AKS beyond Hello World

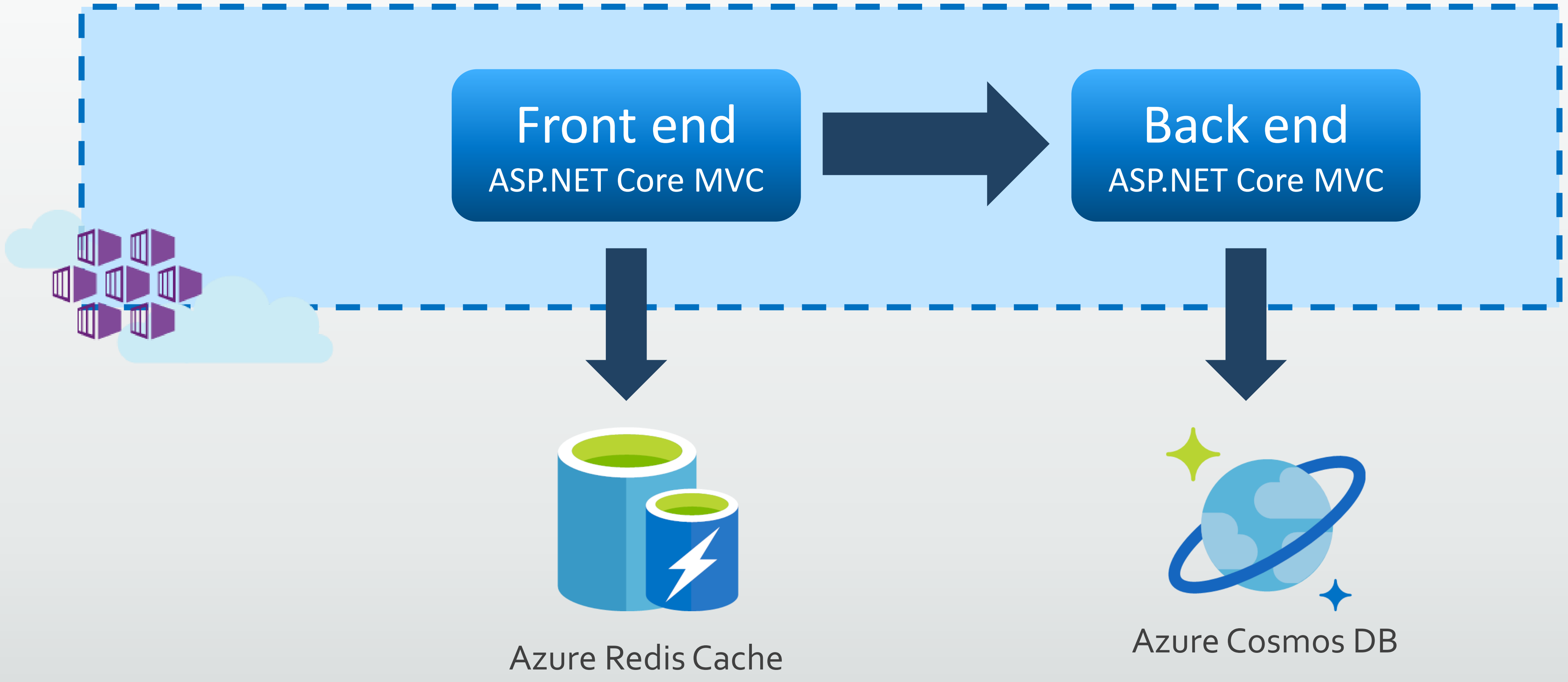
Bring your ASP.NET Core solution to production

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

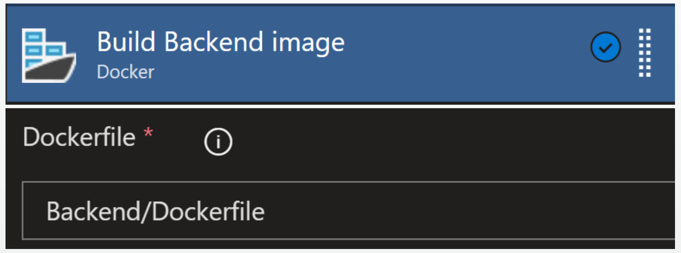
Agenda

- Build and Release pipelines
- Bring your own domain
- Pod Identity
- Governance
- Handling traffic spikes

