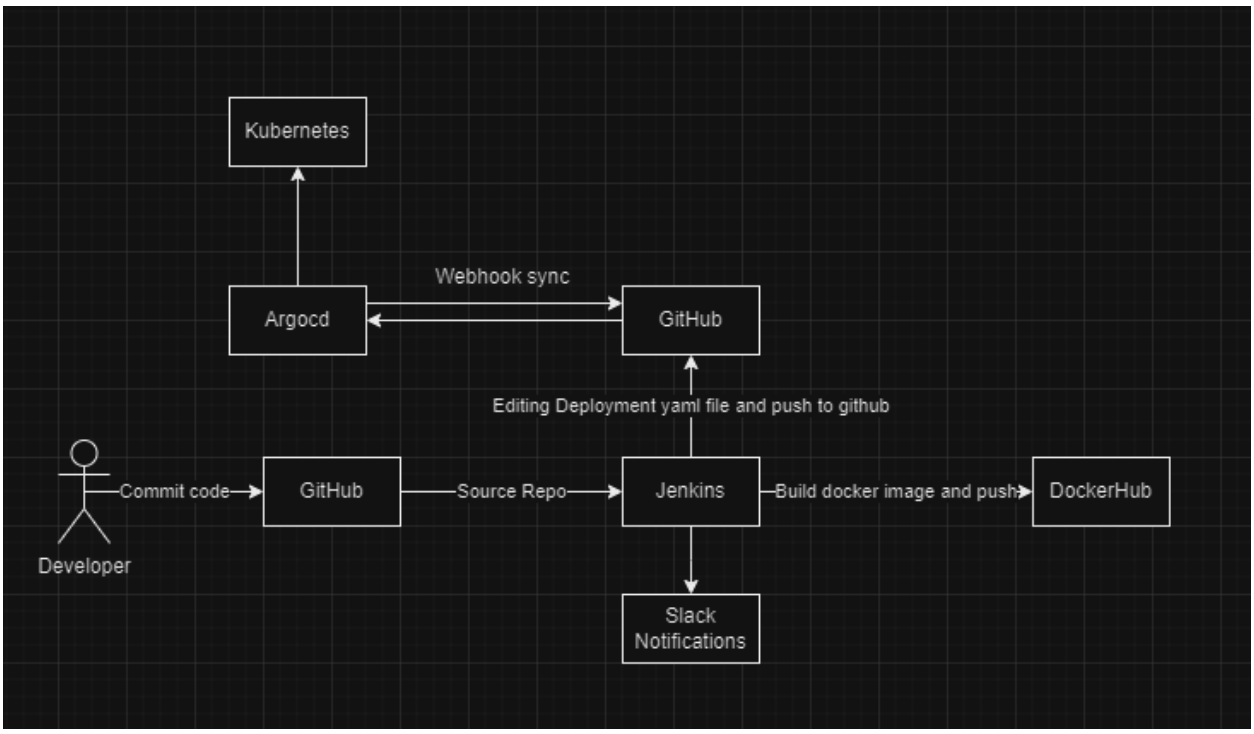
# DevOps Project

The Flow of our project :




Creating two machines and connecting them with Jenkins agent.

1- Docker, Jenkins, Jfrog
2- Kubernetes , Argocd

For Jenkins stages:

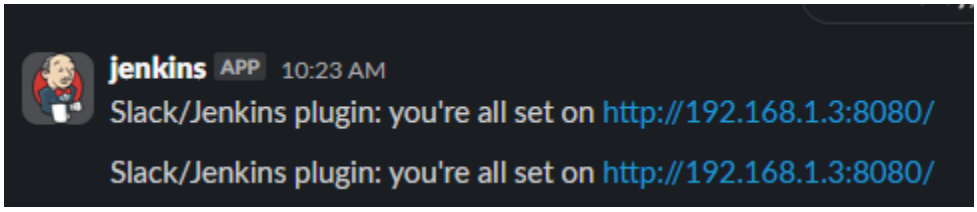
**Stage View**

| | Declarative: Checkout SCM | Cleanup Workspace | Checkout SCM on github | Build Docker Image | Push Docker Image To DockerHub | Delete Docker Images | Updating Kubernetes deployment file | Push the changed deployment file to Git | Argocd login | Declarative: Post Actions |
|---|---|---|---|---|---|---|---|---|---|---|
| Average stage times: (Average full run time: ~1min 45s) | 3s | 462ms | 3s | 16s | 42s | 865ms | 1s | 5s | 23s | 2s |
| #121 Jun 10 11:36 3 commits | 3s | 462ms | 3s | 16s | 42s | 865ms | 1s | 5s | 23s | 2s |

