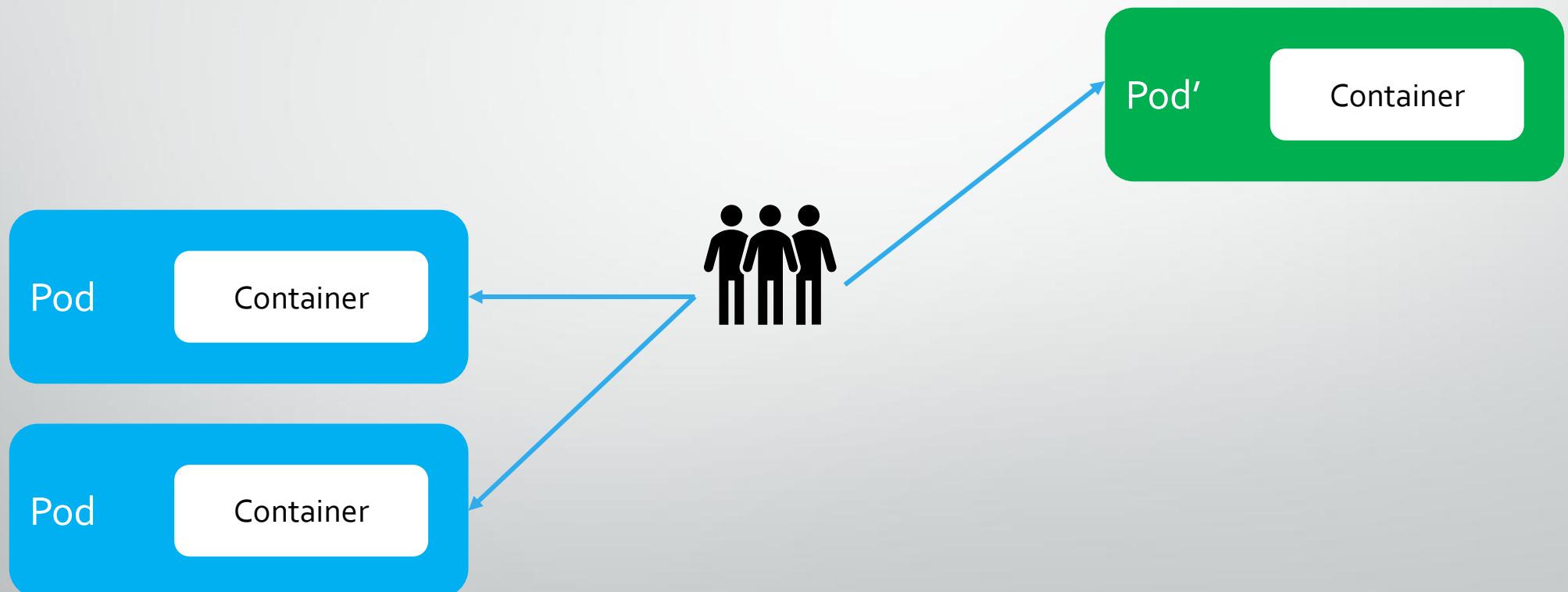
Replace old pods with new ones until all are replaced



Deployment Strategies

Rolling deployment

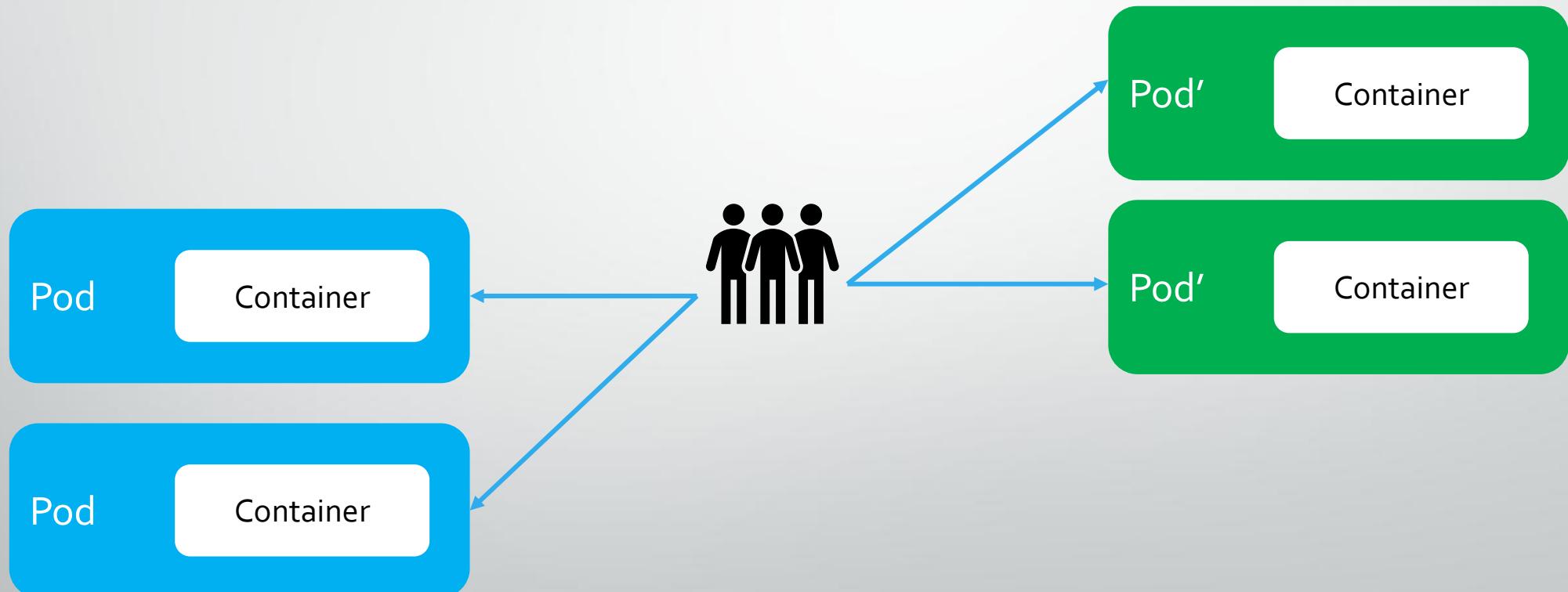
Replace old pods with new ones until all are replaced



