1. Deploy a 2-Node K8S Cluster Using Kops
2. Clone or Fork the repo <https://github.com/mavrick202/argocdtesting.git>
3. Visit following URL for step-by-step deployment of ArgoCD

<https://www.eksworkshop.com/intermediate/290_argocd/install/>

ArgoCD Documentation:

<https://argoproj.github.io/argo-cd/getting_started/>

<https://luktom.net/en/e1683-argocd-vs-flux>

Deploying ArgoCD and Deploying application

kubectl create namespace argocd

kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml

Download Latest ArgoCD using below:

VERSION=$(curl --silent "https://api.github.com/repos/argoproj/argo-cd/releases/latest" | grep '"tag\_name"' | sed -E 's/.\*"([^"]+)".\*/\1/')

sudo curl --silent --location -o /usr/local/bin/argocd <https://github.com/argoproj/argo-cd/releases/download/$VERSION/argocd-linux-amd64>

sudo chmod +x /usr/local/bin/argocd

kubectl patch svc argocd-server -n argocd -p '{"spec": {"type": "LoadBalancer"}}'

Wait for 5 min to deploy classic load balancer and instances in to service.

export ARGOCD\_SERVER=`kubectl get svc argocd-server -n argocd -o json | jq --raw-output .status.loadBalancer.ingress[0].hostname`

--------OLD MEATHOD BEFORE 1.8 VERSION---------

ARGO\_PWD=`kubectl get pods -n argocd -l app.kubernetes.io/name=argocd-server -o name | cut -d'/' -f 2`

--------------------------------------------------------------

-----FROM VERSION 1.9 & Later--------------------------------

kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath="{.data.password}" | base64 -d

------------------------------------------------------------

argocd login $ARGOCD\_SERVER --username admin --password <Password> --insecure

CONTEXT\_NAME=`kubectl config view -o jsonpath='{.contexts[].name}'`

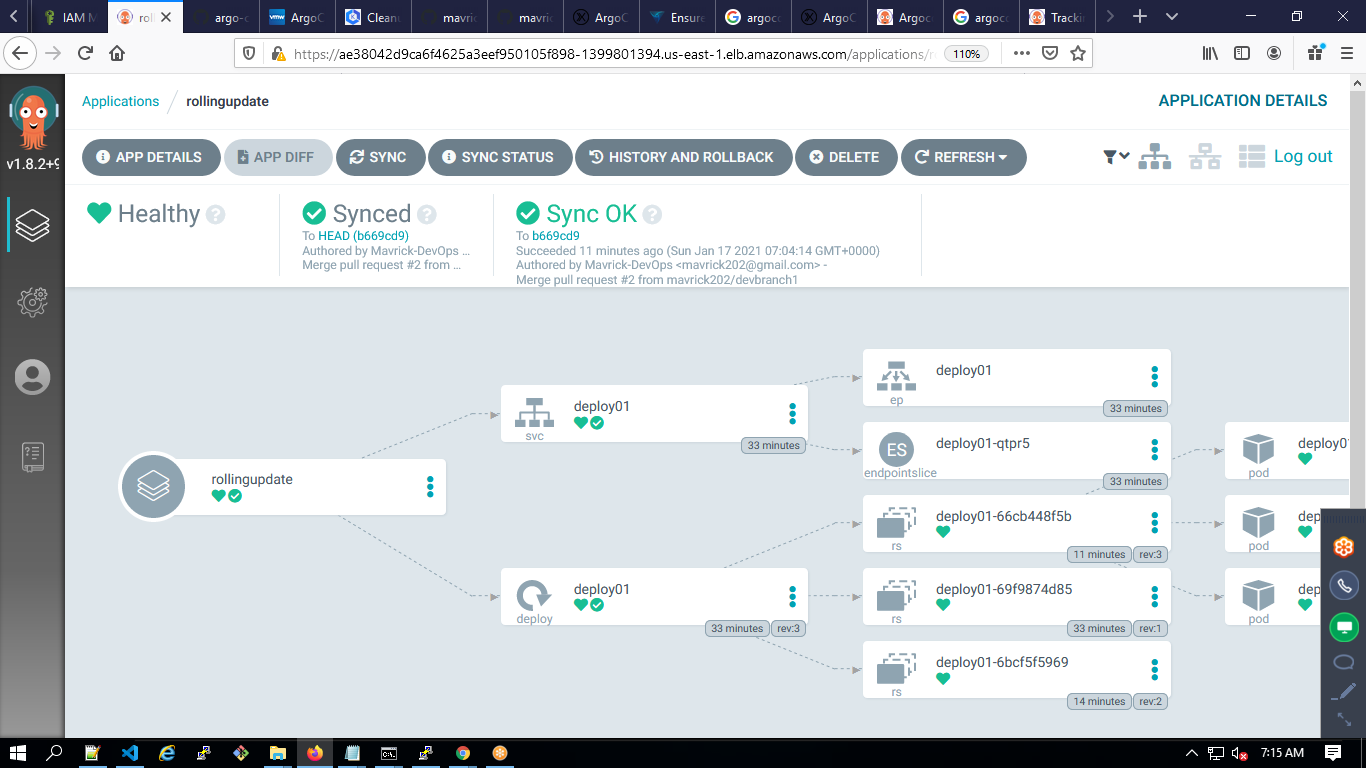
argocd cluster add $CONTEXT\_NAME

argocd app create rollingupdate --repo https://github.com/mavrick202/argocdtesting.git --path deploy --dest-server https://api.trainingk8s.xyz(Replace-with-ur-cluster) --dest-namespace default

