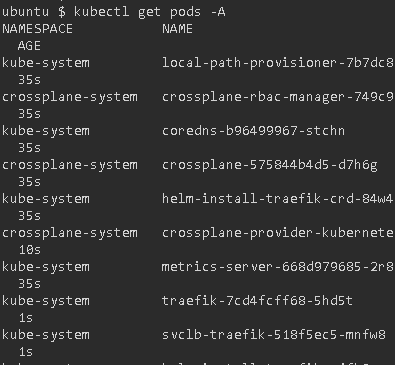
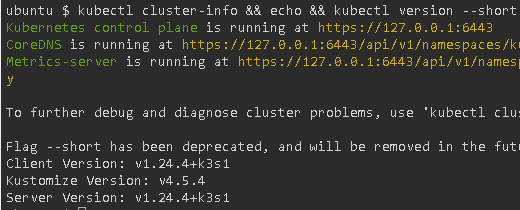
# Crossplane

# Interactive Crossplane Workshop

Check the cluster setup kubectl get pods –A



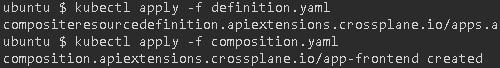
Check Kuberentes cluster info and version kubectl cluster-info && echo && kubectl version –short



## Composed App in Action

kubectl apply -f definition.yaml

kubectl apply -f composition.yaml



kubectl get xrd



kubectl get compositions



kubectl create ns devops-team

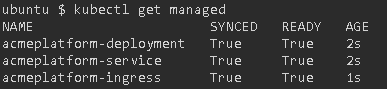


kubectl apply -f app-claim.yaml

kubectl wait deployment.apps/acmeplatform --namespace devops-team --for condition=AVAILABLE=True --timeout 1m



kubectl get managed



kubectl port-forward deployment/acmeplatform -n devops-team --address 0.0.0.0 8080:80



## Application Lifecycle

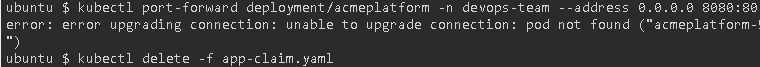
kubectl apply -f app-claim-blue.yaml

kubectl wait deployment.apps/acmeplatform --namespace devops-team --for condition=AVAILABLE=True --timeout 1m



kubectl port-forward deployment/acmeplatform -n devops-team --address 0.0.0.0 8080:80

kubectl delete -f app-claim.yaml

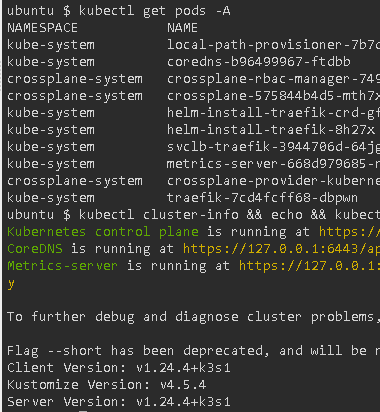


# Validate Crossplane compositions with Datree

## Setting up the Environment

kubectl get pods –A

kubectl cluster-info && echo && kubectl version –short



## Introduction to Datree

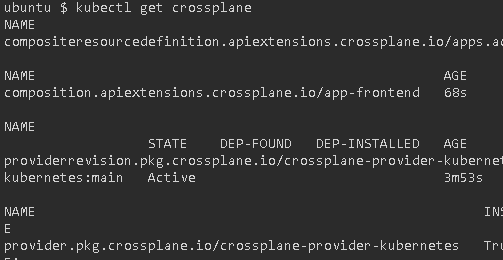
Datree prevents Kubernetes misconfigurations from reaching production.

## Prepare the resources

* kubectl apply -f definition.yaml
* kubectl apply -f composition.yaml
* kubectl create ns devops-team
* kubectl apply -f app-claim.yaml



kubectl get crossplane



kubectl describe service acmeplatform -n devops-team | grep ^Labels -A2

kubectl describe deployment acmeplatform -n devops-team | grep ^Labels -A2

kubectl describe hpa acmeplatform -n devops-team | grep ^Labels -A2

