ArgoCD

1) Setup ARGO CD

--create namespace

Kubectl create ns argocd

```
MINGW64:/c/Users/naren

naren@narendar MINGW64 ~ (master)
$ kubectl create namespace argocd
namespace/argocd created
```

```
naren@narendar MINGW64 ~ (master)
$ kubectl get ns
NAME              STATUS   AGE
argocd            Active   8h
default           Active   2d1h
kube-node-lease   Active   2d1h
kube-public       Active   2d1h
kube-system       Active   2d1h
myapp             Active   33m
```

--install argocd

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

```
naren@narendar MINGW64 ~ (master)
$ kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml

customresourcedefinition.apiextensions.k8s.io/applications.argoproj.io created
customresourcedefinition.apiextensions.k8s.io/applicationsets.argoproj.io created
customresourcedefinition.apiextensions.k8s.io/appprojects.argoproj.io created
serviceaccount/argocd-application-controller created
serviceaccount/argocd-applicationset-controller created
serviceaccount/argocd-dex-server created
serviceaccount/argocd-notifications-controller created
serviceaccount/argocd-redis created
serviceaccount/argocd-repo-server created
serviceaccount/argocd-server created
role.rbac.authorization.k8s.io/argocd-application-controller created
role.rbac.authorization.k8s.io/argocd-applicationset-controller created
role.rbac.authorization.k8s.io/argocd-dex-server created
role.rbac.authorization.k8s.io/argocd-notifications-controller created
role.rbac.authorization.k8s.io/argocd-redis created
```

--check pods

Kubectl get pods -n argocd

```
naren@narendar MINGW64 ~ (master)
$ kubectl get pods -n argocd
NAME                                                READY   STATUS    RESTARTS   AGE
argocd-application-controller-0                     1/1     Running   0          7h29m
argocd-applicationset-controller-67d7969f54-qpsth   1/1     Running   0          7h29m
argocd-dex-server-6647665474-7vm99                  1/1     Running   0          7h29m
argocd-notifications-controller-d7f66d965-rtvj2     1/1     Running   0          7h29m
argocd-redis-658ccf897d-vbsnm                       1/1     Running   0          7h29m
argocd-repo-server-67bcc6f6c7-5tq6w                 1/1     Running   0          7h29m
argocd-server-68d8fc7cf4-qs2dv                      1/1     Running   0          7h29m
```

-- Expose the Argo CD API server

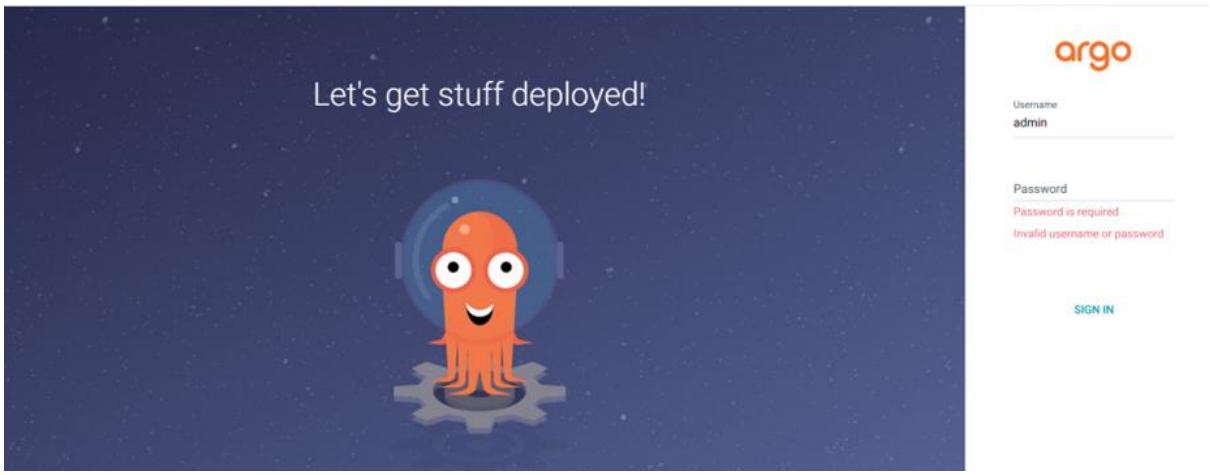
```
naren@narendar MINGW64 ~ (master)
$ kubectl patch svc argocd-server -n argocd \
>   -p '{"spec": {"type": "LoadBalancer"}}'
```