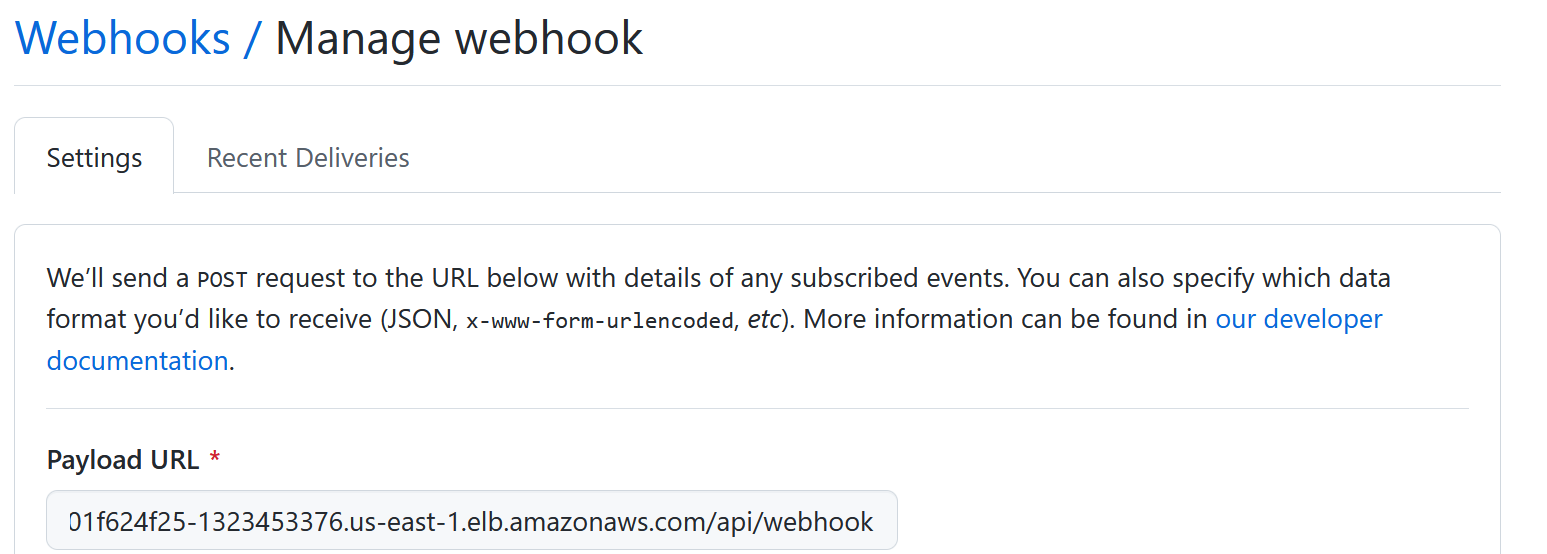
argocd app create votingapp --repo https://github.com/mavrick202/argocdtesting.git --path votingapp --dest-server https://api.trainingk8s.xyz --dest-namespace default

argocd app get rollingupdate (or) Can be done fom the GUI

argocd app sync rollingupdate (or) Can be done fom the GUI



Argo CD polls Git repositories every three minutes to detect changes to the manifests. To eliminate this delay from polling, the API server can be configured to receive webhook events. Argo CD supports Git webhook notifications from GitHub, GitLab, Bitbucket, Bitbucket Server and Gogs. The following explains how to configure a Git webhook for GitHub, but the same process should be applicable to other providers.



We can also enable auto sync by enabling it as below.

*argocd app list*

*argocd app set rollingupdate --sync-policy automated*

*argocd app delete votingapp rollingupdate --yes*

Argocd vs Flux:

*https://blog.container-solutions.com/fluxcd-argocd-jenkins-x-gitops-tools*

Argocd Jenkins Pipeline:

<https://yetiops.net/posts/argocd-jenkins-pipeline/>

<https://blog.argoproj.io/migrating-from-jenkins-to-argo-at-sendible-2ad4268837e9>

Using Azure Devops:

1. Create SP using Kubeconfig.
2. Use the repo with a folder deploy and inside create deployment.yaml
3. Kubectl apply -f ./\_azureb13terraformaws/deploy/

<https://medium.com/hootsuite-engineering/using-gitops-argocd-to-ship-kubernetes-changes-faster-at-hootsuite-4d35628a3fb7>

<https://www.dataversity.net/openshift-vs-kubernetes-the-seven-most-critical-differences/>