Deployment Strategies

Rolling deployment

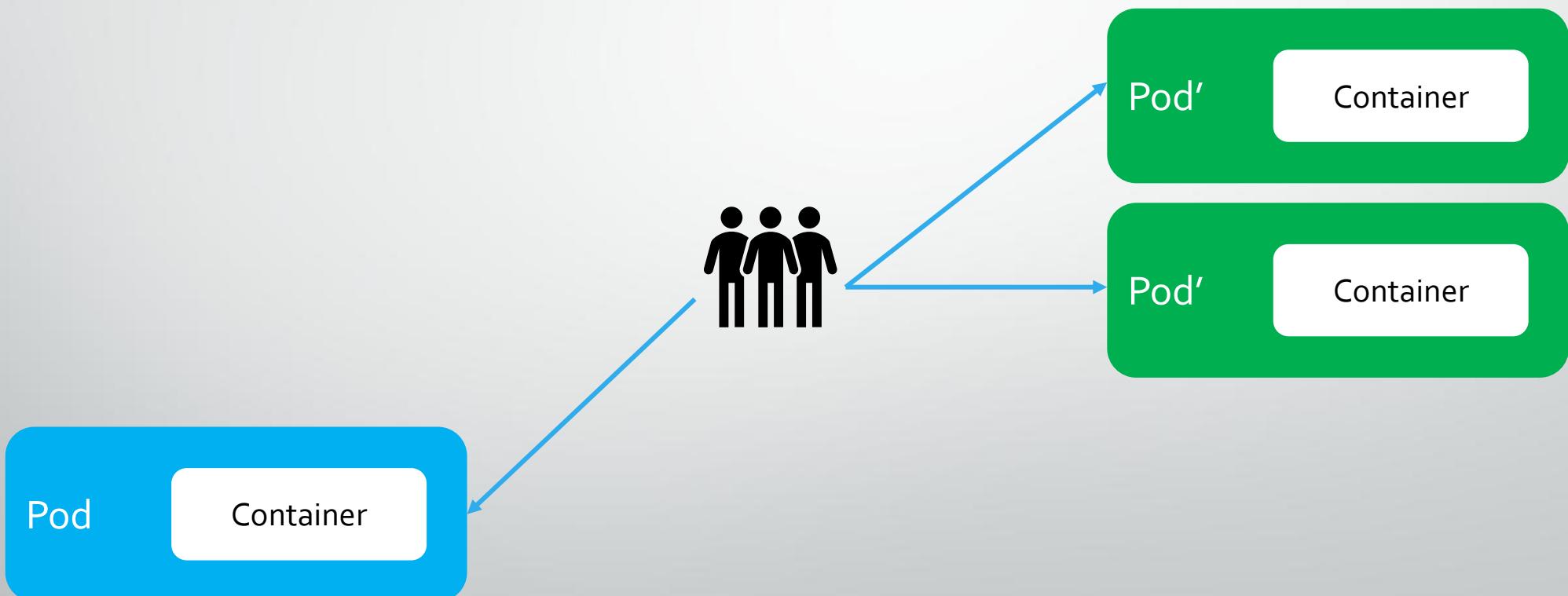
Replace old pods with new ones until all are replaced



