

Steps for implementing the Jenkins CI/CD project

1. Install Java jdk and Jenkins

```
sudo apt install openjdk-17-jre
```

```
curl -fsSL https://pkg.jenkins.io/debian/jenkins.io-2023.key | sudo tee \
```

```
/usr/share/keyrings/jenkins-keyring.asc > /dev/null
```

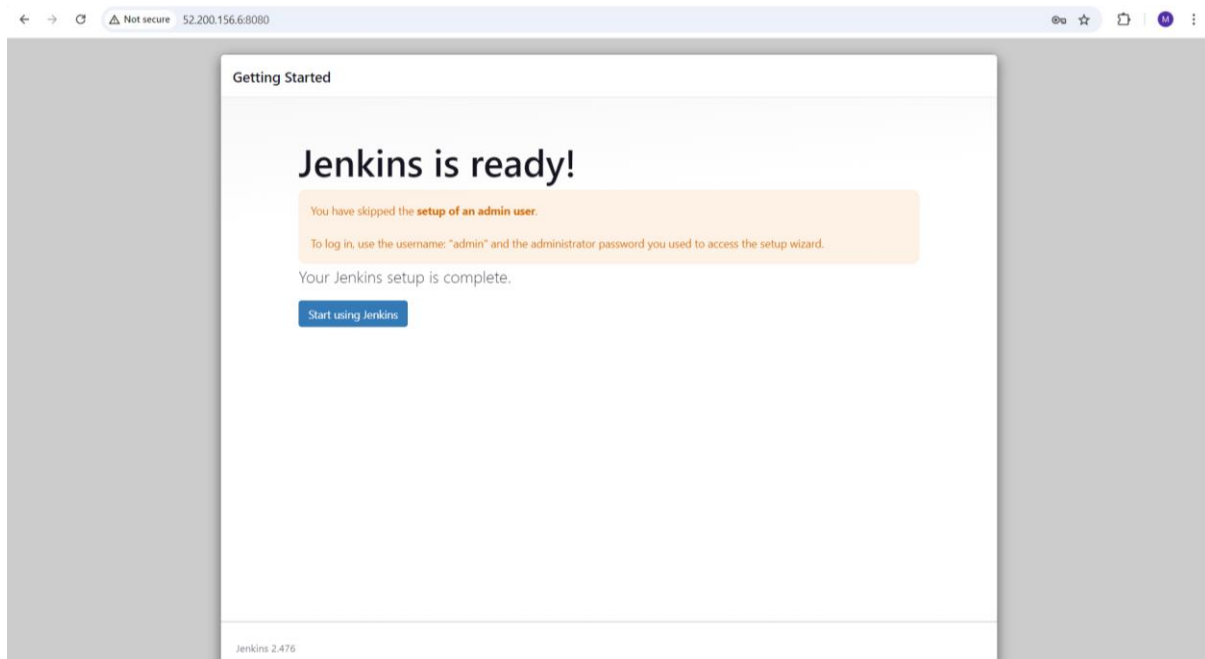
```
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
```

```
https://pkg.jenkins.io/debian binary/ | sudo tee \
```

```
/etc/apt/sources.list.d/jenkins.list > /dev/null
```

```
sudo apt-get update
```

```
sudo apt-get install Jenkins
```



1.create pipeline

2.install necessary plugins(sonarqube,docker)

3.Install Sonarqube

```
apt install unzip
```

```
adduser sonarqube
```

```
wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-9.4.0.54424.zip
```