Our application



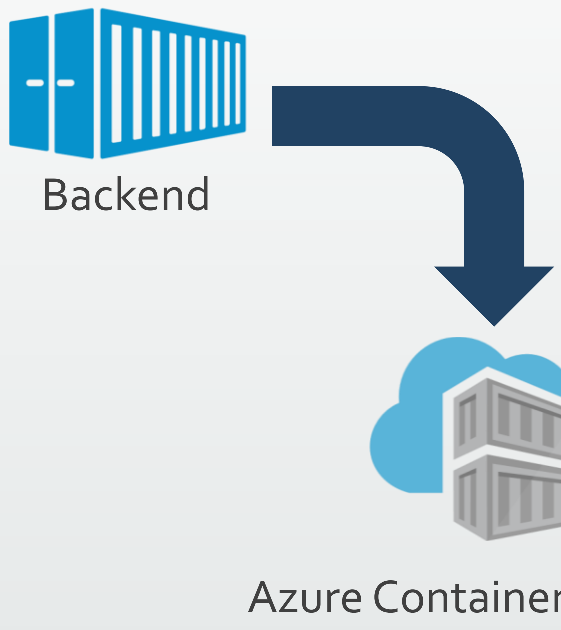
A simple build pipeline

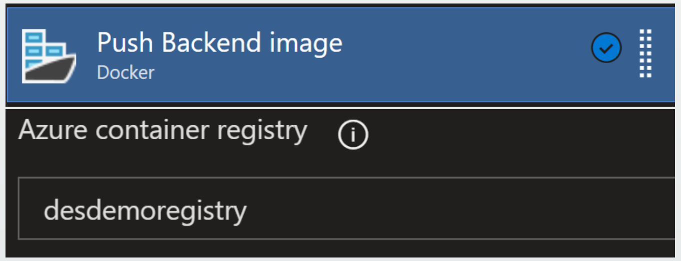


Build Backend image
Docker

Dockerfile *

Backend/Dockerfile

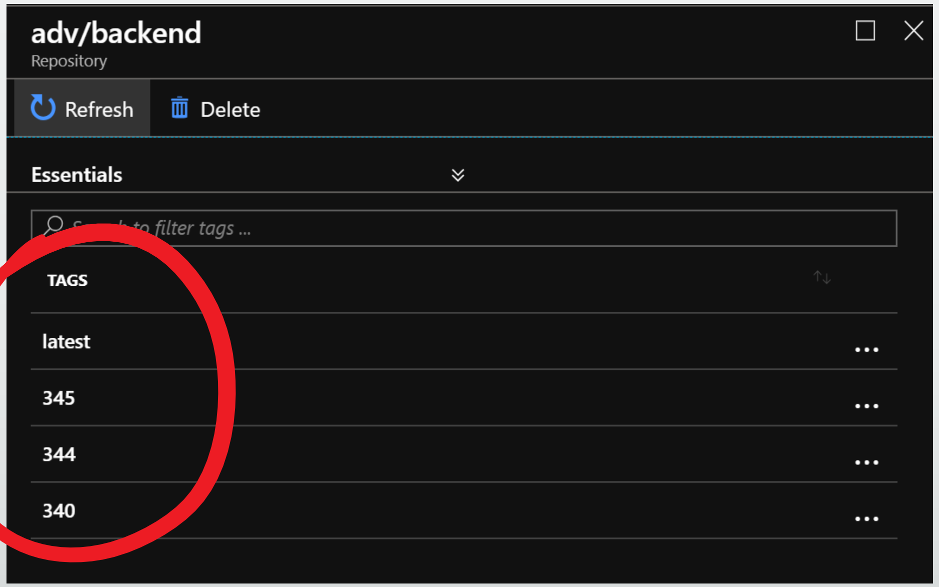




Push Backend image
Docker

Azure container registry

desdemoregistry



adv/backend
Repository

Refresh Delete

Essentials

Search to filter tags ...