Deployment Strategies

Rolling deployment

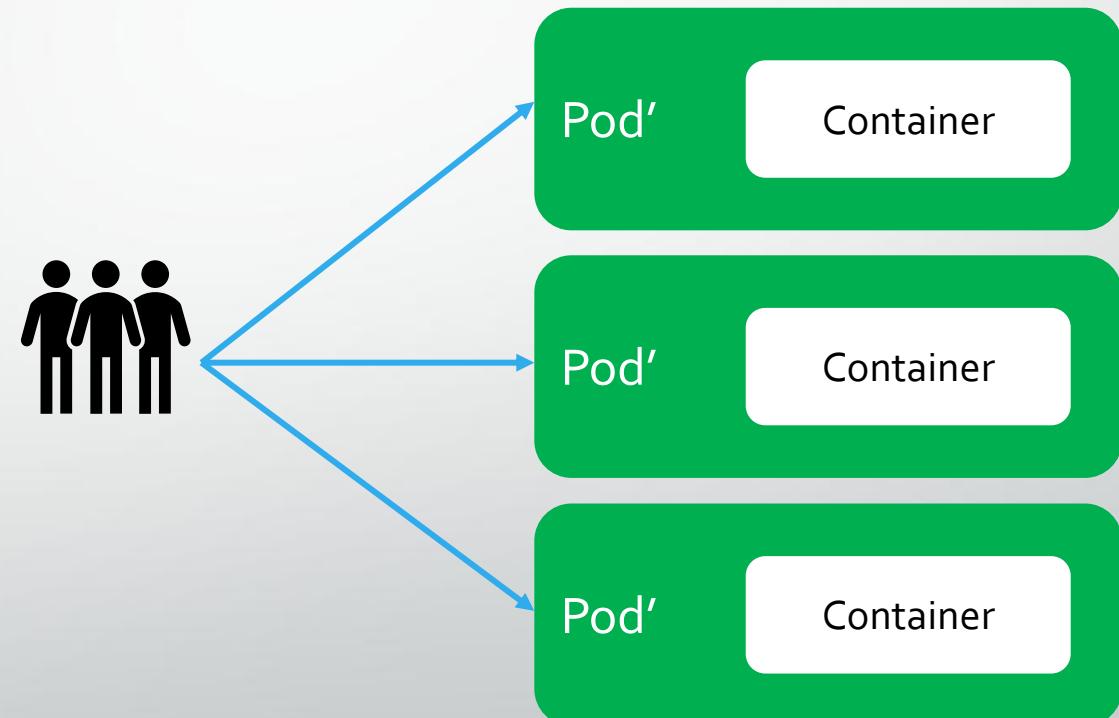
Replace old pods with new ones until all are replaced