-check service

```
naren@narendar MINGW64 ~ (master)
$ kubectl get svc argocd-server -n argocd

NAME            TYPE           CLUSTER-IP     EXTERNAL-IP                                                              PORT(S)
   AGE
argocd-server   LoadBalancer   10.100.36.6    a338c9ebfcabc45abbcb5076d46ebb17-1223162770.us-east-2.elb.amazonaws.com   80:30662/TCP,443:31194/TCP
   4m21s
```

-now access with worker-01

Pul ip:port



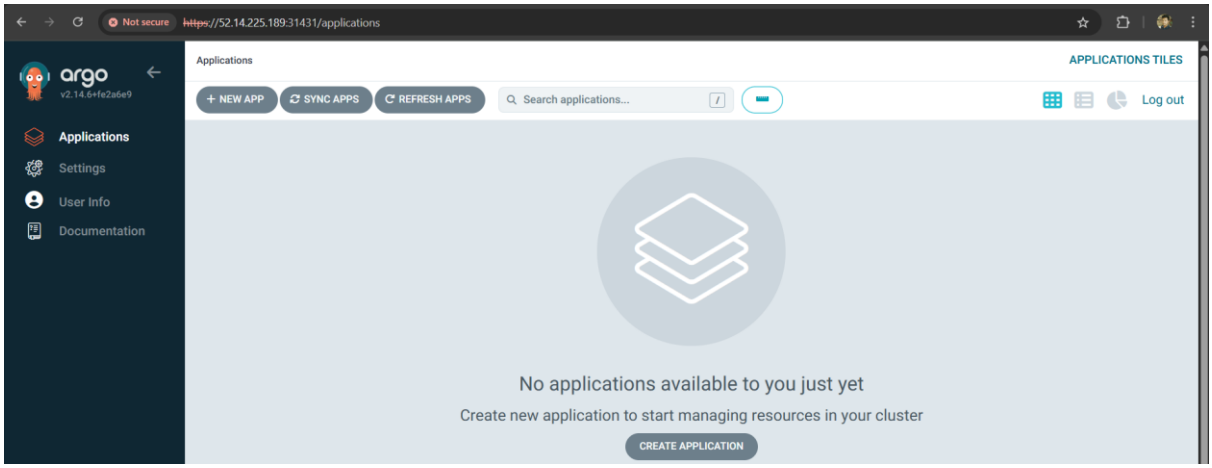
-do this for password

--curl -sSL -o argocd https://github.com/argoproj/argo-cd/releases/latest/download/argocd-linux-amd64

-- chmod +x argocd

-- sudo mv argocd /usr/local/bin/

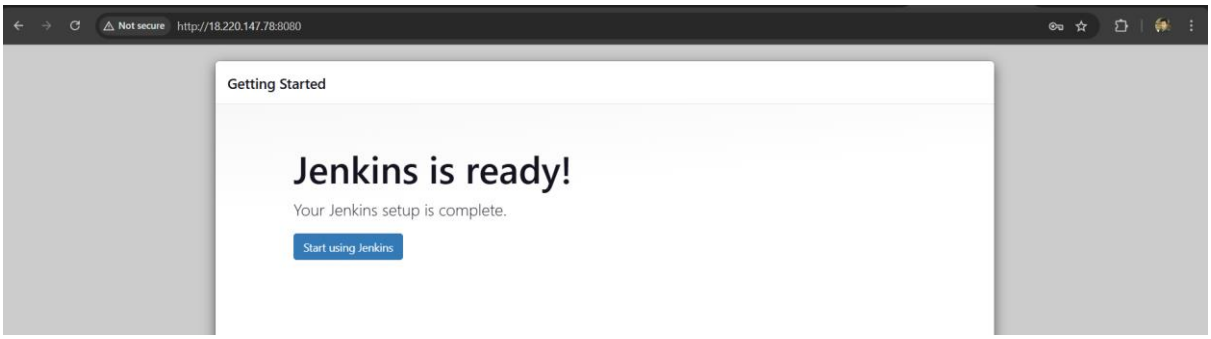
--argocd admin initial-password -n argocd



2) Create Jenkins job for CI.