TAGS
latest
345
344
340

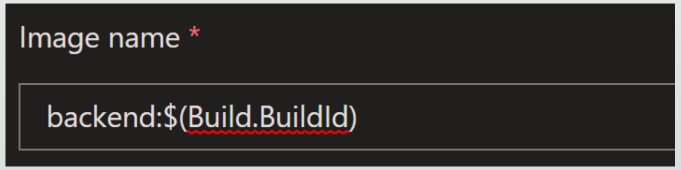


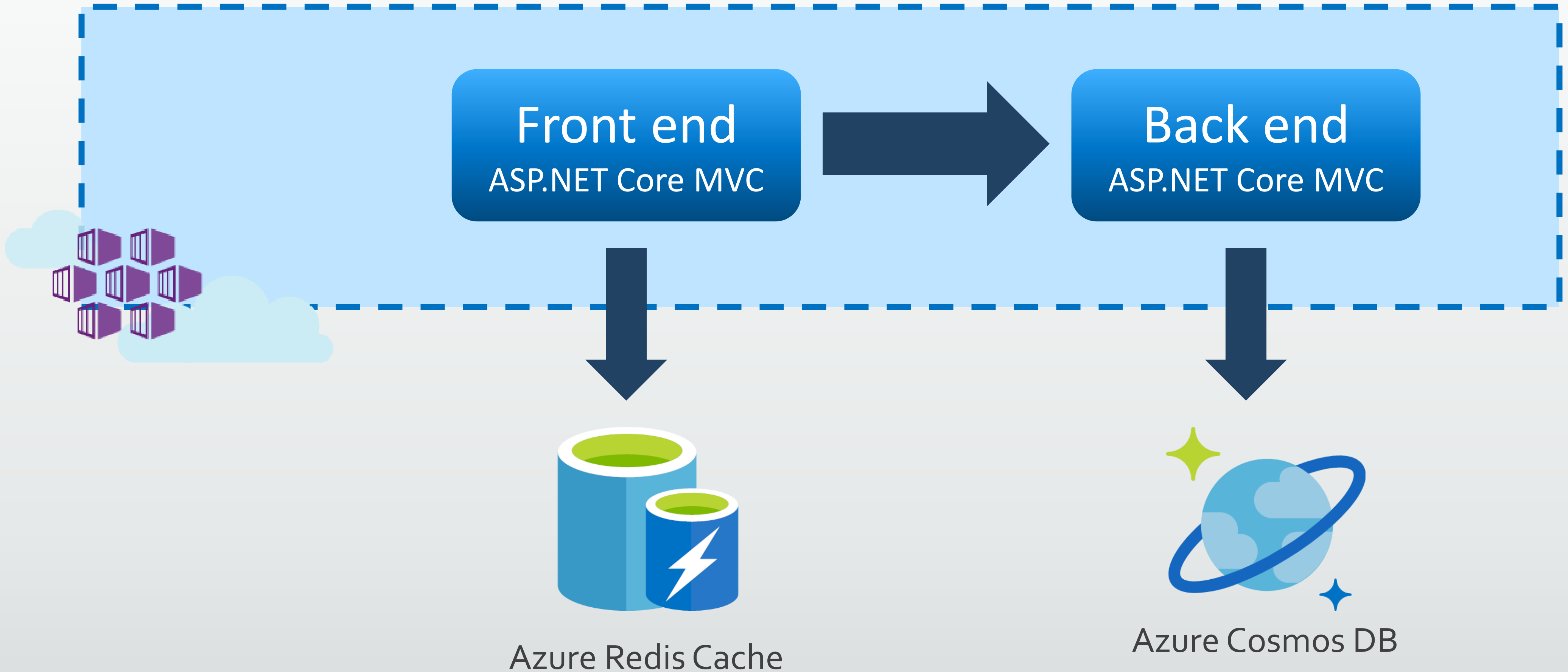
Image name *

backend:$(Build.BuildId)