Deployment Strategies

Rolling deployment

Replace old pods with new ones until all are replaced

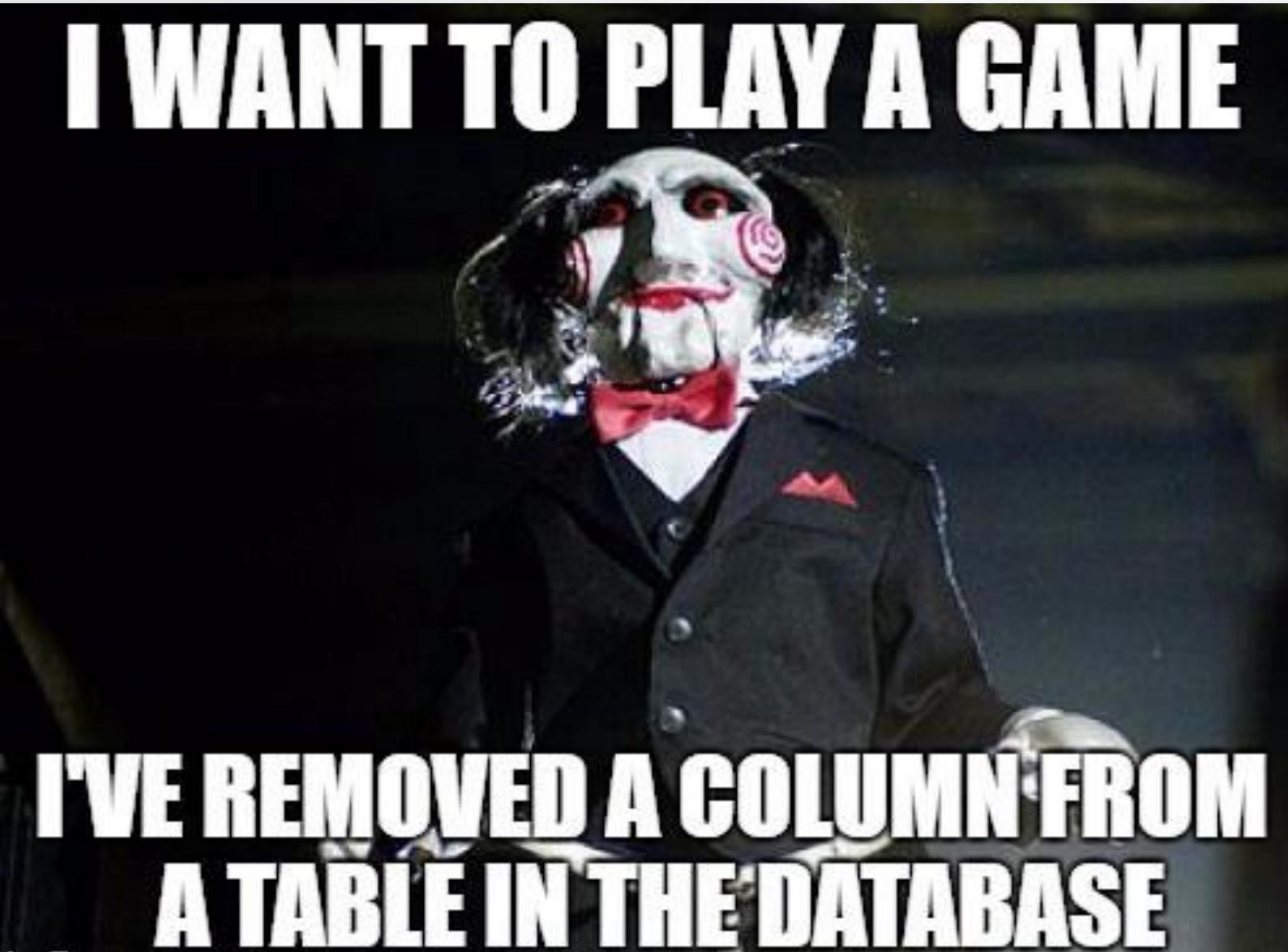


Deployment Strategies

Rolling deployment

- Default deployment mode
- Application must support two versions

Deployment Strategies



Deployment Strategies

Canary deployment

- Route only a small percentage of traffic to the new version
- Reduce the blast radius of a bad deployment
- A/B testing



Zero Downtime Deployment Demo

Deployment Strategies

maxSurge

- Max number of pods that can be created at a time
- Absolut number or percentage
- Default: 25%

maxUnavailable

- Max number of pods that can be available during the deployment
- Absolut number or percentage

Default: 25%



Considerations when using K8s

Cloud-native architecture

Microservices

.NET Full Frameworks vs .NET (Core)

DevOps process and culture

Deploy fast and often

Fast paced development and deployment

Cloud hosted vs. on-premises

When not to use Kubernetes

Skills and experience of the team

Application that will be barely change

Big monolithic applications

Quick results

Very simple applications

Cloud hosted vs. on-premises



Kubectl Commands

Get resource

kubectl get pods/service/deployment

Delete resource

kubectl delete pod/service/deployment

Display information about resource

kubectl describe pod/node/service resource-name

Add/update new resource

kubectl apply -f myfile.yaml [namespace=my-namespace]

Set current namespace

kubectl config set-context --current --namespace=my-namespace

Kubernetes Cheat Sheet:

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

Exercise

Exercise

Play around with Kubernetes

- Connect to the cluster
- See what components are available using the CLI and the dashboard
- Update an existing application
- Change the ports in the Service
- Implement an HPA and test autoscaling
- Add a Readiness and Liveness Probe

Useful links

- [Kubernetes Documentation](#)
- [Demo Application CustomerApi](#)
- [Microservice Series - From Zero to Hero](#)
- [Phippy Goes To The Zoo – PDF](#)
- [Phippy Goes To The Zoo - Video](#)



Helm

Complex Configuration

```
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
    meta.helm.sh/release-name: customerapi-customerapi-test
    meta.helm.sh/release-namespace: customerapi-test
  creationTimestamp: "2021-11-01T15:10:45Z"
  generation: 4
  labels:
    app: customerapi
    app.kubernetes.io/managed-by: Helm
    chart: customerapi-0.1.126
    draft: draft-app
    heritage: Helm
    release: customerapi-customerapi-test
```

```
  manager: kube-controller-manager
  operation: Update
  time: "2021-11-01T15:13:51Z"
  name: customerapi
  namespace: customerapi-test
  resourceVersion: "19234"
  uid: f45511b7-7599-41fb-a129-4973a5926ca2
spec:
  progressDeadlineSeconds: 600
  replicas: 1
  revisionHistoryLimit: 0
  selector:
    matchLabels:
      app: customerapi
      release: customerapi-customerapi-test
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
      type: RollingUpdate
  template:
    metadata:
      annotations:
        buildID: ""
      creationTimestamp: null
      labels:
        app: customerapi
        draft: draft-app
        release: customerapi-customerapi-test
    spec:
      containers:
        - env:
            - name: AzureServiceBus__ConnectionString
              valueFrom:
                secretKeyRef:
                  key: AzureServiceBus__ConnectionString
                  name: customerapi-connectionstrings
```