Create ec2 and connect

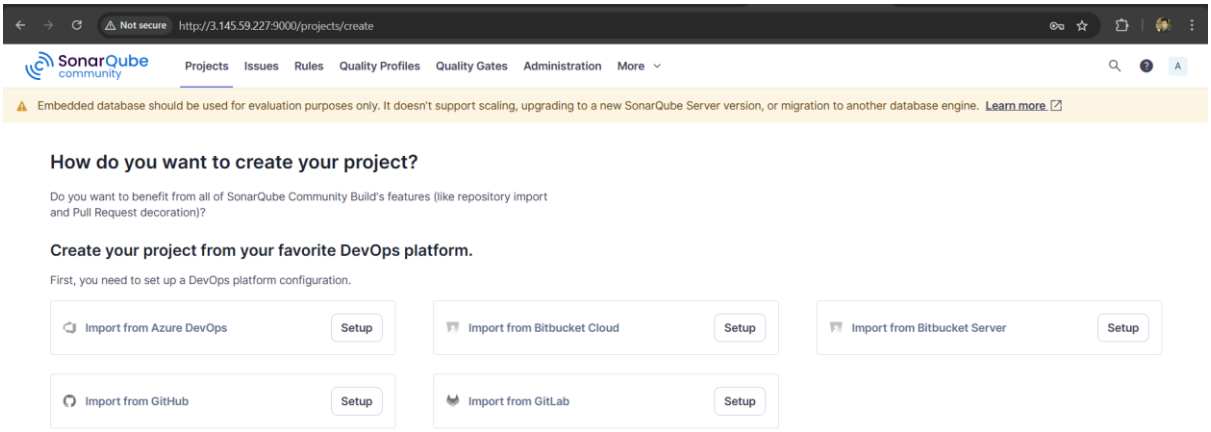
--install Jenkins



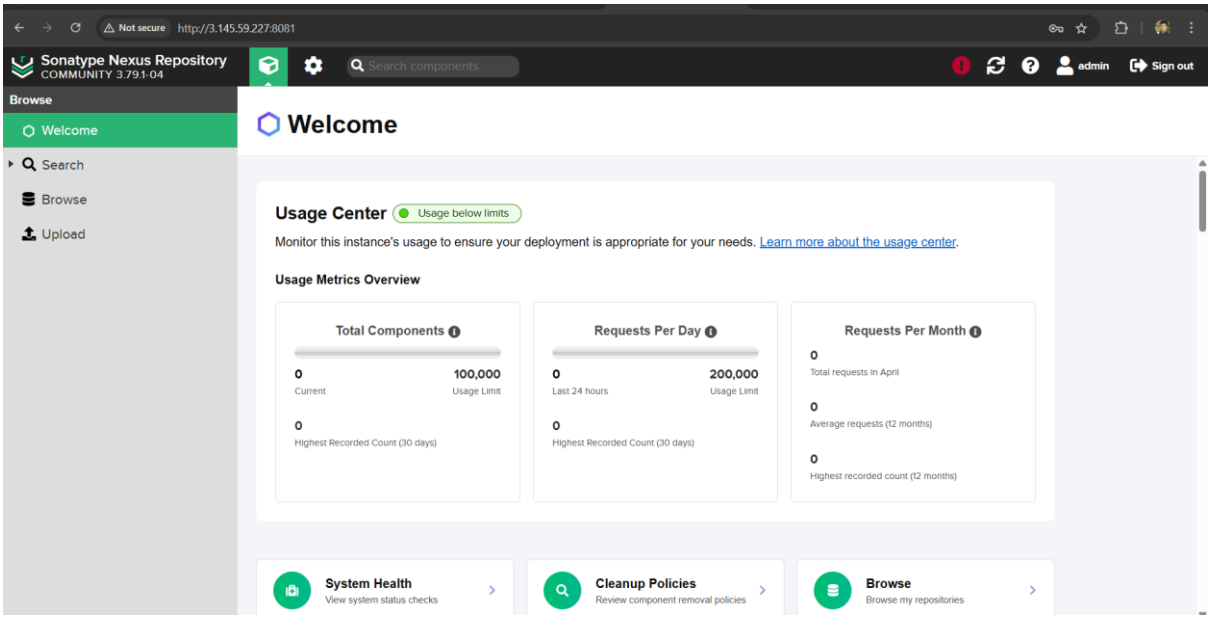
--install docker

```
[root@ip-172-31-2-219 ~]# systemctl start docker
[root@ip-172-31-2-219 ~]# systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: disabled)
     Active: active (running) since Fri 2025-04-25 12:30:08 UTC; 11s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
    Process: 28693 ExecStartPre=/bin/mkdir -p /run/docker (code=exited, status=0/SUCCESS)
    Process: 28694 ExecStartPre=/usr/libexec/docker/docker-setup-runtimes.sh (code=exited, status=0/SUCCESS)
   Main PID: 28695 (dockerd)
      Tasks: 10
     Memory: 29.7M
        CPU: 369ms
     CGroup: /system.slice/docker.service
             └─28695 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Apr 25 12:30:07 ip-172-31-2-219.us-east-2.compute.internal systemd[1]: Starting docker.service - Docker Application Container Engine...
Apr 25 12:30:07 ip-172-31-2-219.us-east-2.compute.internal dockerd[28695]: time="2025-04-25T12:30:07.495784360Z" level=info msg="Starting up"
Apr 25 12:30:07 ip-172-31-2-219.us-east-2.compute.internal dockerd[28695]: time="2025-04-25T12:30:07.537652047Z" level=info msg="Loading containers: start."
Apr 25 12:30:08 ip-172-31-2-219.us-east-2.compute.internal dockerd[28695]: time="2025-04-25T12:30:08.062764012Z" level=info msg="Loading containers: done."
Apr 25 12:30:08 ip-172-31-2-219.us-east-2.compute.internal dockerd[28695]: time="2025-04-25T12:30:08.080254504Z" level=info msg="Docker daemon" commit=71907c
Apr 25 12:30:08 ip-172-31-2-219.us-east-2.compute.internal dockerd[28695]: time="2025-04-25T12:30:08.080372481Z" level=info msg="Daemon has completed initial
Apr 25 12:30:08 ip-172-31-2-219.us-east-2.compute.internal dockerd[28695]: time="2025-04-25T12:30:08.122319653Z" level=info msg="API listen on /run/docker.so
Apr 25 12:30:08 ip-172-31-2-219.us-east-2.compute.internal systemd[1]: Started docker.service - Docker Application Container Engine.
```