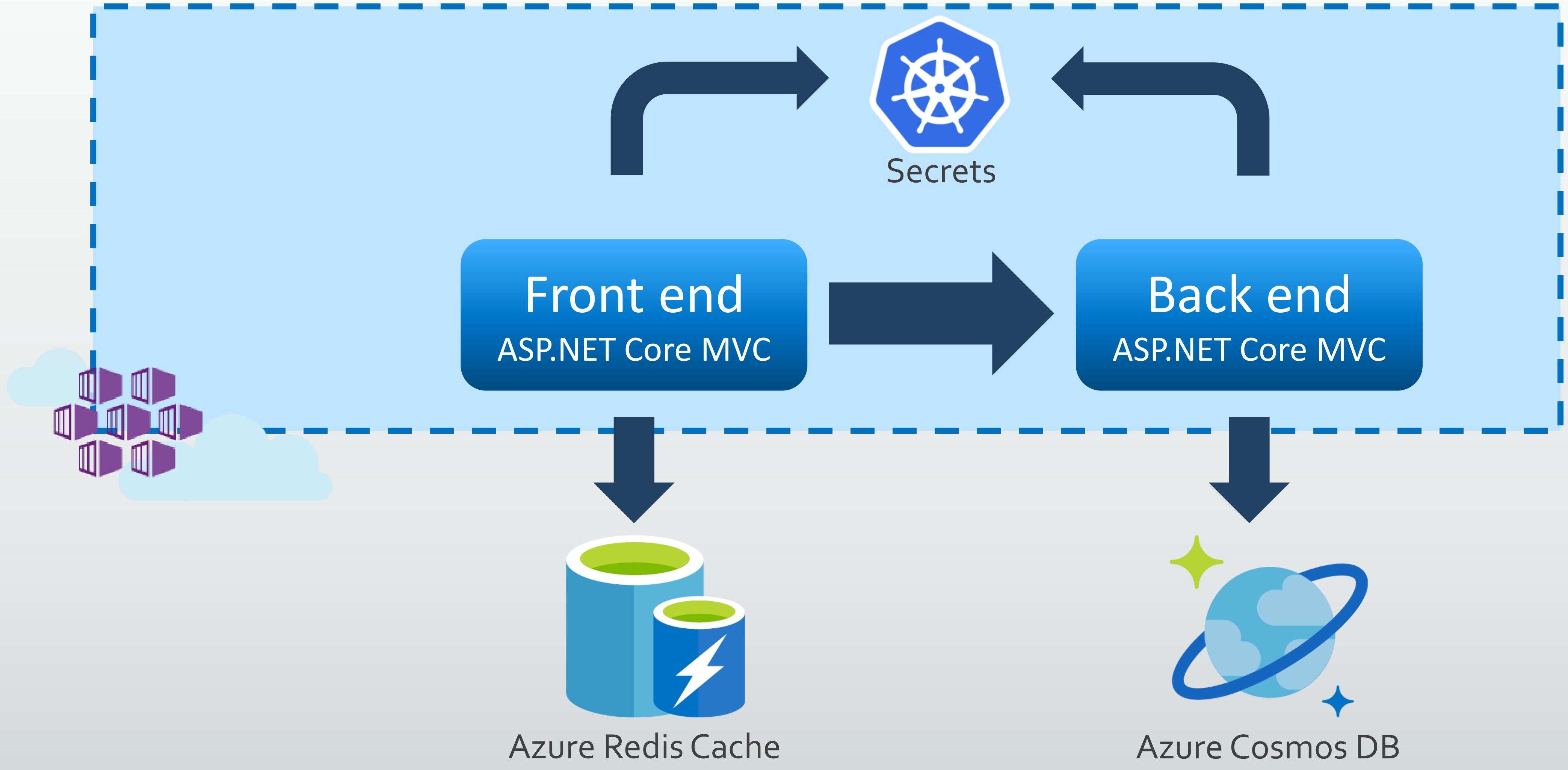
Must Have settings

- RBAC
- Advanced Networking

Our application

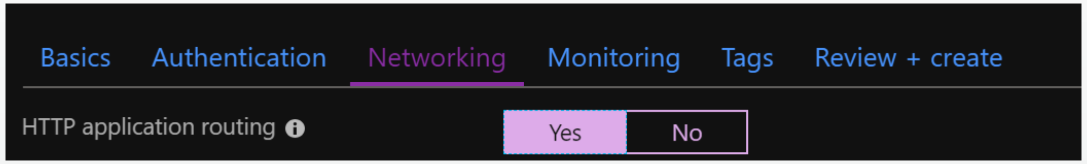


Where do we store our connection strings?