Helm

Packet Manager for Kubernetes

Helps to manage Kubernetes applications

Template Engine

Bundle of YAML files is called Helm charts

Helm charts describe applications

Simple sharing of Helm charts via ArtifactHub.io

Helm

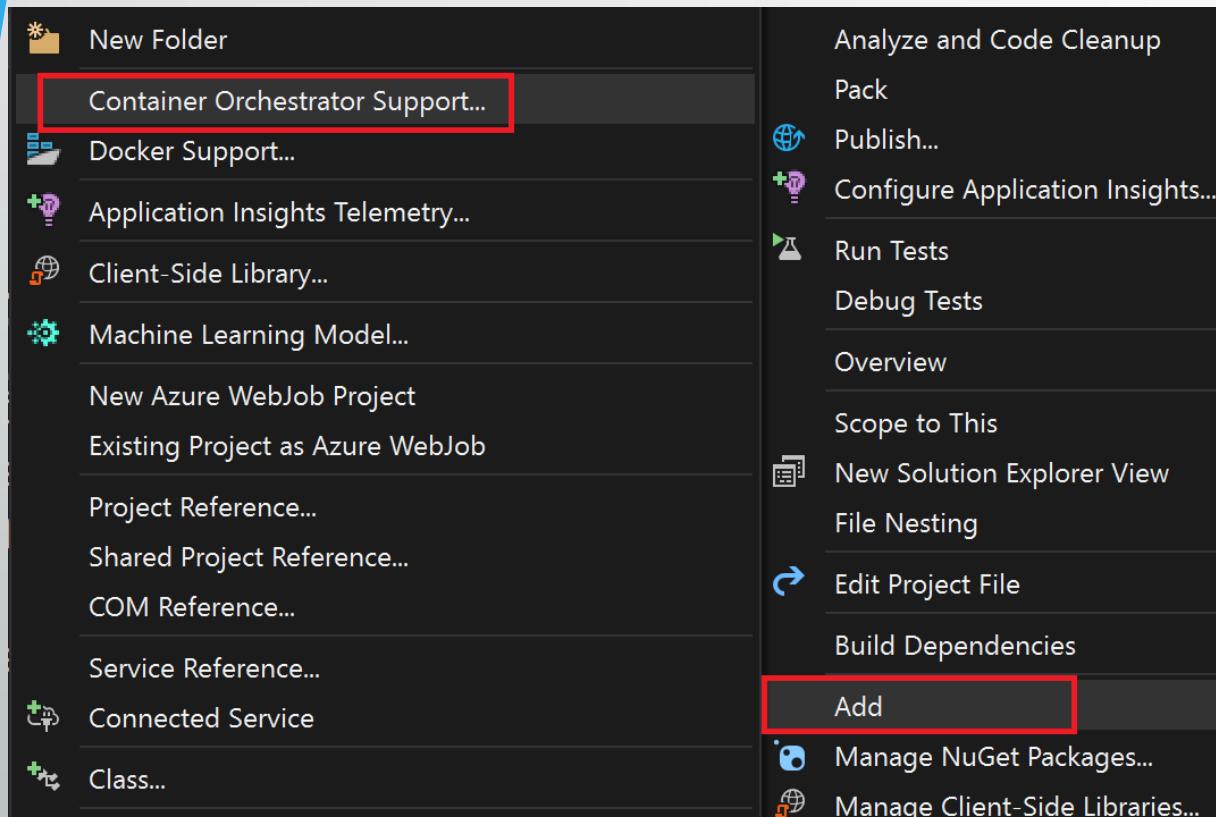
Install Helm

- Linux: curl
<https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3> | bash
- Windows: winget install Helm.Helm
- Mac: brew install helm

