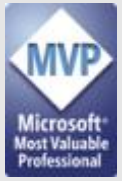
Bring your ASP.NET Core solutions to Kubernetes in Azure

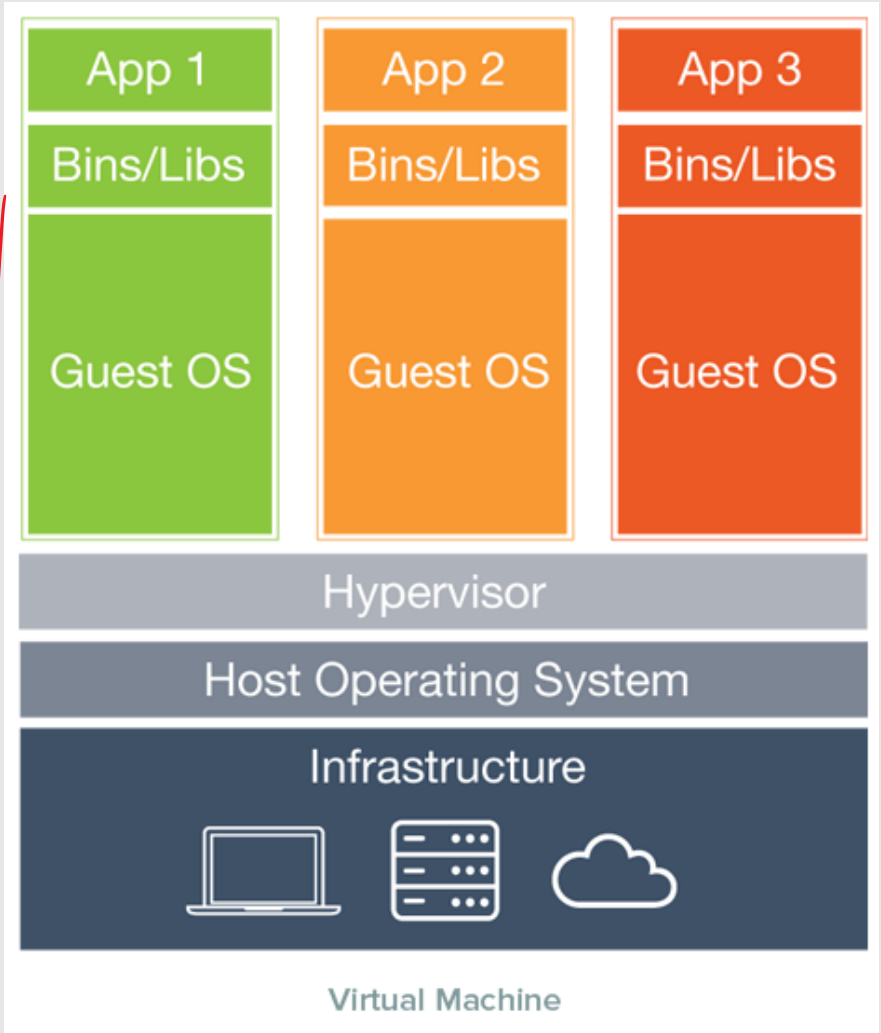


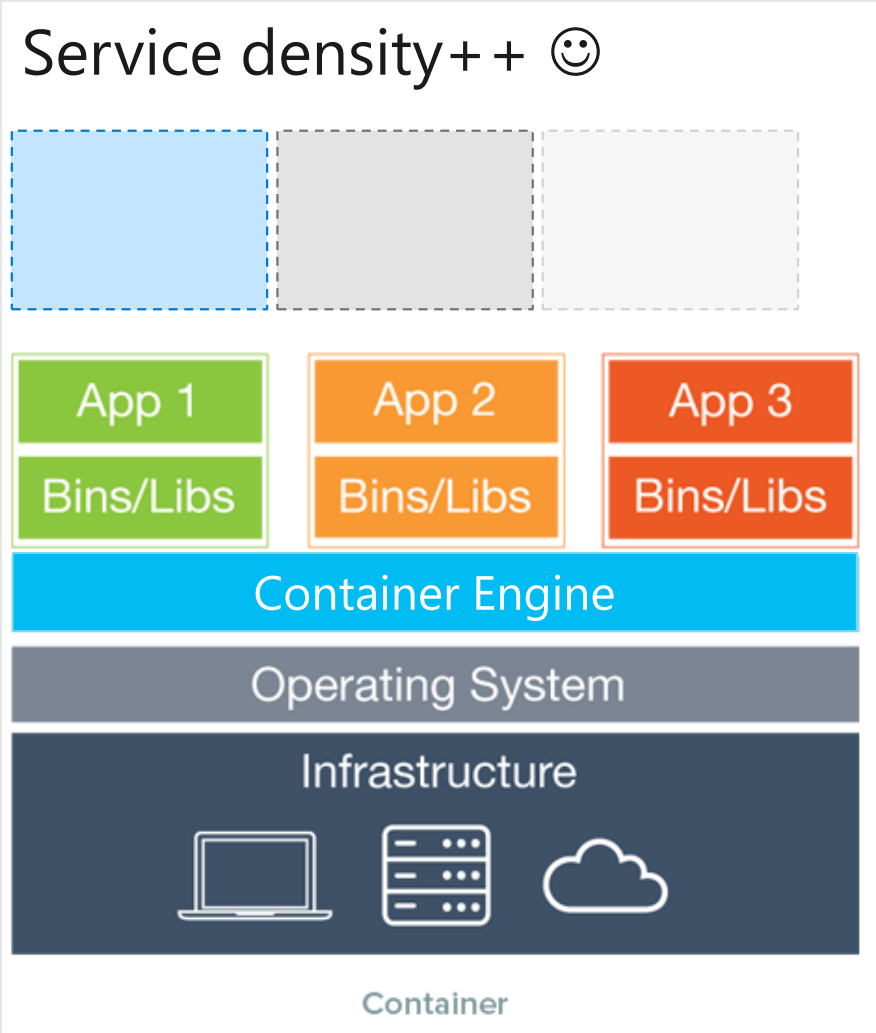
Marco De Sanctis
Visual Studio and Development Technologies MVP
info@marcodesanctis.it - @crad77