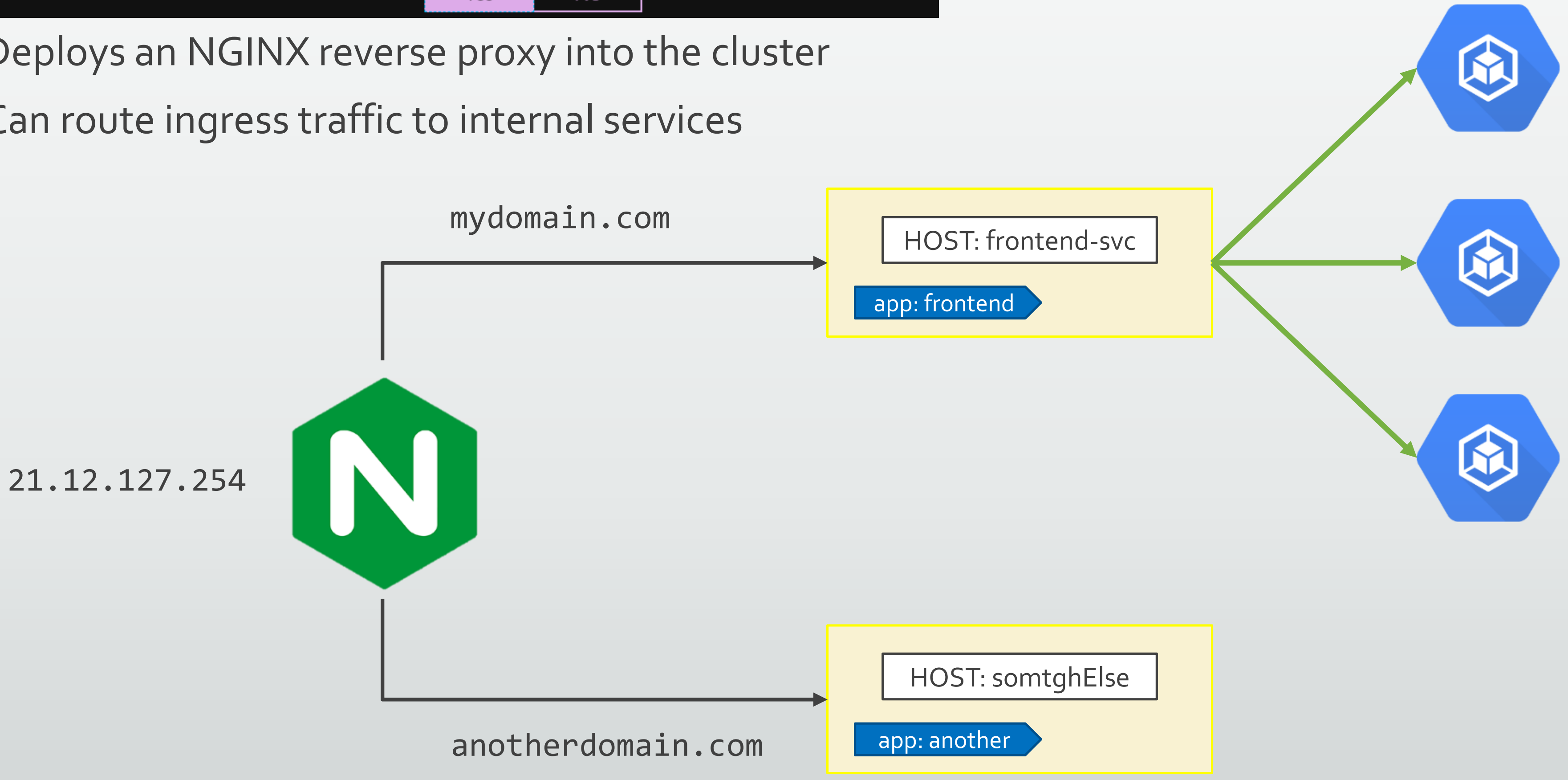
Ingress controller

- Installed and configured by HTTP-Application-Routing



HTTP application routing [✓ Yes] No

- Deploys an NGINX reverse proxy into the cluster
- Can route ingress traffic to internal services



Azure DevOps integration

Demo

Bring your own domain under HTTPS in 3 steps

1) Install **CertManager** on the Cluster to add the capability of requesting certificates

```
helm install stable/cert-manager \
    --namespace kube-system \
    --set ingressShim.defaultIssuerName=letsencrypt-prod \
    --set ingressShim.defaultIssuerKind=ClusterIssuer
```

2) Create a **ClusterIssuer** object

```
apiVersion: certmanager.k8s.io/v1alpha1
kind: ClusterIssuer
metadata:
  name: letsencrypt-prod
spec:
  ...
```