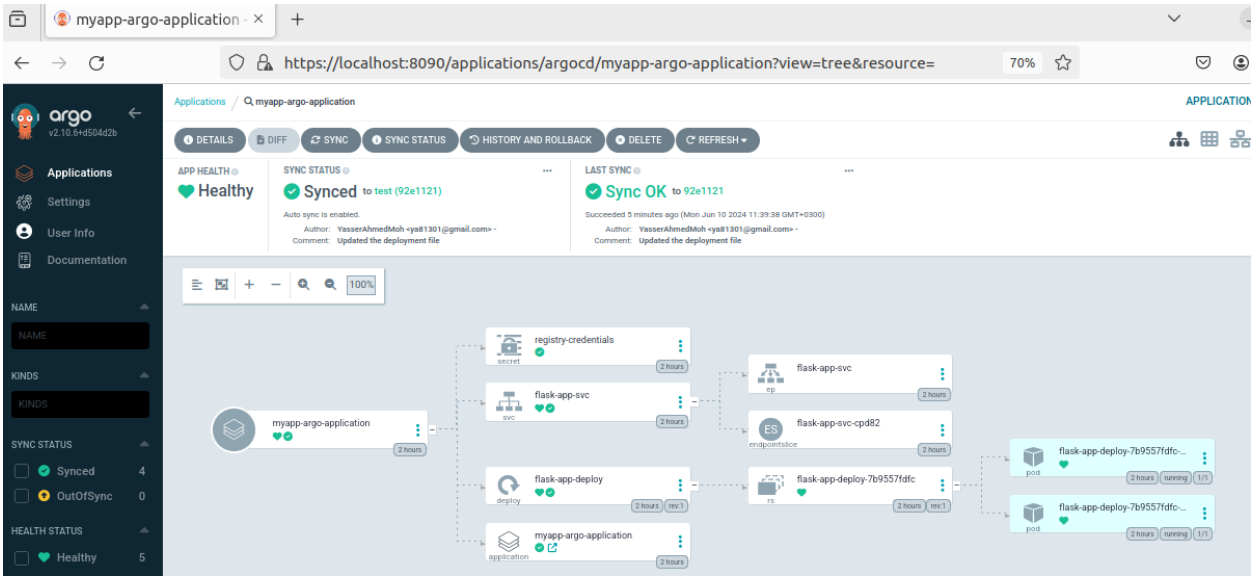
**For DockerHub:**

| Tag | OS | Type | Pulled | Pushed |
|---|---|---|---|---|
| latest | | Image | 2 hours ago | 4 minutes ago |
| 121 | | Image | 2 hours ago | 5 minutes ago |

**For Slack Notifications:**




**For Argocd:**

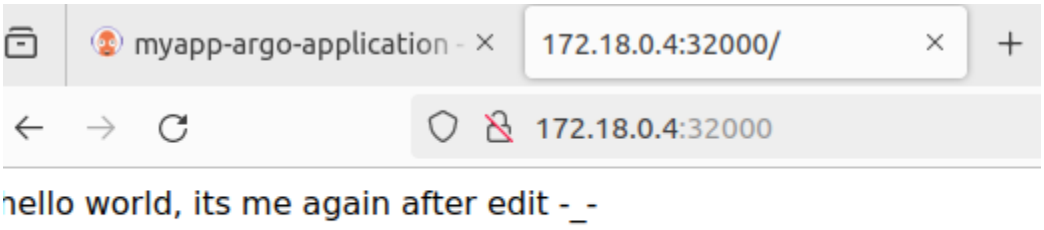


**Finally, Open application**

Use this command to get worker node ip.

```
Kubectl get nodes -o wide
```

And for port it will be as defined in deployment yaml file 32000.