Agenda

- The Basics of Docker and why I should care
- Running ASP.NET Core in Docker
- Let's port a non-trivial project to Docker and AKS

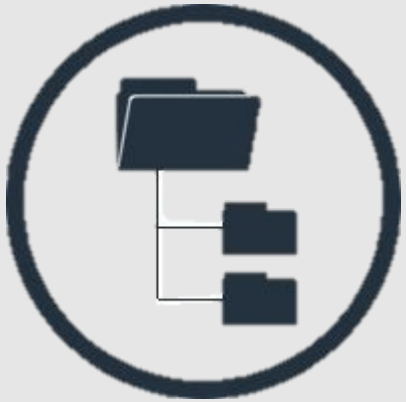
How did we get here



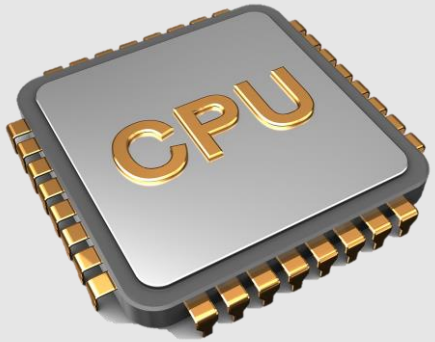


Containers == Virtualized Operating System

Kernel namespaces (Linux e Windows)



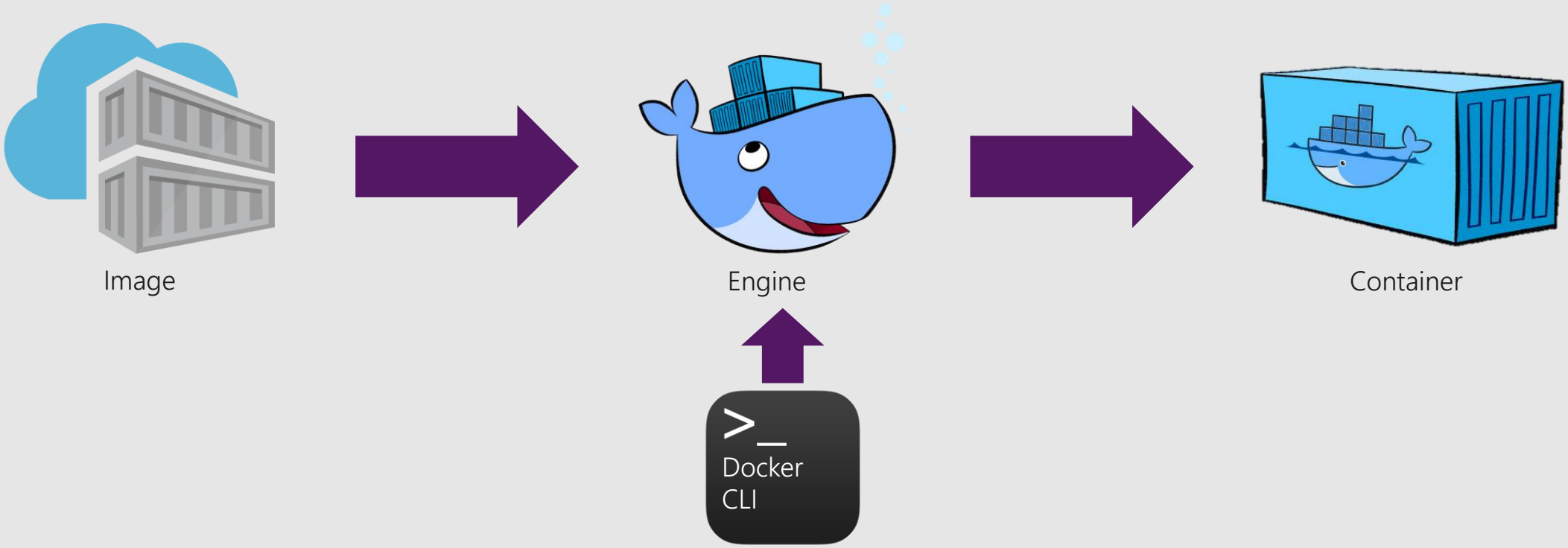
Virtual File System



Virtual Process Tree

Virtual Network

How does a container-based flow work?

