[Kubernetes Tutorial for Beginners [FULL COURSE in 4 Hours] - YouTube](https://www.youtube.com/watch?v=X48VuDVv0do)

<https://www.youtube.com/watch?v=X48VuDVv0do>

<https://github.com/in28minutes/devops-master-class>

<https://github.com/in28minutes/devops-master-class>

# Kubernetes Microservices

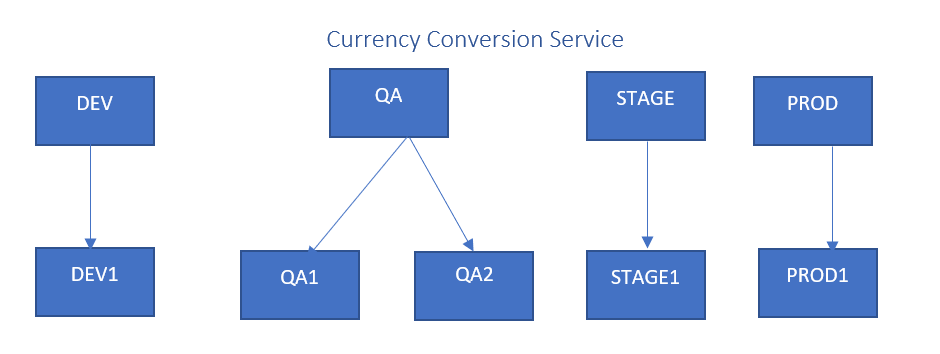
## Deploy Microservices using Kubernetes

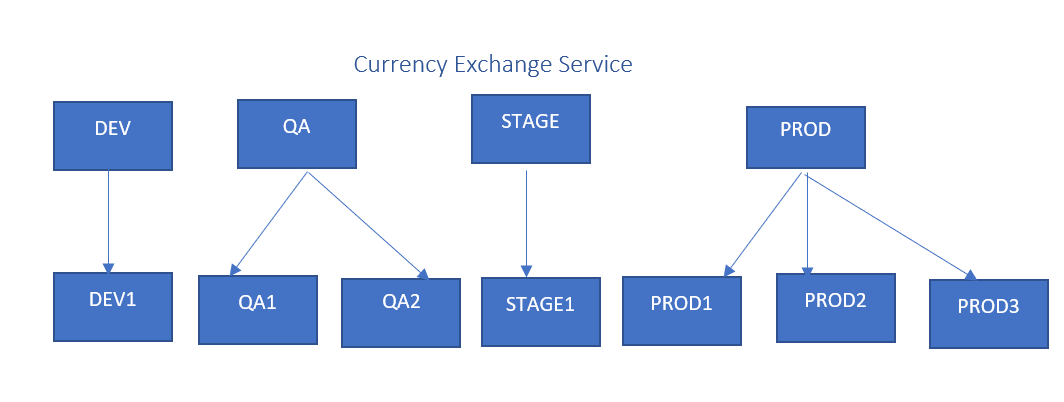
kubectl apply -f deployment.yaml

kubectl get svc –watch

# Understanding environment variables

# Microservices Centralised Configurations





## Currency Exchange Service

PROD

STAGE

QA

DEV

PROD3

PROD1

STAGE1

QA2

QA1

PROD2

DEV1

kubectl apply -f 00-configmap-currency-conversion.yaml

kubectl get configMaps

kubectl describe configmap <<configmap\_name>>

Ex: kubectl describe configmap currency-conversion-config-map

kubectl logs -f currency-conversion-5c8f698797-qf4pp

# Micro Services in Kubernetes

## Install Istio on Kubernetes

**1. Deploy the Cluser with Medium Machine as Istio need memory to start and work.**

kops create cluster --state="s3://kops-bucket-a87654" --zones="ap-south-1a,ap-south-1b" --node-size=t2.medium --master-size=t2.micro --master-count 1 --node-count 2 --authorization=RBAC --name=level360degree.uk --yes

Note: Please make sure you will update the S3 Bucket location as per your own Account.

**2. Validate Cluster is Running**

kops validate cluster --state=s3://kops-bucket-a87654

Note: Please make sure you will update the S3 Bucket location as per your own Account.

**3. Install Helm on your Cluster**

wget https://storage.googleapis.com/kubernetes-helm/helm-v2.14.1-linux-amd64.tar.gz

tar -xzvf helm-v2.14.1-linux-amd64.tar.gz

sudo mv linux-amd64/helm /usr/local/bin/helm

**4. Verify Helm**

helm -h

**5. Initialize helm**

kubectl create -f helm-rbac.yml

helm init --service-account helm-tiller

**6. Verify Where it deployed**

kubectl get pods -n kube-system

**7. Install Istio with HELM Package Manager**

helm repo add istio.io https://storage.googleapis.com/istio-release/releases/1.3.0/charts/

(This will enable you to use the Helm charts in the repository to install Istio.)

**8. Check that you have the repo:**

helm repo list

**9. Install Istio’s Custom Resource Definitions (CRDs) with the istio-init chart.**

helm install --name istio-init --namespace istio-system istio.io/istio-init

(This command commits 53 CRDs to the kube-apiserver, making them available for use in the Istio mesh.)

**10. To check that all of the required CRDs have been committed, run the following command:**

kubectl get crds | grep 'istio.io\|certmanager.k8s.io'

kubectl get crds | grep 'istio.io\|certmanager.k8s.io' | wc -l

**11. Now we will Install Istio Chart. To ensure that the Grafana telemetry addon is installed with the chart, we will use the --set grafana.enabled=true configuration option with our helm install command.**

helm install --name istio --namespace istio-system --set grafana.enabled=true istio.io/istio

**12. User can verify that the Service objects we expect for the default profile have been created with the following command:**

kubectl get svc -n istio-system

**13. We can also check for the corresponding Istio Pods with the following command:**

kubectl get pods -n istio-system

Note : If you see unexpected phases in the STATUS column, remember that you can troubleshoot your Pods with the following commands:

kubectl describe pods your\_pod -n pod\_namespace

kubectl logs your\_pod -n pod\_namespace

**14. The final step in the Istio installation will be enabling the creation of Envoy proxies, which will be deployed as sidecars to services running in the mesh.**

There are two ways of accomplishing this goal: manual sidecar injection and automatic sidecar injection. We’ll enable automatic sidecar injection by labeling the namespace in which we will create our application objects with the label istio-injection=enabled.

We’ll use the default namespace to create our application objects, so we’ll apply the istio-injection=enabled label to that namespace with the following command:

kubectl label namespace default istio-injection=enabled

**15. We can verify that the command worked as intended by running:**

kubectl get namespace -L istio-injection