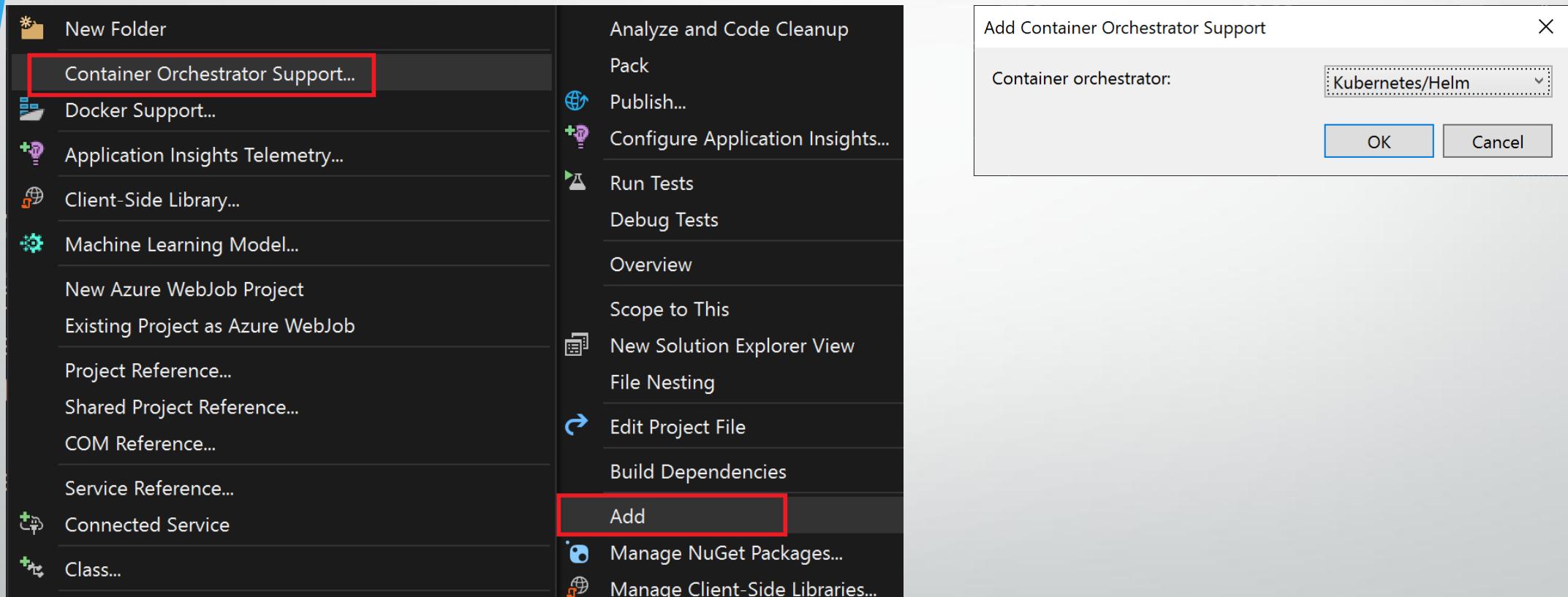
Helm Charts

chartname/	
Chart.yaml	A YAML file containing information about the chart
LICENSE	OPTIONAL: A plain text file containing the license for the chart
README.md	OPTIONAL: A human-readable README file
values.yaml	The default configuration values for this chart
charts/	A directory containing any charts upon which this chart depends.
templates/	A directory of templates that, when combined with values, will generate valid Kubernetes manifest files.
templates/NOTES.txt	:OPTIONAL: A plain text file containing short usage notes