3) Create a **Certificate** object

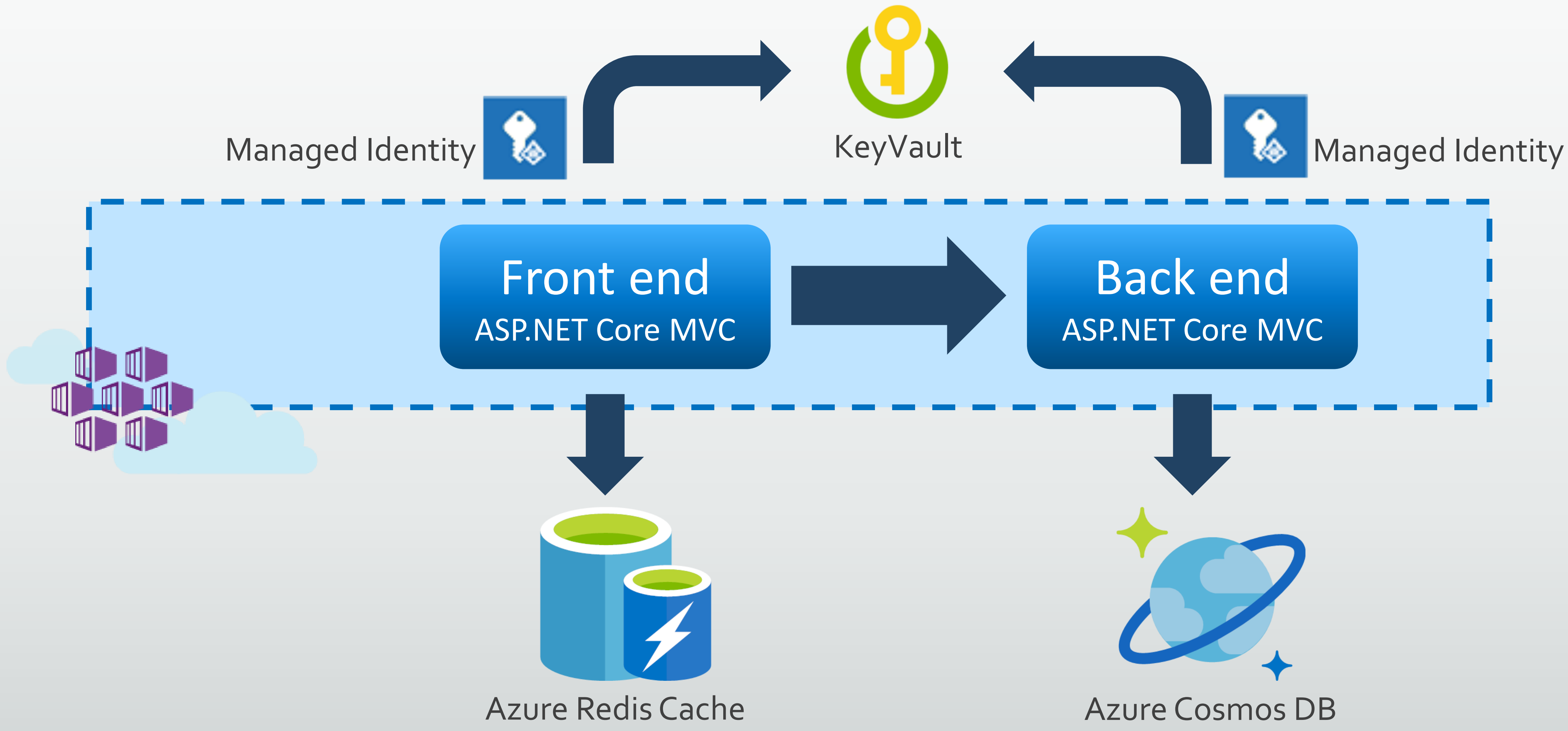
```
apiVersion: certmanager.k8s.io/v1alpha1
kind: Certificate
metadata:
  name: tls-people-secret
spec:
  issuerRef:
    name: letsencrypt-prod
...
```

https://docs.microsoft.com/en-us/azure/aks/ingress-tls

Custom domain under HTTPS

Demo

Store Connection Strings in KeyVault



Add support for Pod Identity

1) Install **Azure Pod Identity** on the Cluster

```
kubectl apply -f https://raw.githubusercontent.com/Azure/aad-pod-
identity/master/deploy/infra/deployment-rbac.yaml
```