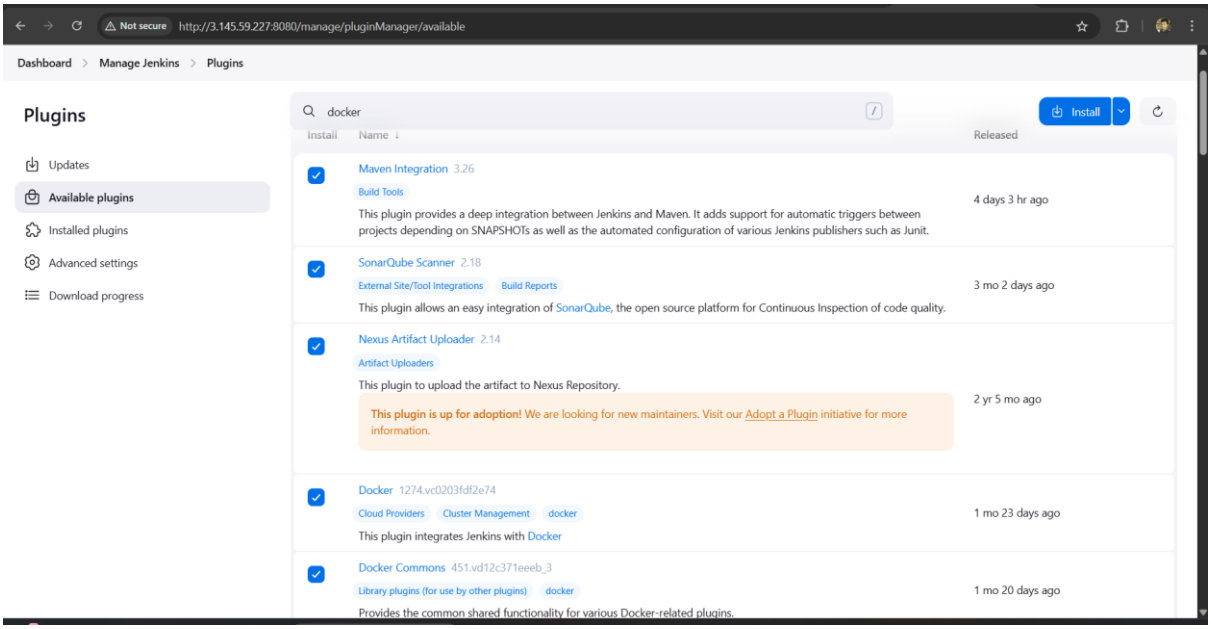
--install sonarqube



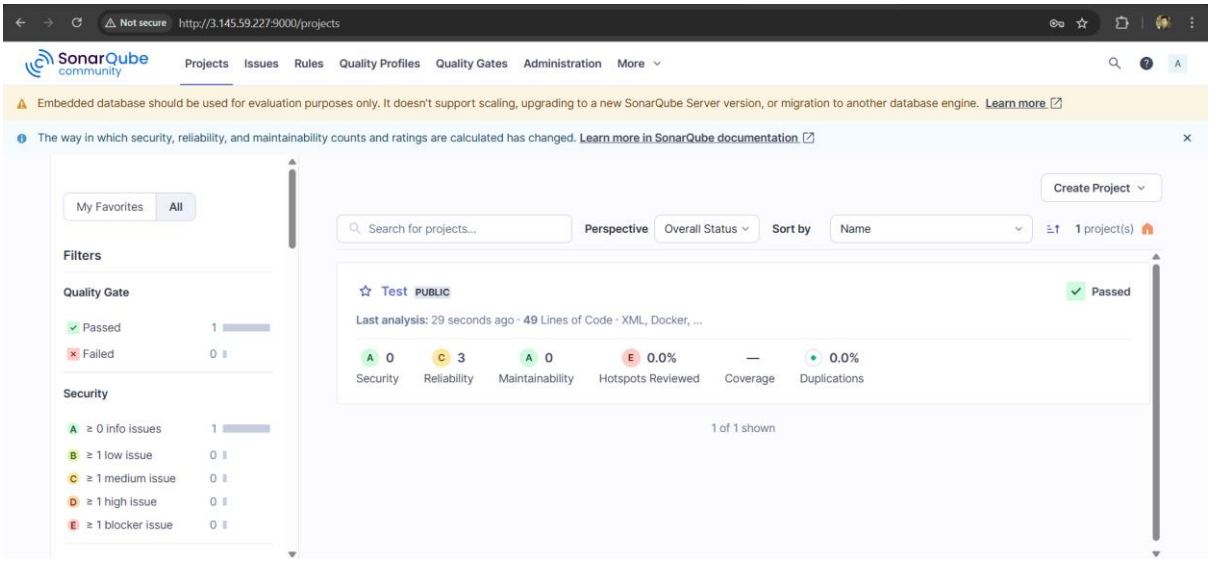
--install nexus



--install plugins

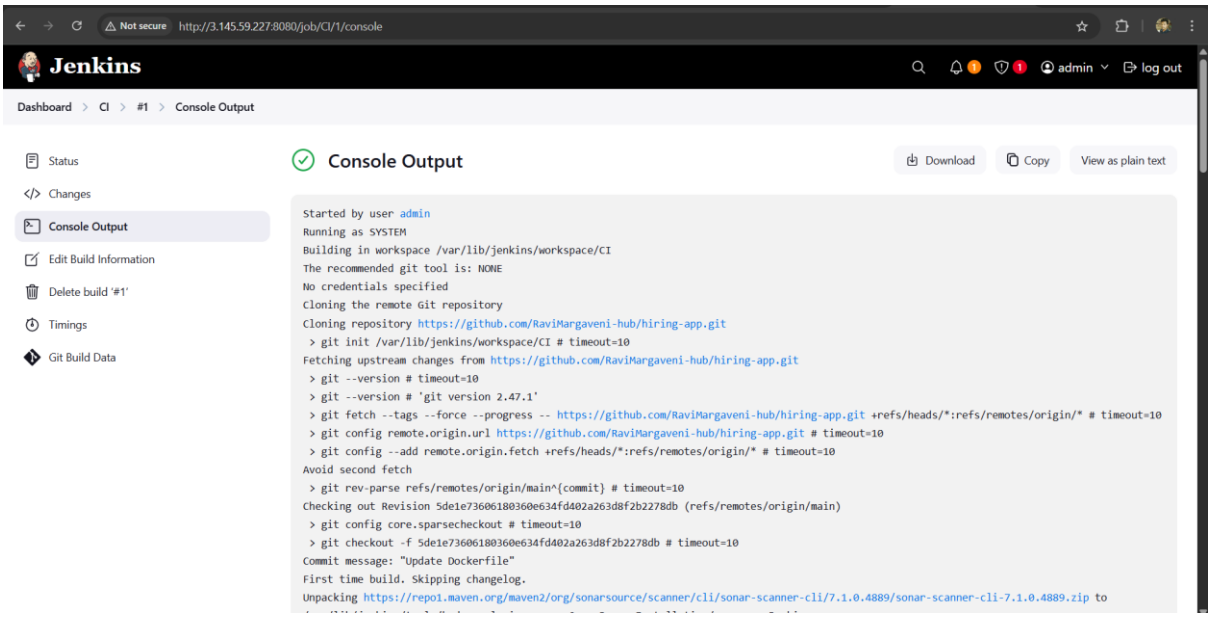