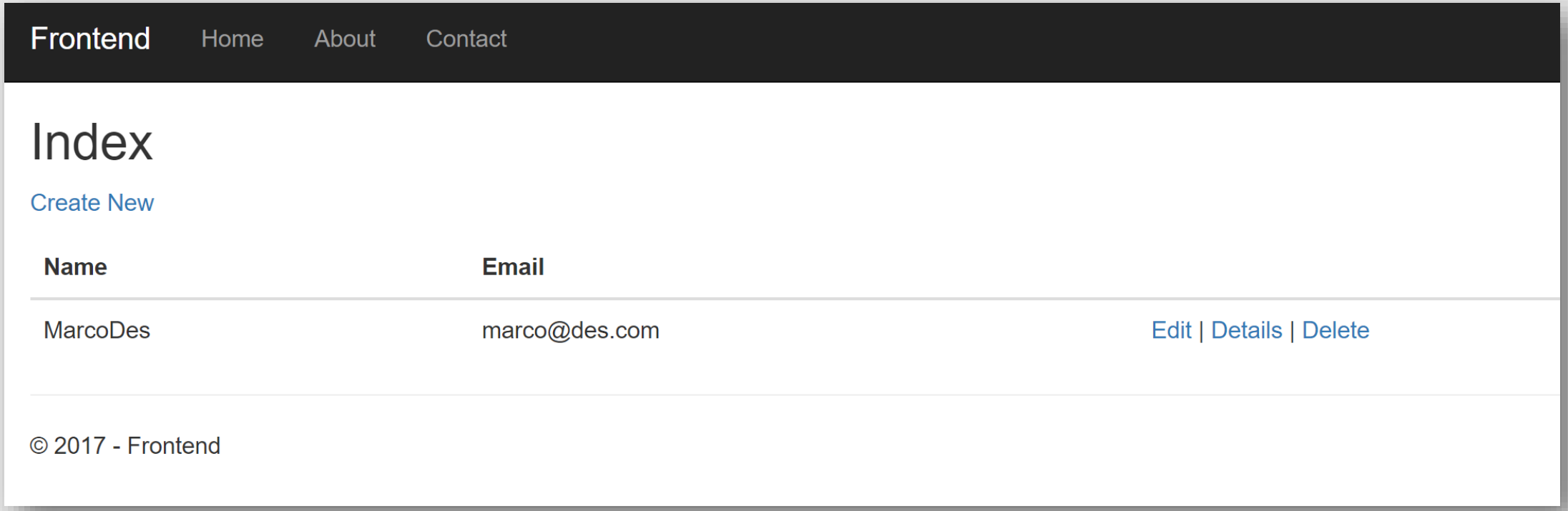


- Docker (aka Moby project) is free and open source, no limitations
- There's an enterprise edition (hosting, support, certification…)
- It's the de-facto standard for container-based applications