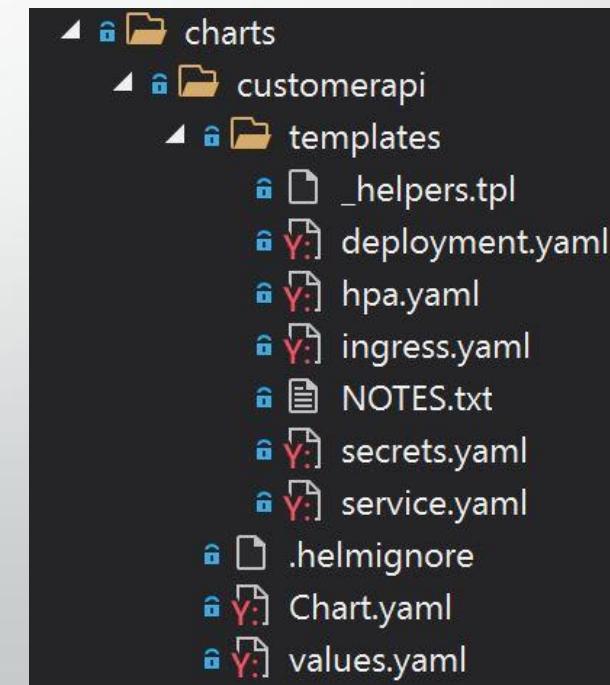
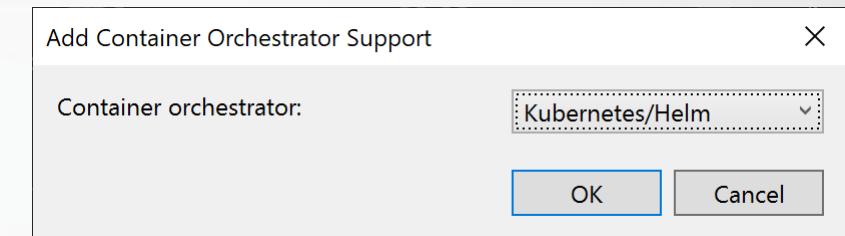
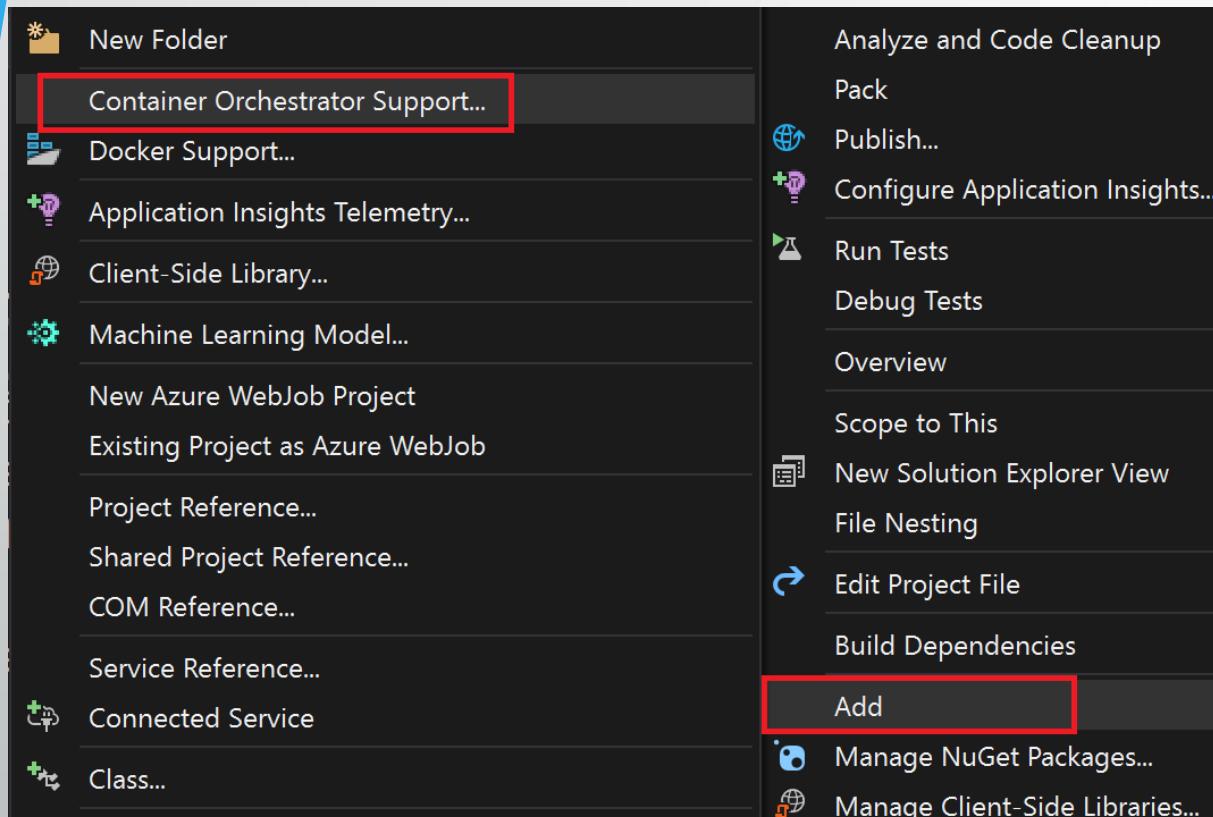
Add Helm Charts in Visual Studio



Add Helm Charts in Visual Studio



Add Helm Charts in Visual Studio



Add Helm Charts with CLI

mkdir charts

cd charts

helm create <ChartName>

```
template:
  metadata:
    annotations:
      buildID: ""
    creationTimestamp: null
    labels:
      app: customerapi
      draft: draft-app
      release: customerapi-customerapi-test
  spec:
    containers:
      - env:
          - name: AzureServiceBus__ConnectionString
            valueFrom:
              secretKeyRef:
                key: AzureServiceBus__ConnectionString
                name: customerapi-connectionstrings
          - name: ConnectionStrings__CustomerDatabase
            valueFrom:
              secretKeyRef:
                key: ConnectionStrings__CustomerDatabase
                name: customerapi-connectionstrings
        image: wolfgangofner/customerapi:0.1.402
        imagePullPolicy: IfNotPresent
    livenessProbe:
      failureThreshold: 3
      httpGet:
        path: /health
        port: http
        scheme: HTTP
      initialDelaySeconds: 15
      periodSeconds: 10
      successThreshold: 1
      timeoutSeconds: 1
    name: customerapi
    ports:
      - containerPort: 80
        name: http
```

```
apiVersion: apps/v1
kind: Deployment
  metadata:
    name: {{ template "customerapi.fullname" . }}
    labels:
      app: {{ template "customerapi.name" . }}
      chart: {{ template "customerapi.chart" . }}
      draft: {{ .Values.draft | default "draft-app" }}
      release: {{ .Release.Name }}
      heritage: {{ .Release.Service }}
  spec:
    revisionHistoryLimit: 0
    replicas: {{ .Values.replicaCount }}
    selector:
      matchLabels:
        app: {{ template "customerapi.name" . }}
        release: {{ .Release.Name }}
    template:
      metadata:
        labels:
          app: {{ template "customerapi.name" . }}
          draft: {{ .Values.draft | default "draft-app" }}
          release: {{ .Release.Name }}
        annotations:
          buildID: {{ .Values.buildID | default "" | quote }}
    spec:
      containers:
        - name: {{ .Chart.Name }}
          image: "{{ .Values.image.repository }}:{{ .Values.image.tag }}"
          imagePullPolicy: {{ .Values.image.pullPolicy }}
        ports:
          - name: http
            containerPort: {{ .Values.deployment.containerPort }}
            protocol: TCP
        {{- if .Values.probes.enabled --}}
        livenessProbe:
          httpGet:
            path: /health
            port: http
          initialDelaySeconds: 15
```