2) Create an **AzureIdentity** object

```
apiVersion: "aadpodidentity.k8s.io/v1"
kind: AzureIdentity
metadata:
  name: demoapp-identity
spec:
  type: 0
  ResourceID: "#{managedIdentityResourceId}#"
  ClientID: #{managedIdentityClientId}#
```

3) Create an **AzureIdentityBinding** Object

```
apiVersion: "aadpodidentity.k8s.io/v1"
kind: AzureIdentityBinding
metadata:
  name: demoapp-azure-identity-binding
spec:
  AzureIdentity: demoapp-identity
  Selector: "demo"
```

```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: backend
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: backend
        aadpodidbinding: demo
```

https://github.com/Azure/aad-pod-identity

Pod Identity

Demo

Governance

- Kubernetes must be able to check the Pod health status

```
readinessProbe:
  httpGet:
    path: /healthz
    port: 80
  initialDelaySeconds: 10
  periodSeconds: 5
livenessProbe:
  httpGet:
    path: /healthz
    port: 80
  initialDelaySeconds: 10
  periodSeconds: 5
```

- `ReadinessProbe` determines when a pod is ready to accept requests
- `LivenessProbe` determines the pod health status over time

- Pods must declare the resources they need so Kubernetes can safely allocate them

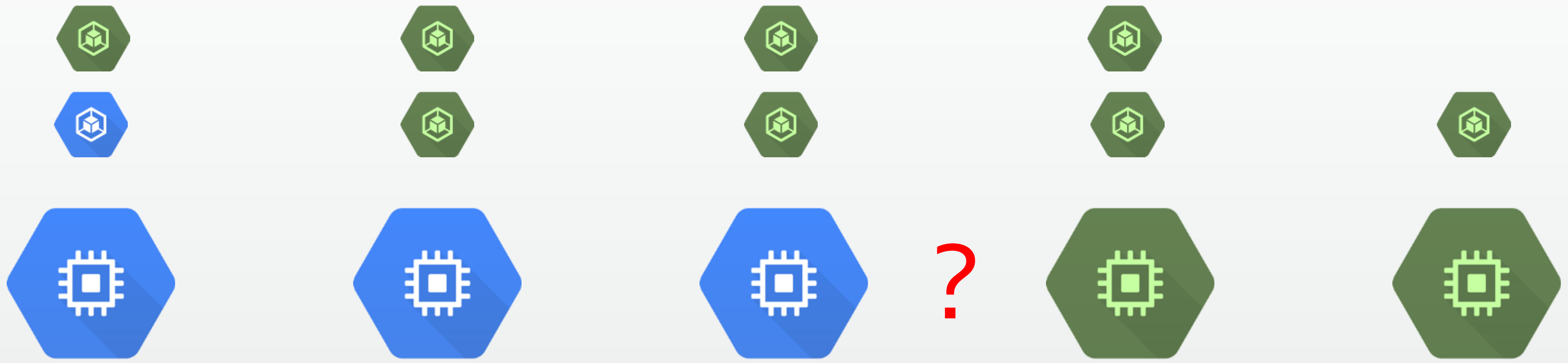
```
resources:
  requests:
    cpu: 250m
  limits:
    cpu: 350m
```

- Milli-CPUs and Memory consumption
- `Request` is used for allocation
- `Limit` is used for throttling and termination
- We can set defaults and global limits at the namespace level
- Best practices here https://goo.gl/6zQUqN

Governance (Resources and Probes)

Demo

Dealing with traffic spikes - Autoscaling



Horizontal Pod Autoscaler

- Support for **Standard**, **Custom** and **External** metrics
- https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/