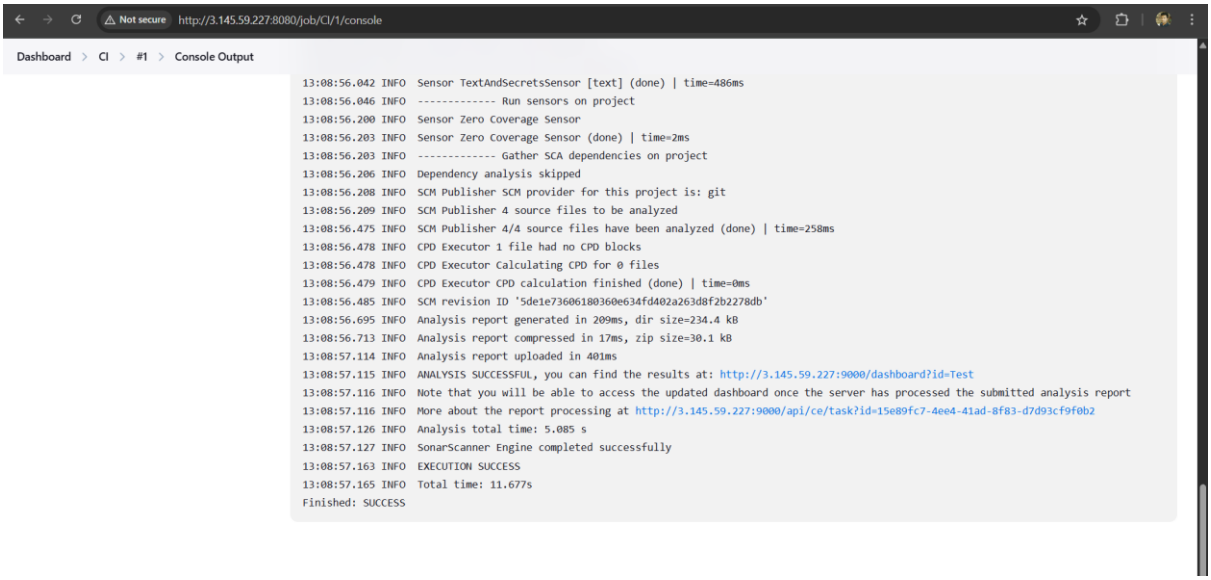
--check sonarqube



--finally created Jenkins job for integration

-integrated maven for build, sonarqube for quality check and nexus for artifact storage





