Computer Science & Information Systems

**DevOps for Cloud - Lab Sheet 5 - Module 7**

**(M7: Continuous Deployment)**

This lab sheet needs to be administered along with Module 7: Continuous Deployment

**Notation used in the document**

* ‘>’ represents the terminal, where we type the commands.
* The text mentioned within ‘[‘ and ‘]’ brackets provides additional documentation for the step.

1. Objectives:
2. To demonstrate the GitOps based continuous deployment process using Argo CD tool
3. Pre-requisite:
4. Minikube is installed and running
5. Kubectl is installed
6. Container image is available in DockerHub (either custom image or predefined image).
7. Lab Exercise:

[Ref URL - <https://argo-cd.readthedocs.io/en/stable/getting_started/> ]

**Task 1: Install Argo CD and open the Argo UI**

1. Open terminal (command prompt) and type the following commands

> minikube start

> kubectl create namespace argocd

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

[Argo CD is now running as an Agent “within the Minikube cluster”. It has access to all components within the Minikube]

Additionally, one can install Argo CD CLI at the URL <https://github.com/argoproj/argo-cd/releases/tag/v2.8.16>

For Windows: Rename argocd-windows-amd64.exe to argocd and make it available in path before running the argocd cli.

1. Check whether all Argo Pods are running

> kubectl get pods -n argocd [Check if the argocd pods are running, otherwise wait]

1. Provide Username and Password for login to Argo UI

Username: “admin”

Password: > argocd admin initial-password -n argocd

1. Port Forwarding [Connect to the API Server]

> kubectl port-forward svc/argocd-server -n argocd 8080:443

[Open the browser and type localhost:8080 🡪 opens the ArgoCD homepage]

**Task 2: Demonstrate CD for Nginx application [pre-defined image from dockerhub]**

*Step 1: Create a Manifest file and push to GitHub repo*

* Create a new folder called “nginx-deploy”. Run “git init” command on the folder. Create a subfolder called “manifest” and place the below yaml file in that folder.
* The “nginx-deployment.yaml” file is shown below

# File: nginx-deployment.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

  name: nginx-deployment

  labels:

    app: nginx

  namespace: default

spec:

  replicas: 2

  selector:

    matchLabels:

      app: nginx

  template:

    metadata:

      labels:

        app: nginx

    spec:

      containers:

      - name: nginx

        image: nginx:latest

        ports:

        - containerPort: 80

---

# Nodeport service to expose the application

apiVersion: v1

kind: Service

metadata:

  name: nginx-service

  namespace: default

spec:

  selector:

    app: nginx

  ports:

    - protocol: TCP

      port: 80

      targetPort: 80

  type: NodePort

* Login to GitHub, create a repo called “nginx-deploy” and establish connection between your local repo and GitHub repo.

*Step 2: Create Argo CD Application in ArgoCD UI*

* Open the browser and type localhost:8080 🡪 opens the ArgoCD homepage
* Click on “NEW APP” and provide the following details under “GENERAL” category
  + Application Name: nginx-app [all lower case]
  + Project Name: default
  + SYNC POLICY: Manual
* Under “SOURCE” category
  + Repository URL - <https://github.com/shreyassureshrao/nginx-deploy.git> [replace by your specific repo name in GitHub] and select “GIT” in the dropdown [default]
  + Revision: HEAD [Refers to Master branch] and select “BRANCHES” in the dropdown [default]
  + Path: manifest [this will refer to the “manifest” folder created in the repo which hosts the yaml file]
* Under “DESTINATION” category
  + Cluster URL: <https://kubernetes.default.svc> and select “URL” in dropdown [default]
  + Namespace: <leave this field empty…don’t fill>
* Under “DIRECTORY” category
  + Check “DIRECTORY RECURSE”
* Click on “Create” button in the top.
* [nginx-app application will be created. Verify the following statuses:
* App Health: Healthy
* Sync Status: Synced
* Pods will be created and running]

*Step 3: Make changes to the manifest file and view the CD in action*

* In the yaml file, change ReplicaSet from 2 to 3
* Push the changes to GitHub
* In the Argo CD UI, click on “Sync” button and then “Synchronize”
* We can see that a new pod will be created in the minikube
* [This demonstrates both “Declarative” mode of behaviour and auto-sync features, applied to kubernetes cluster.]

*Step 4: Run the nginx service and access the pods*

* Create a tunnel using the NodePort

> minikube service nginx-service --url [replace with your service name mentioned in the yaml file]

* [Ex: http://127.0.0.1:58061 -> open in the browser as localhost:58061 and we can see “Welcome to Nginx” homepage]

*Step 5: Release all Argo resources from Minikube [optional]*

* > kubectl delete -n argocd --all all
* > kubectl delete namespace argocd

4. Outputs/Results:

Students are expected to perform the tasks provided in the lab capsule, and thereby gain a practical understanding of the concept of GitOps based Continuous Deployment using Argo CD as an agent.

1. Observations:

* None

References:

* <https://argo-cd.readthedocs.io/en/stable/getting_started/>
* <https://argoproj.github.io/cd/>