Cluster Autoscaler

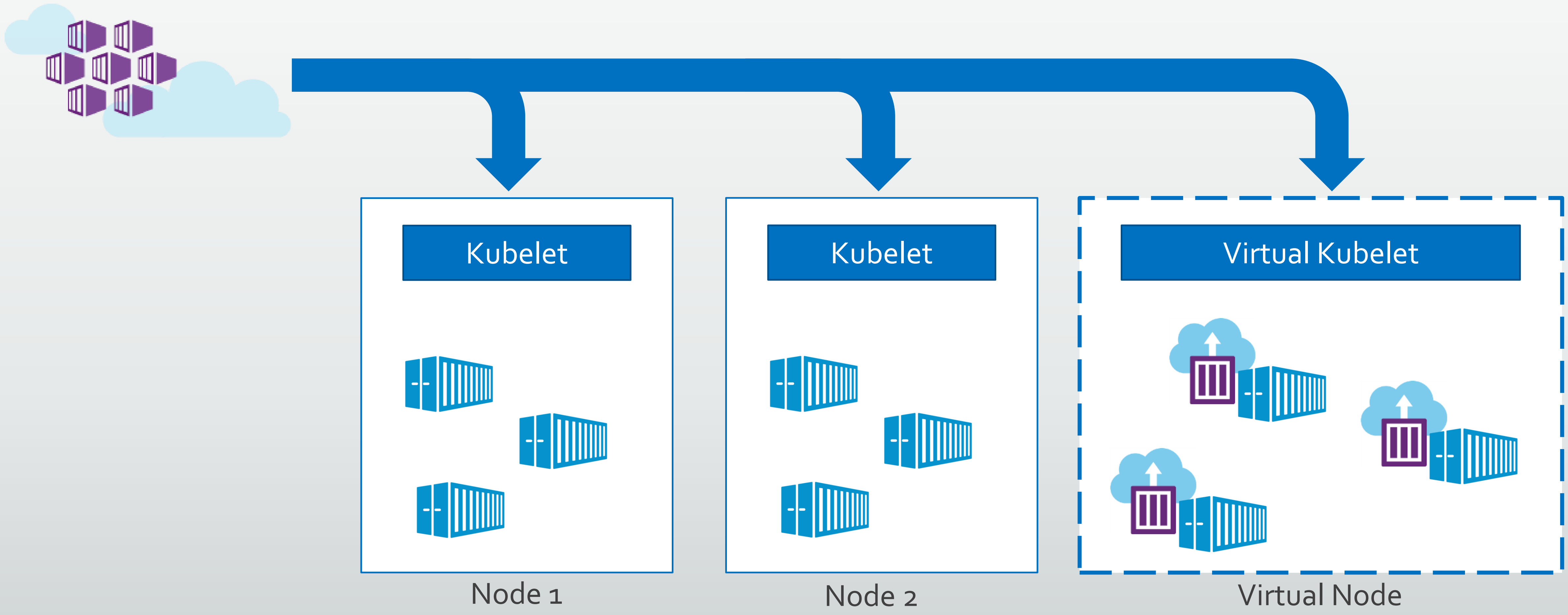
- Standalone component, in GA since version 1.0.0
- https://github.com/kubernetes/autoscaler
- https://docs.microsoft.com/en-us/azure/aks/cluster-autoscaler

Autoscale

Demo

A glimpse in the future – Virtual Node (preview)

https://docs.microsoft.com/en-us/azure/aks/virtual-nodes-portal
https://github.com/virtual-kubelet/virtual-kubelet/tree/master/providers/azure



Virtual Node

Demo

Recap

- Configured a **CI/CD pipeline** in Azure DevOps
  - https://medium.com/@marcodesanctis2/a-build-and-release-pipeline-in-vsts-for-docker-and-azure-kubernetes-service-aks-41efc9a0c5c4
  - https://medium.com/@marcodesanctis2/consume-cosmos-db-or-other-paas-services-from-azure-kubernetes-service-4ee0e304cfc1
- Leveraged **Deployment** object to roll updates
  - https://kubernetes.io/docs/concepts/workloads/controllers/deployment/
- Used **CertManager** and **Ingress** controller to bring our own domain under HTTPS
  - https://docs.microsoft.com/en-us/azure/aks/ingress-tls
- Accessed secrets KeyVault using **Pod Identity**
  - https://github.com/Azure/aad-pod-identity
- Set **resources** and **probes**
  - https://kubernetes.io/docs/concepts/configuration/manage-compute-resources-container/
- Managed traffic spikes through **Horizontal Pod Autoscaler** and **Cluster Autoscaler**
  - https://docs.microsoft.com/en-us/azure/aks/tutorial-kubernetes-scale
  - https://docs.microsoft.com/en-us/azure/aks/cluster-autoscaler
- Had a glimpse at **Virtual Node**

Thank you!

@crad77

info@marcodesanctis.it

Get the code at
https://github.com/cradle77/AksAdvanced