Values.yaml

```
fullnameOverride: customerapi
replicaCount: 1
image:
  repository: __Repository__
  tag: __BuildNumber__
  pullPolicy: IfNotPresent
imagePullSecrets: []
service:
  type: LoadBalancer
  port: 80

deployment:
  containerPort: 80

probes:
  enabled: false
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: {{ template "customerapi.fullname" . }}
  labels:
    app: {{ template "customerapi.name" . }}
    chart: {{ template "customerapi.chart" . }}
    draft: {{ .Values.draft | default "draft-app" }}
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
spec:
  revisionHistoryLimit: 0
  replicas: {{ .Values.replicaCount }}
  selector:
    matchLabels:
      app: {{ template "customerapi.name" . }}
      release: {{ .Release.Name }}
  template:
    metadata:
      labels:
        app: {{ template "customerapi.name" . }}
        draft: {{ .Values.draft | default "draft-app" }}
        release: {{ .Release.Name }}
      annotations:
        buildID: {{ .Values.buildID | default "" | quote }}
    spec:
      containers:
        - name: {{ .Chart.Name }}
          image: "{{ .Values.image.repository }}:{{ .Values.image.tag }}"
          imagePullPolicy: {{ .Values.image.pullPolicy }}
          ports:
            - name: http
              containerPort: {{ .Values.deployment.containerPort }}
              protocol: TCP
            {{- if .Values.probes.enabled }}
              livenessProbe:
                httpGet:
                  path: /health
                  port: http
                initialDelaySeconds: 15
```

Override Values in CI/CD Pipeline

```
fullnameOverride: customerapi
replicaCount: 1
image:
  repository: __Repository__
  tag: __BuildNumber__
  pullPolicy: IfNotPresent
imagePullSecrets: []
service:
  type: LoadBalancer
  port: 80

deployment:
  containerPort: 80

probes:
  enabled: false
```

```
variables:
  - ApiName: 'customerapi'
  - BuildNumber: $(GitVersion.FullSemVer)
  - Repository: 'wolfgangofnerbbv/${ApiName}'

steps:
  - task: Tokenizer@0
    displayName: 'Run Tokenizer'
```

Helm Commands

List all deployments

helm ls

Install Helm Chart

helm install my-release-name my-helm-chart-name --namespace my-namespace

helm install customer customerapi

Update Release

helm upgrade customer customerapi

Uninstall Release

helm uninstall customer

Rollback Release

helm rollback customer

Release Management

Install or upgrade Charts

Helm will only update components that have changed since the last release

Release Management

Install or upgrade Charts

Helm will only update components that have changed since the last release

NAME	NAMESPACE	REVISION	UPDATED	STATUS	CHART	APP VERSION
cert-manager	cert-manager	1	2021-10-17 12:00:51.12066323 +0000 UTC	deployed	cert-manager-v1.5.4	v1.5.4
customerapi-customerapi-test	customerapi-test	1	2021-10-17 12:08:31.037732341 +0000 UTC	deployed	customerapi-0.1.402	1.0
ingress-nginx	ingress-basic	1	2021-10-17 12:01:25.840929561 +0000 UTC	deployed	ingress-nginx-4.0.6	1.0.4
keda	keda	1	2021-10-17 12:04:21.520467691 +0000 UTC	deployed	keda-2.4.0	2.4.0
kedademoapi-kedademoapi-prod	kedademoapi-prod	1	2021-10-17 12:09:50.399832095 +0000 UTC	deployed	kedademoapi-0.1.417	1.0
kedademoapi-kedademoapi-test	kedademoapi-test	1	2021-10-17 12:06:59.357610554 +0000 UTC	deployed	kedademoapi-0.1.417	1.0
loki	loki-grafana	1	2021-10-17 12:02:10.25707972 +0000 UTC	deployed	loki-stack-2.0.3	v2.0.0
orderapi-orderapi-test	orderapi-test	1	2021-10-17 12:08:05.018913511 +0000 UTC	failed	orderapi-0.1.421	1.0



Helm Demo

Exercise

Play around with Helm

- Create a Helm chart
- Install a public Helm chart in your K8s cluster
- Deploy an application to K8s using your Helm chart
- Update a Helm deployment

Useful links:

- [Helm Documentation](#)
- [Demo Application CustomerApi](#)
- [Microservice Series - From Zero to Hero](#)

Solved all your problems. You're welcome.