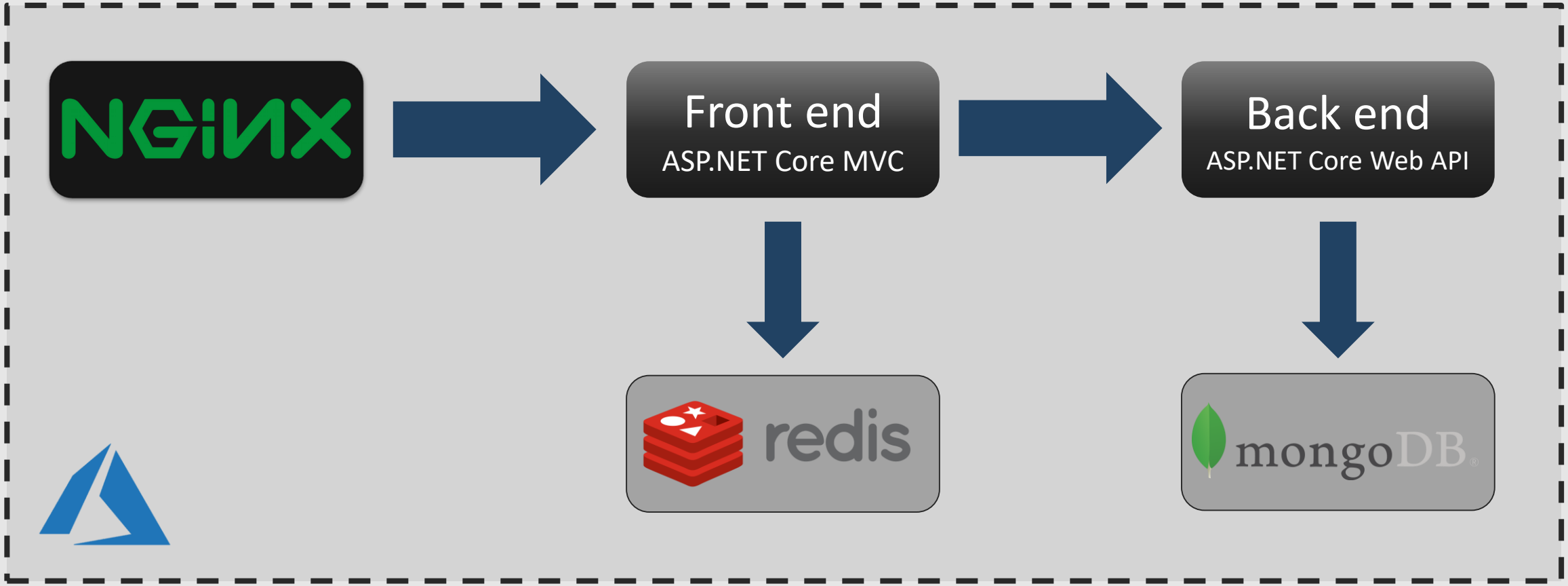
Let’s get started ☺

Demo

The application for today

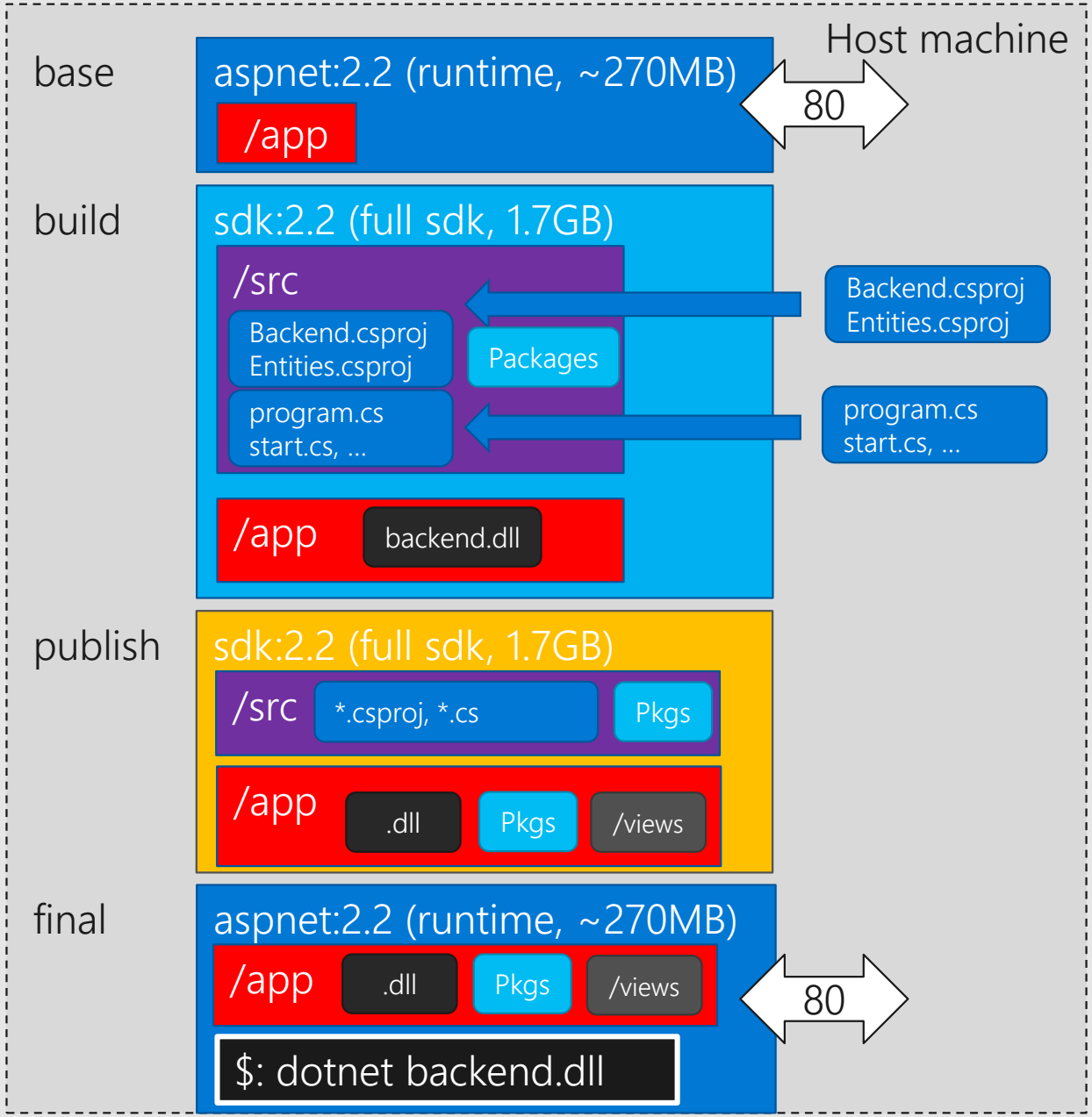


The application for today



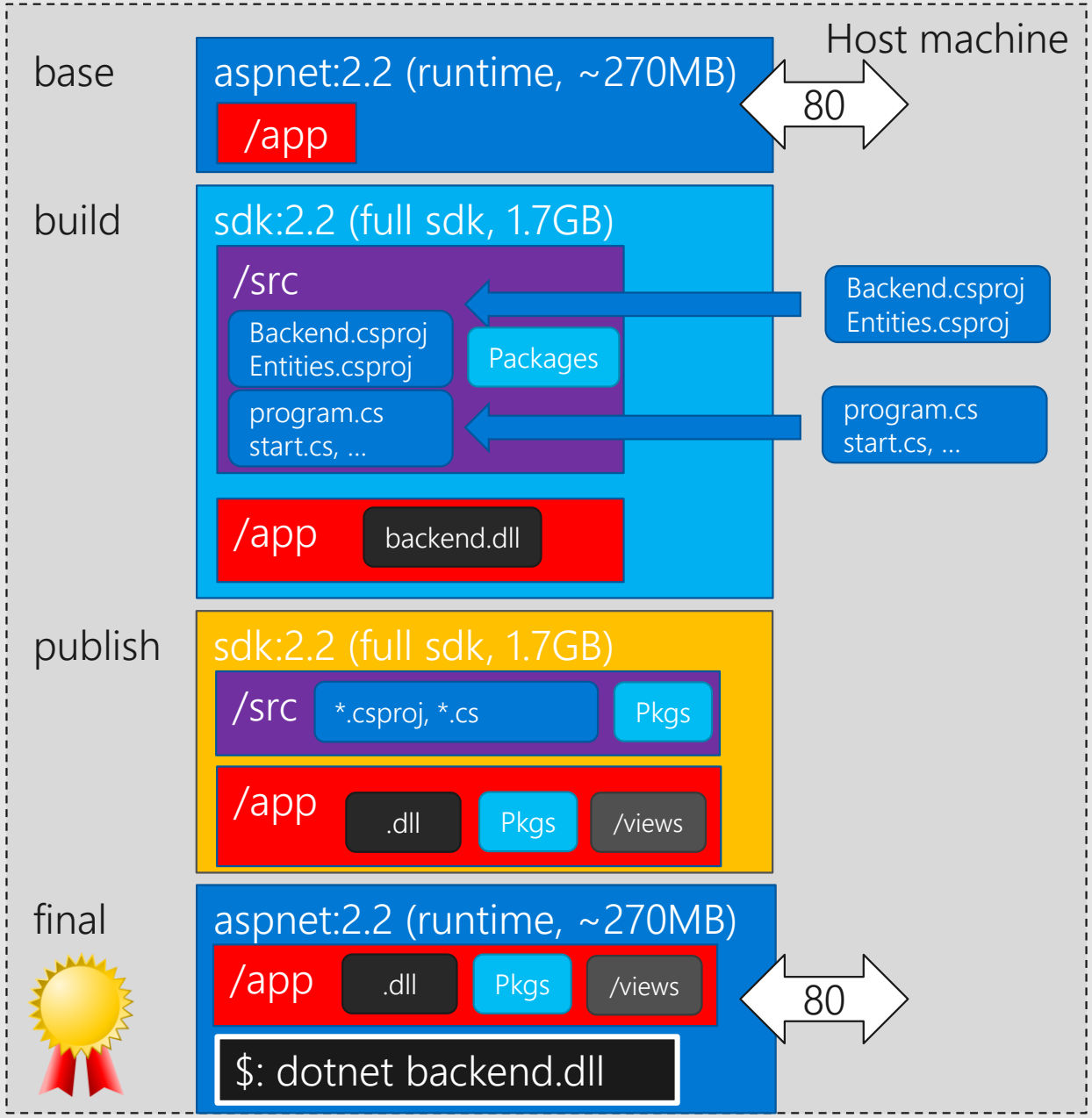
Dockerfile explained

```
FROM mcr.microsoft.com/dotnet/core/aspnet:2.2 AS base