3) Create Jenkins job to create docker image and build image.

---Created pipeline for modifying k8s Deployment manifest:

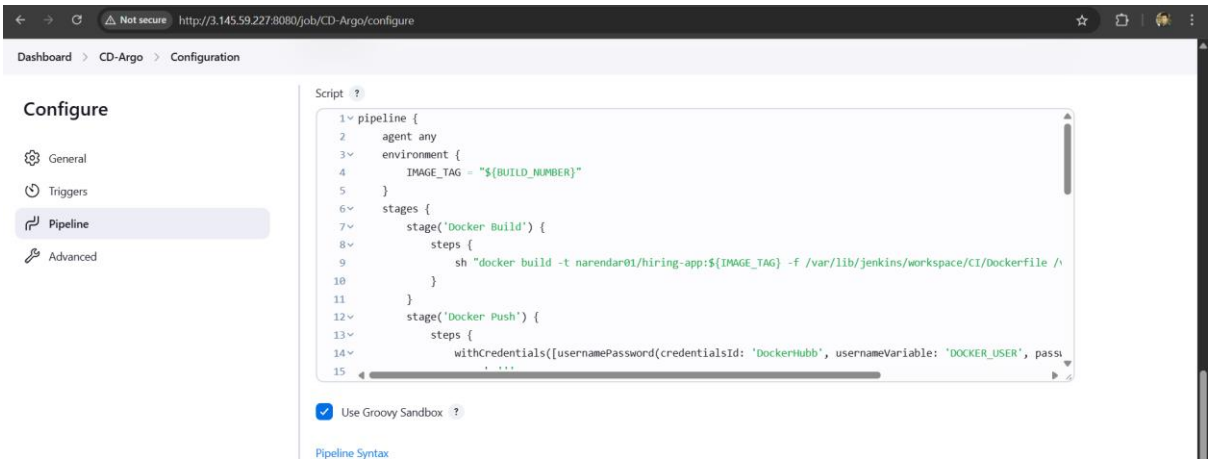
build docker image from Docker file.