# Docker

## Steps to install docker

- sudo apt update
- sudo apt install apt-transport-https ca-certificates curl software-properties-common
- curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
- echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
- sudo apt update
- apt-cache policy docker-ce
- sudo apt install docker-ce
- sudo systemctl status docker
- sudo usermod -aG docker ${USER}
- su - ${USER}
- groups
- sudo usermod -aG docker username

# Jenkins

## Steps to install Jenkins on Ubuntu 22.04 or 20.04:

### 1- Update ubuntu

- **sudo apt update && sudo apt upgrade**

### 2- install OpenJDK

- **sudo apt install default-jdk**

### 3- Add Jenkins GPG key

The packages to install are not available in the default repository of Ubuntu, hence to add its repository first add the GPG key used to sign its packages

- **sudo mkdir -p /usr/share/keyrings**
- **curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null**

### 4- Enable Jenkins repository

- **echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \ https://pkg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list > \ /dev/null**

### 5- Install Jenkins

**sudo apt install jenkins**

### 6- Check the Service status

**sudo systemctl enable --now Jenkins**

**systemctl status jenkins --no-pager -l**

### 7- Find Jenkins Administrator password

- **sudo cat/var/lib/jenkins/secrets/initialAdminPassword**

# Steps to install Jenkins on Centos 7:

## 8- Install java

9- **Wget https://download.oracle.com/java/17/latest/jdk-17_linux-x64_bin.rpm**

10- **sudo yum -y install ./jdk-17_linux-x64_bin.rpm**

11- **java -version**

## 12- Setup Jenkins repository

- **sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo**
- **curl --silent --location http://pkg.jenkins-ci.org/redhat-stable/jenkins.repo | sudo tee /etc/yum.repos.d/jenkins.repo**
- **sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key**
- **cat /etc/yum.repos.d/jenkins.repo**

## 13- install Jenkins

- **sudo yum install Jenkins**
- **sudo systemctl start Jenkins**
- **sudo systemctl enable Jenkins**
- **sudo systemctl status Jenkins**

## 14- Configure Firewall

- **firewall-cmd –permanent –zone=public –add-port=8080/tcp**
- **firewall-cmd --reload**

## Another way by using Jenkins docker image:

Docker run -d -p 8080:8080 jenkins/Jenkins:lts

## Access Web Interface

http://server-ip:8080

# jcr

### Steps to install JFrog-jcr

- docker pull docker.bintray.io/jfrog/artifactory-jcr:latest
- mkdir -p /artifactory-jcr/data
- docker run --name artifactory-jcr -d -p 8081:8081 -p 8082:8082 -v /artifactory-jcr/data:/var/opt/jfrog/artifactory  docker.bintray.io/jfrog/artifactory-jcr:latest
- chmod -R 777 /artifactory-jcr/data
- docker restart artifactory-jcr

## Access Web Interface

http:// server-ip:8082

http:// server-ip:8081

# Kind

## Steps to install kind:

- [ $(uname -m) = x86_64 ] && curl -Lo ./kind https://kind.sigs.k8s.io/dl/v0.22.0/kind-linux-amd64
- chmod +x ./kind
- sudo mv ./kind /usr/local/bin/kind

### Create folder called kind

- cd /kind
- sudo nano config.yaml
  # three node (two workers) cluster config
  kind: Cluster
  apiVersion: kind.x-k8s.io/v1alpha4
  nodes:
  - role: control-plane
  - role: worker
  - role: worker
- kind create cluster –name k8s --config config.yaml

# KUBECTL

## Steps to install kind:

**Write the following command in one line**

- curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetesrelease/release/stable.txt)/bin/linux/amd64/kubectl
- chmod +x ./kubectl
- sudo mv ./kubectl /usr/bin/kubectl
- sudo mv ./kubectl /usr/local/bin/kubectl
- kubectl version –client

# Argocd

- kubectl create namespace argocd

  **Write the following command in one line**
- kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj/argocd/stable/manifests/install.yaml
- kubectl port-forward svc/argocd-server -n argocd 8090:443

  **For argocd-server in case u want to use Node Port or ClusterIP or load balancer**
- kubectl patch svc argocd-server -n argocd -p '{"spec": {"type": "NodePort"}}'

# Argo cli

- sudo wget https://github.com/argoproj/argocd/releases/download/v2.4.14/argocd-linux-amd64
- chmod +x argocd-linux-amd64
- sudo mv argocd-linux-amd64 /usr/local/bin/argocd

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

# Access Web Interface

https:// server-ip:8090