WORKDIR /app
EXPOSE 80

FROM mcr.microsoft.com/dotnet/core/sdk:2.2 AS build
WORKDIR /src
COPY Backend/Backend.csproj Backend/
COPY Entities/Entities.csproj Entities/
RUN dotnet restore Backend/Backend.csproj

COPY . .
WORKDIR /src/backend
RUN dotnet build -c Release -o /app

FROM build AS publish
RUN dotnet publish -c Release -o /app


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

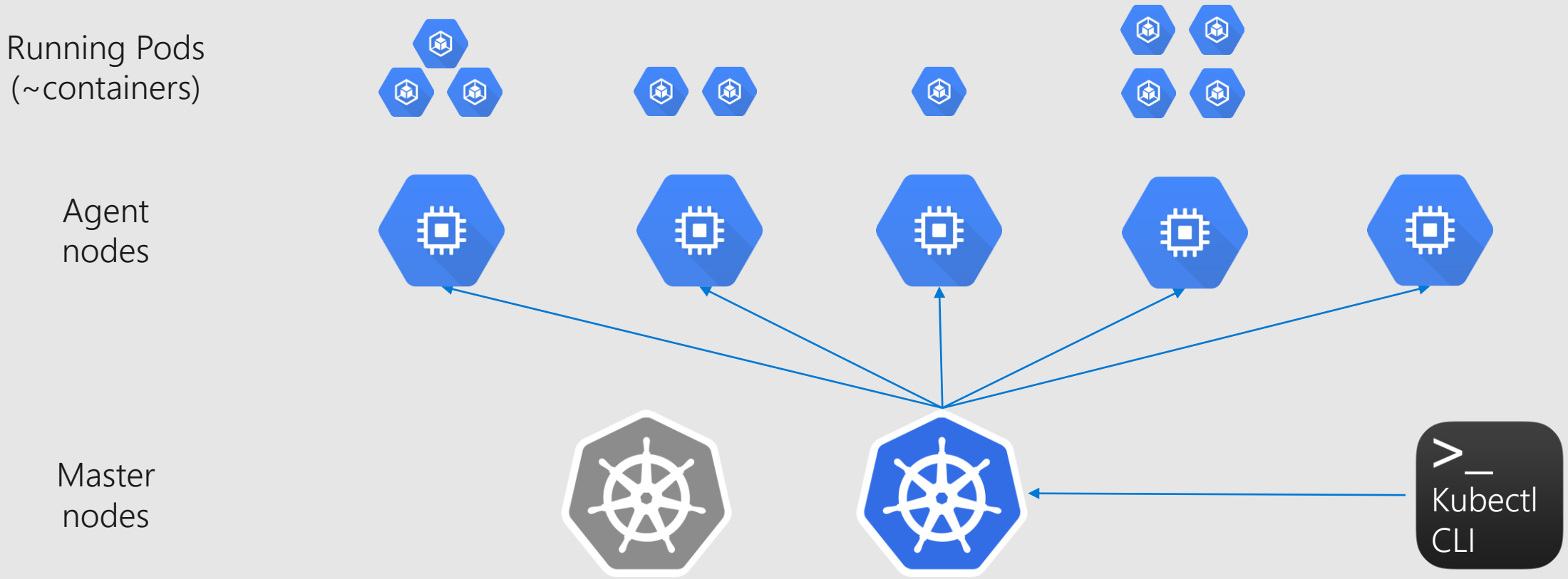


Dockerfile explained

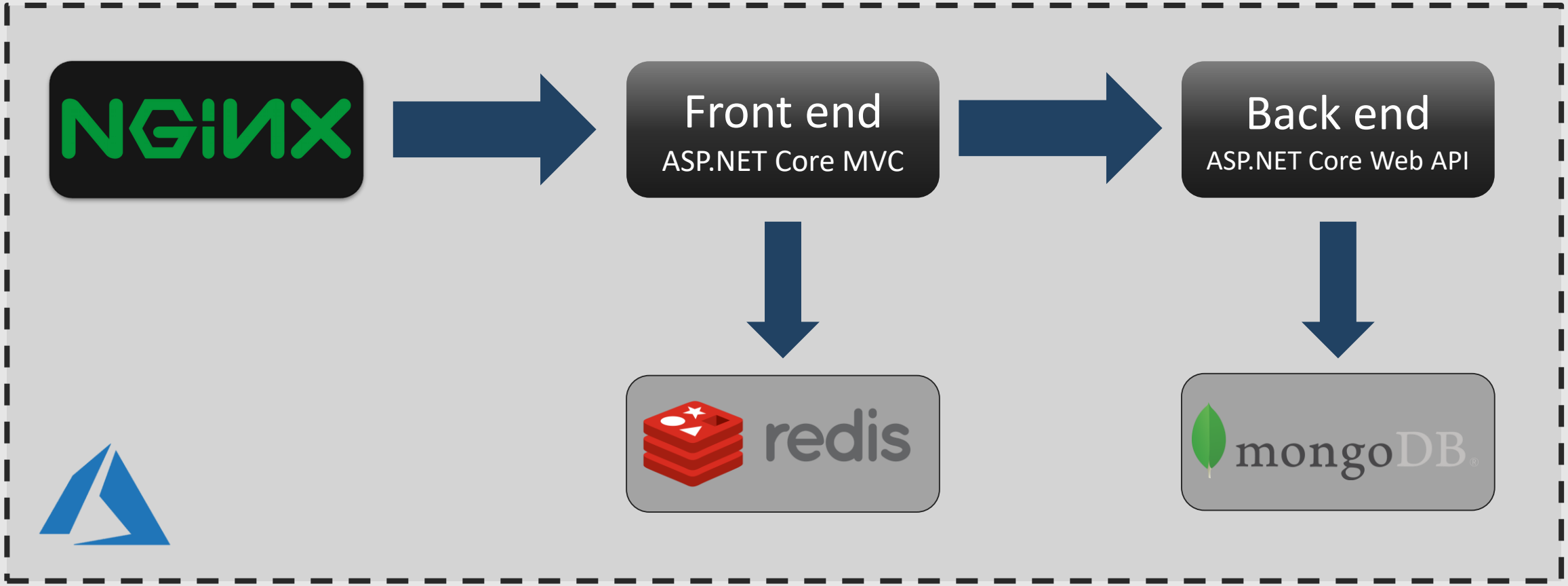
```
FROM mcr.microsoft.com/dotnet/core/aspnet:2.2 AS base
WORKDIR /app
EXPOSE 80