push docker file into docker registry

Checkout to K8S manifest SCM

Update K8S manifest & push to Repo

---Pipeline script

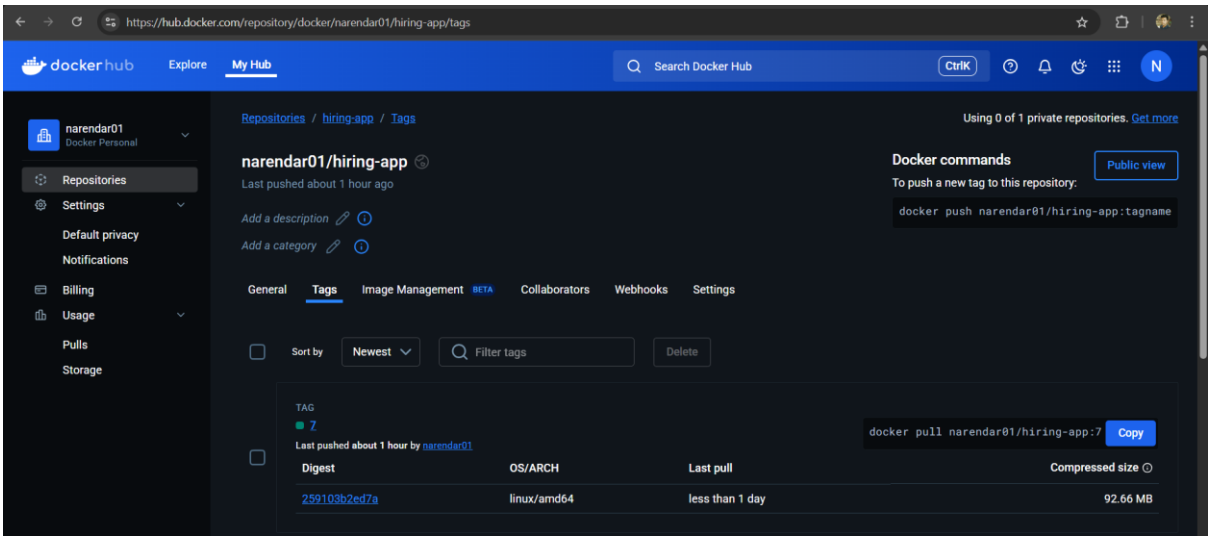



```
pipeline {
   agent any
   environment {
      IMAGE_TAG = "${BUILD_NUMBER}"
   }
   stages {
      stage('Docker Build') {
         steps {
            sh "docker build -t narendar01/hiring-app:${IMAGE_TAG} -f
/var/lib/jenkins/workspace/CI/Dockerfile /var/lib/jenkins/workspace/CI/"
```

```
        }
    }
    stage('Docker Push') {
        steps {
            withCredentials([usernamePassword(credentialsId: 'DockerHubb',
usernameVariable: 'DOCKER_USER', passwordVariable: 'DOCKER_PASS')]) {
                sh '''
                    echo "$DOCKER_PASS" | docker login -u "$DOCKER_USER" --password-
stdin
                    docker push narendar01/hiring-app:${IMAGE_TAG}
                '''
            }
        }
    }
    stage('Checkout K8S manifest SCM') {
        steps {
            git branch: 'main', url: 'https://github.com/narendar-20/Hiring-app-
argocd.git'
        }
    }
    stage('Update K8S manifest & push to Repo') {
        steps {
            script {
                withCredentials([usernamePassword(credentialsId: 'git-crredd',
passwordVariable: 'GIT_PASSWORD', usernameVariable: 'GIT_USERNAME')]) {
                    sh '''
                    cat /var/lib/jenkins/workspace/$JOB_NAME/dev/deployment.yaml
                    PREV_BUILD_NUMBER=$((BUILD_NUMBER - 1))
                    sed -i "s/${PREV_BUILD_NUMBER}/${BUILD_NUMBER}/g"
/var/lib/jenkins/workspace/$JOB_NAME/dev/deployment.yaml
                    cat /var/lib/jenkins/workspace/$JOB_NAME/dev/deployment.yaml
                    git add .
                    git commit -m 'Updated the deploy yaml | Jenkins Pipeline'
                    git remote -v
                    git push
https://$GIT_USERNAME:$GIT_PASSWORD@github.com/narendar-20/Hiring-app-
argocd.git main
                    '''
                }
            }
        }
```

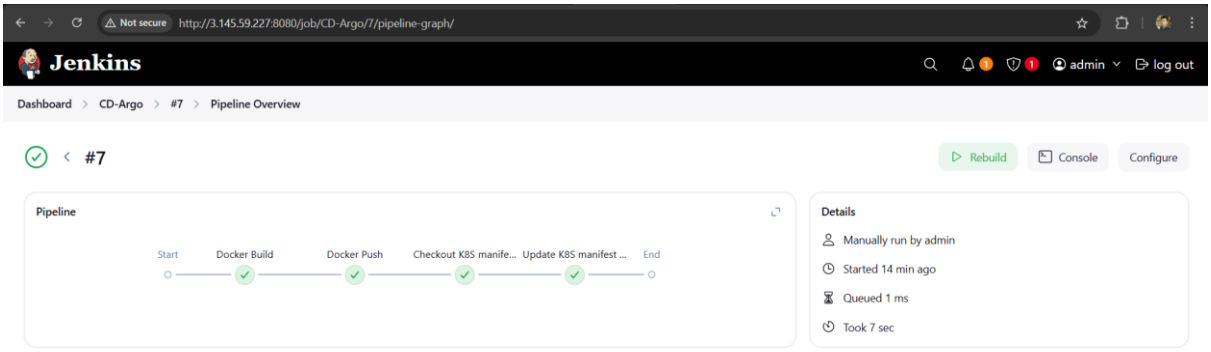
```
        }
    }
}
```

--pushed docker file to dockerhub



--this job has dependency on first job (CI-Job) Configured Build Trigger (Post-build Trigger)

--build job



4) Create ArgoCD job to deploy on k8s cluster.

--this job will modify the k8s manifest (deployment.yml)

--application.yaml

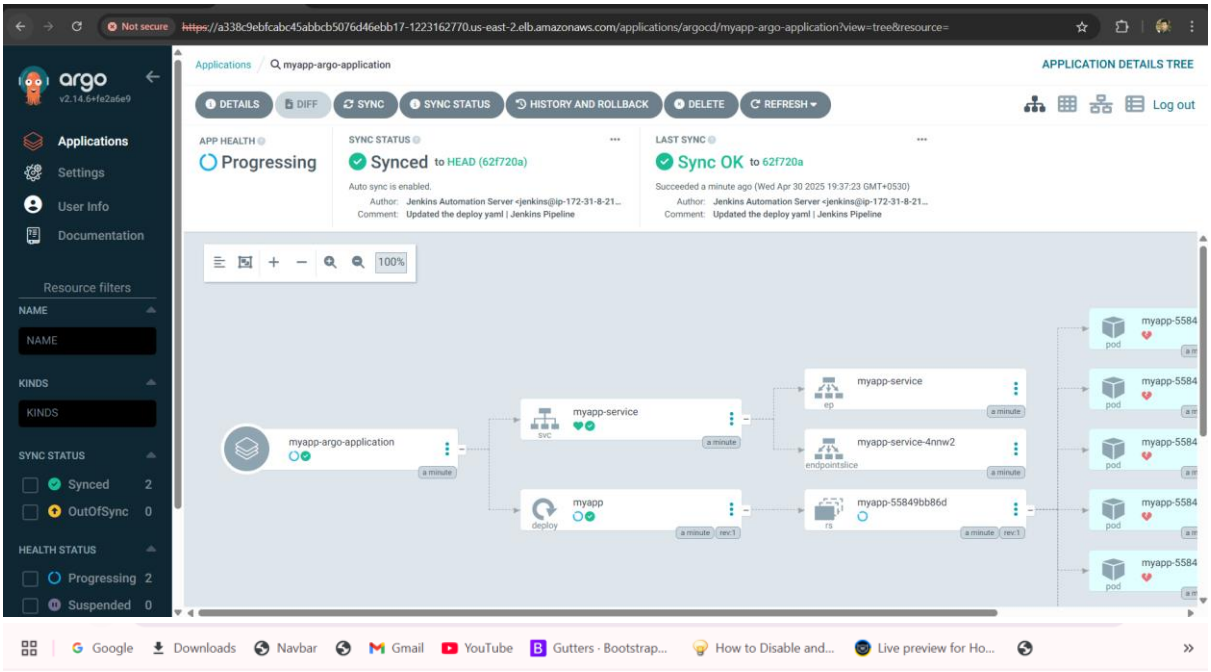
MINGW64:/c/Users/naren

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: myapp-argo-application
  namespace: argocd
spec:
  project: default

  source:
    repoURL: https://github.com/narendar-20/Hiring-app-argocd.git
    targetRevision: HEAD
    path: dev
  destination:
    server: https://kubernetes.default.svc
    namespace: myapp

  syncPolicy:
    syncOptions:
    - CreateNamespace=true

    automated:
      selfHeal: true
      prune: true
~
```

--Apply the application.yml file then it will create an application in argocd webapp

-when ever our 2nd job Updates K8S manifest & push to Repo argocd will make a sync with the newupdates

```
naren@narendar MINGW64 ~ (master)
$ vi application.yaml

naren@narendar MINGW64 ~ (master)
$ kubectl apply -f application.yaml
application.argoproj.io/myapp-argo-application created
```



Welcome to Techie Horizon