FROM mcr.microsoft.com/dotnet/core/sdk:2.2 AS build
WORKDIR /src
COPY Backend/Backend.csproj Backend/
COPY Entities/Entities.csproj Entities/
RUN dotnet restore Backend/Backend.csproj

COPY . .
WORKDIR /src/backend
RUN dotnet build -c Release -o /app

FROM build AS publish
RUN dotnet publish -c Release -o /app


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



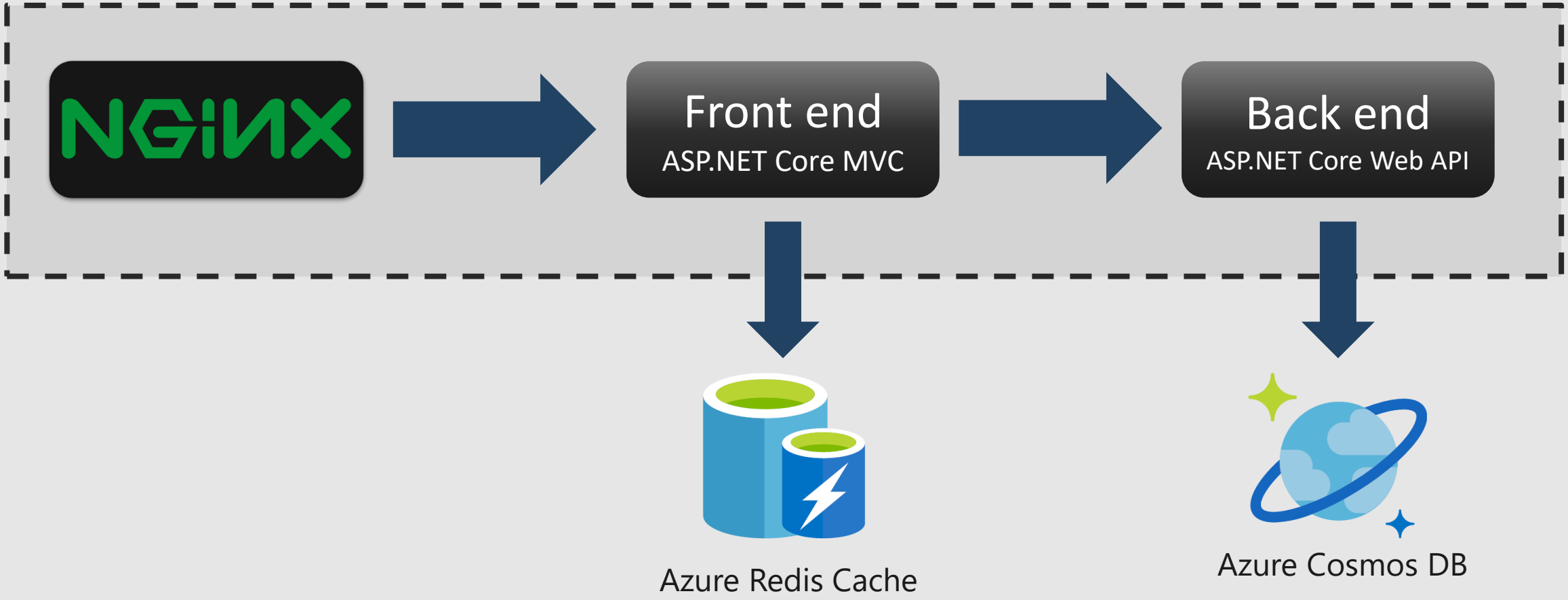
We need an orchestrator: enter Kubernetes and AKS



Our application so far...



Let's take advantage of Azure PaaS



A glimpse into Kubernetes object model

Deployment

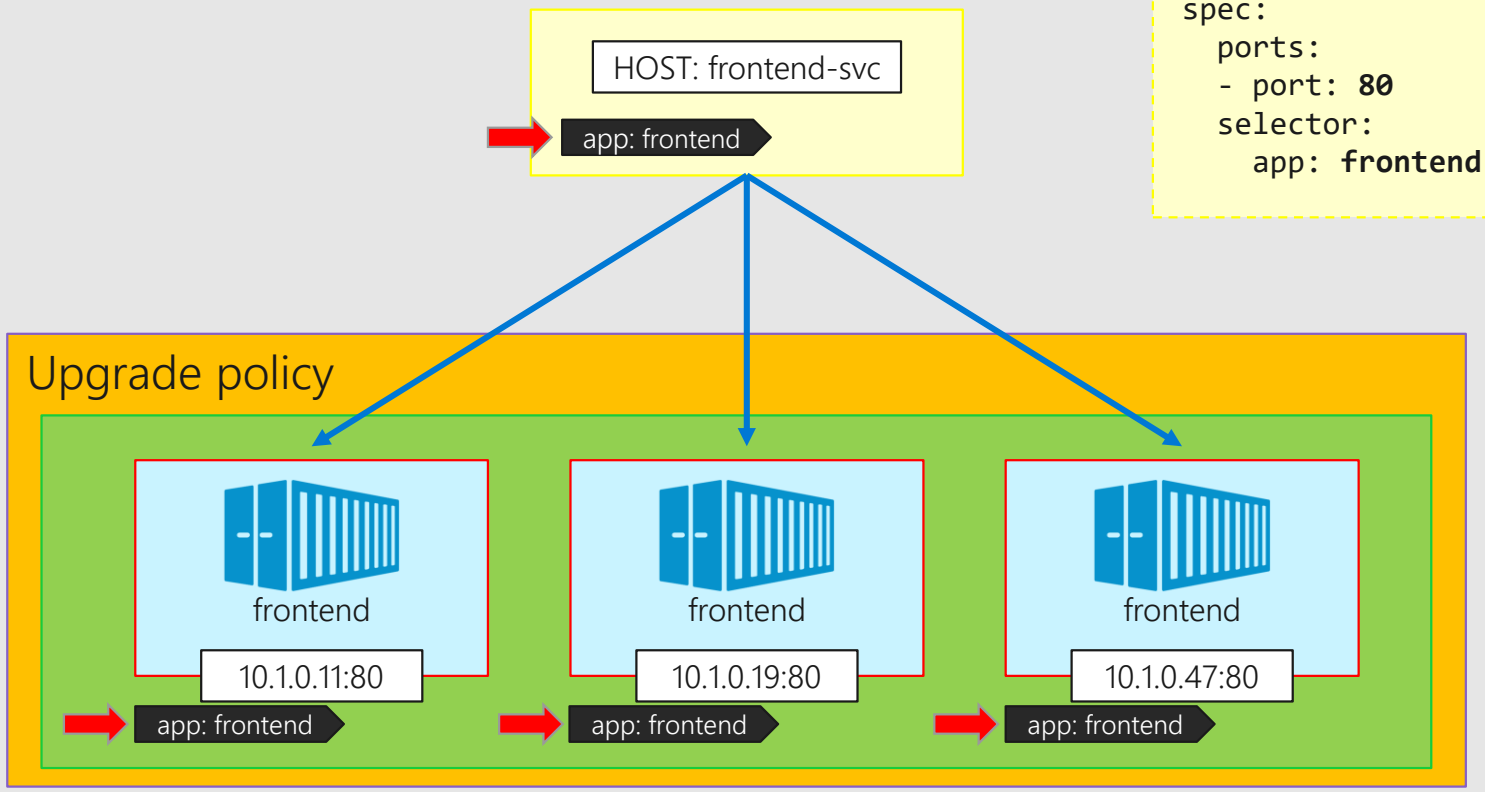
```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: frontend-dpy
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: frontend
    spec:
      containers:
      - name: frontend
        image: myreg/frontend:latest
        ports:
        - containerPort: 80
```

ReplicaSet

Pod

Service

```
apiVersion: v1
kind: Service
metadata:
  name: frontend-svc
spec:
  ports:
  - port: 80
  selector:
    app: frontend
```



Recap

- Run **Mongo** & **Redis** as Docker containers
- Dockerised **frontend** and **backend**
- Added **nginx**
- Described the whole system on **docker-compose**
- Run all of it on our laptop
- Saved the images on **Azure Container Registry**
- Configured a **CI/CD pipeline** for our solution
- Deployed on a Kubernetes cluster in **Azure Kubernetes Service (AKS)**

Recap - Commands

`docker run –p 8080:80 wordpress`	Starts a container from an image, exponing it on 8080
`docker ps`	Lists of all running containers (-a includes stopped ones)
`docker start/stop containerName`	Starts (and stops) a container
`docker rm -f $(docker ps -q)`	Removes all running containers
`docker images`	Lists all images
`docker rmi imageName`	Removes an image
`docker login myregistry.azurecr.io`	Logs in Azure Container Registry
`docker tag frontend myregistry.azurecr.io/frontend`	Tags an image for Azure Container Registry
`docker push myregistry.azurecr.io/frontend`	Pushes an image to Azure Container Registry
`kubectl apply -f filename`	Creates the kubectl objects specified in the file
`kubectl get deployments/services/pods`	Lists all the deployments/services/pods in the cluster
`kubectl proxy --address="0.0.0.0"`	Tunnels the cluster's dashboard to http://localhost:8001/api/v1/namespaces/kube-system/services/http:kubernetes-dashboard:/proxy

Recap – On Azure

- Azure Container Registry
  - Our private repository where we can store Docker images
  - `docker login myregistry.azurecr.io`
  - https://docs.microsoft.com/en-us/azure/container-registry/
- Azure App Services on Linux
  - They can either run one container at a time or a docker-compose file
  - Useful for simple scenarios – e.g. Run a small WordPress site with MySql
  - https://docs.microsoft.com/en-us/azure/app-service/containers/
- Azure Kubernetes Service (AKS)
  - Fully managed orchestrator
  - Based on Kubernetes
  - https://docs.microsoft.com/en-us/azure/aks/
- Container monitoring solution
  - Based on Azure LogAnalytics
  - Uses a DaemonSet to track cluster events
  - https://docs.microsoft.com/en-us/azure/log-analytics/log-analytics-containers

Thank you! ☺

@crad77

info@marcodesanctis.it

Get the code at

https://github.com/cradle77/DockerGettingStarted