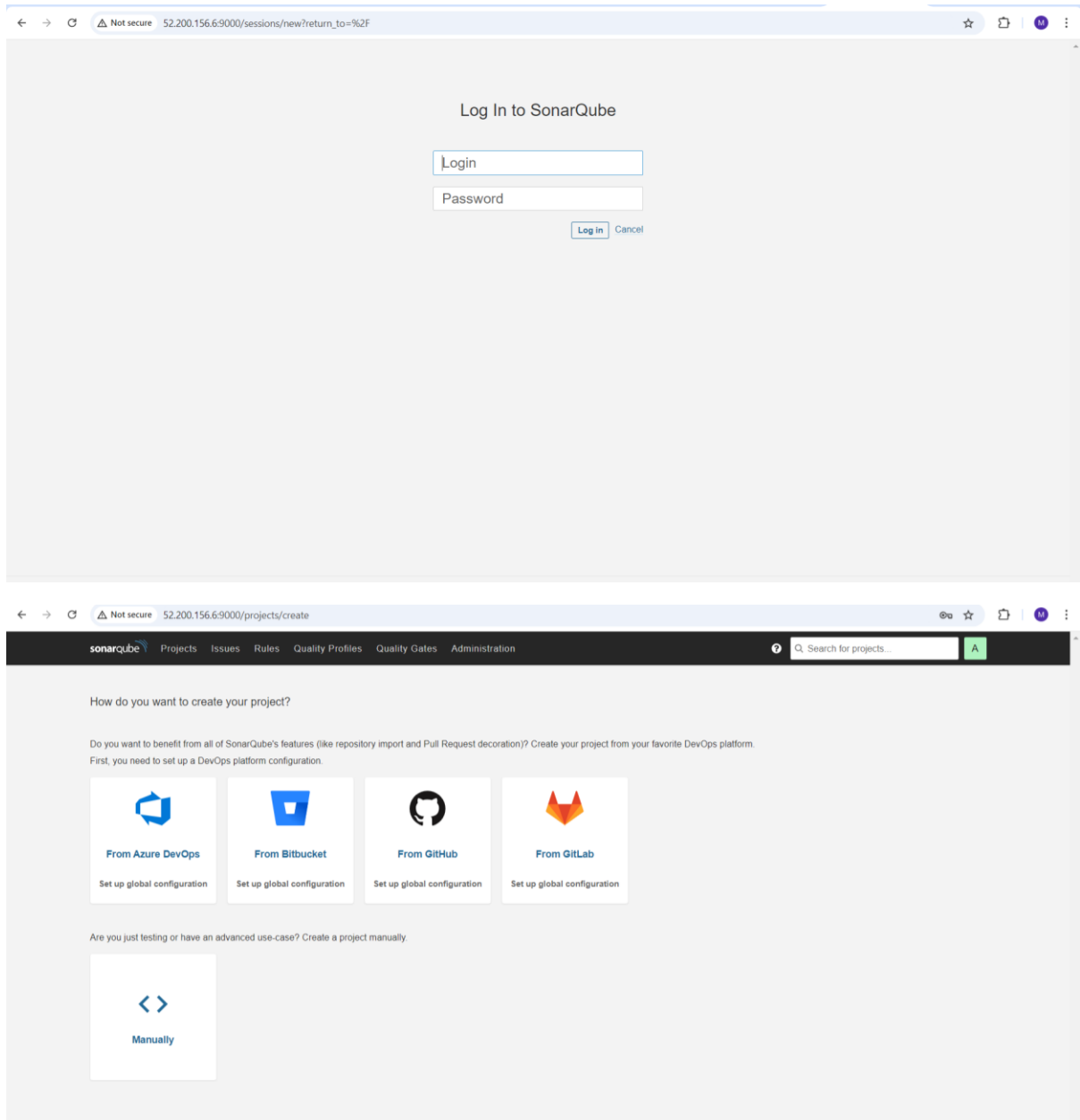
unzip *

chmod -R 755 /home/sonarqube/sonarqube-9.4.0.54424

chown -R sonarqube:sonarqube /home/sonarqube/sonarqube-9.4.0.54424

cd sonarqube-9.4.0.54424/bin/linux-x86-64/

./sonar.sh start



Generate token for sonarqube to access Jenkins

Install docker

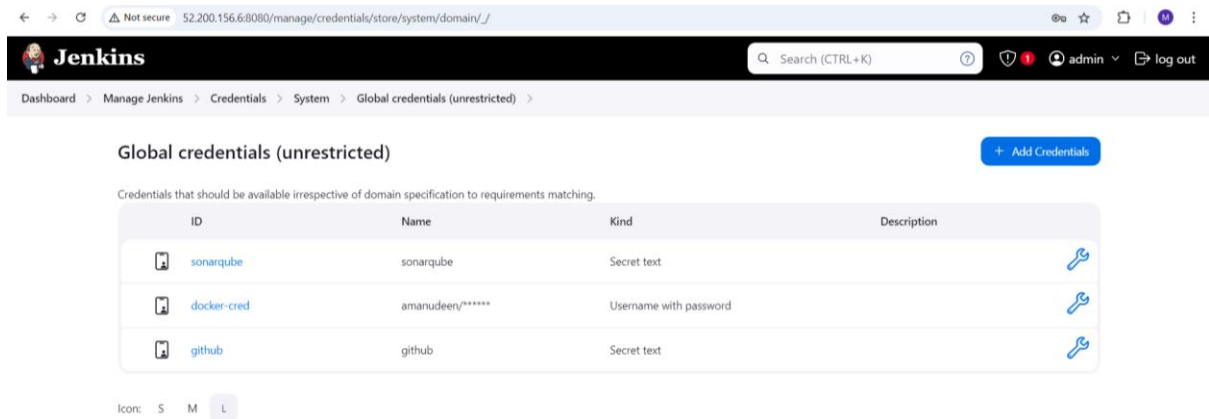
sudo apt install docker.io

sudo su -

usermod -aG docker jenkins

usermod -aG docker ubuntu

systemctl restart docker

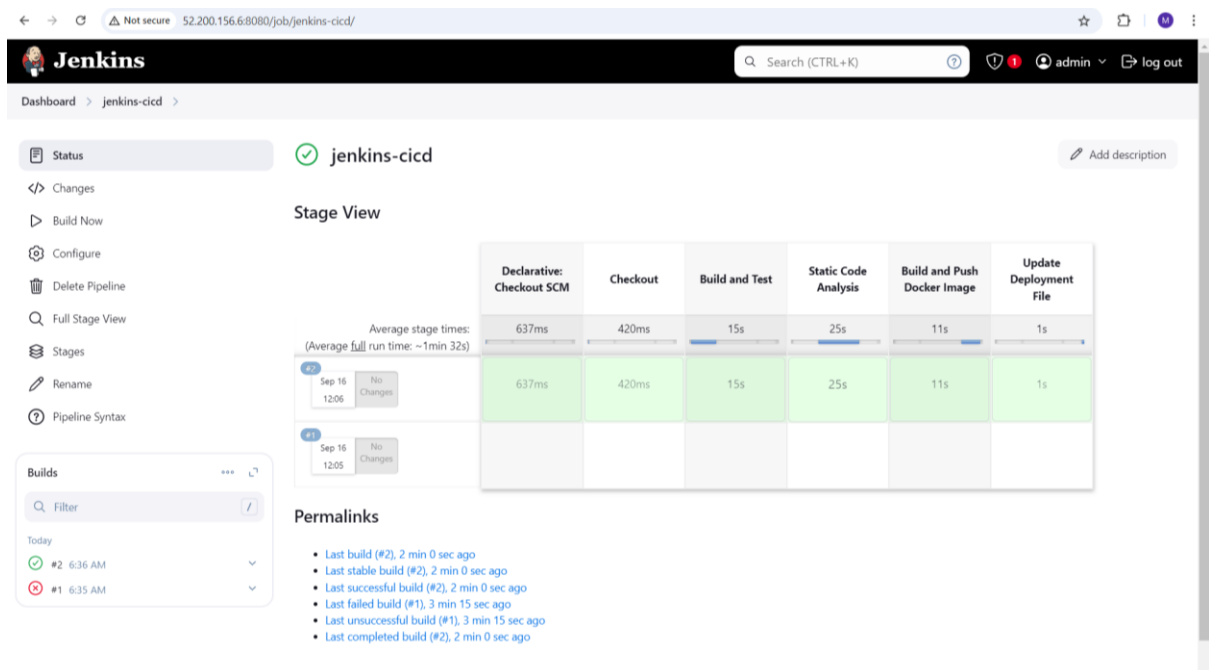


The screenshot shows the Jenkins web interface at the URL `52.200.156.6:8080/manage/credentials/store/system/domain/`. The page title is "Global credentials (unrestricted)". Below the title, it says "Credentials that should be available irrespective of domain specification to requirements matching." There is a table with the following columns: ID, Name, Kind, and Description. The table contains three entries: "sonarqube" (Secret text), "docker-cred" (Username with password), and "github" (Secret text). Each entry has a wrench icon for editing. At the bottom left, there are tabs for "Icons: S M L".

ID	Name	Kind	Description
sonarqube	sonarqube	Secret text	
docker-cred	amanudeen/*****	Username with password	
github	github	Secret text	

Add the credentials

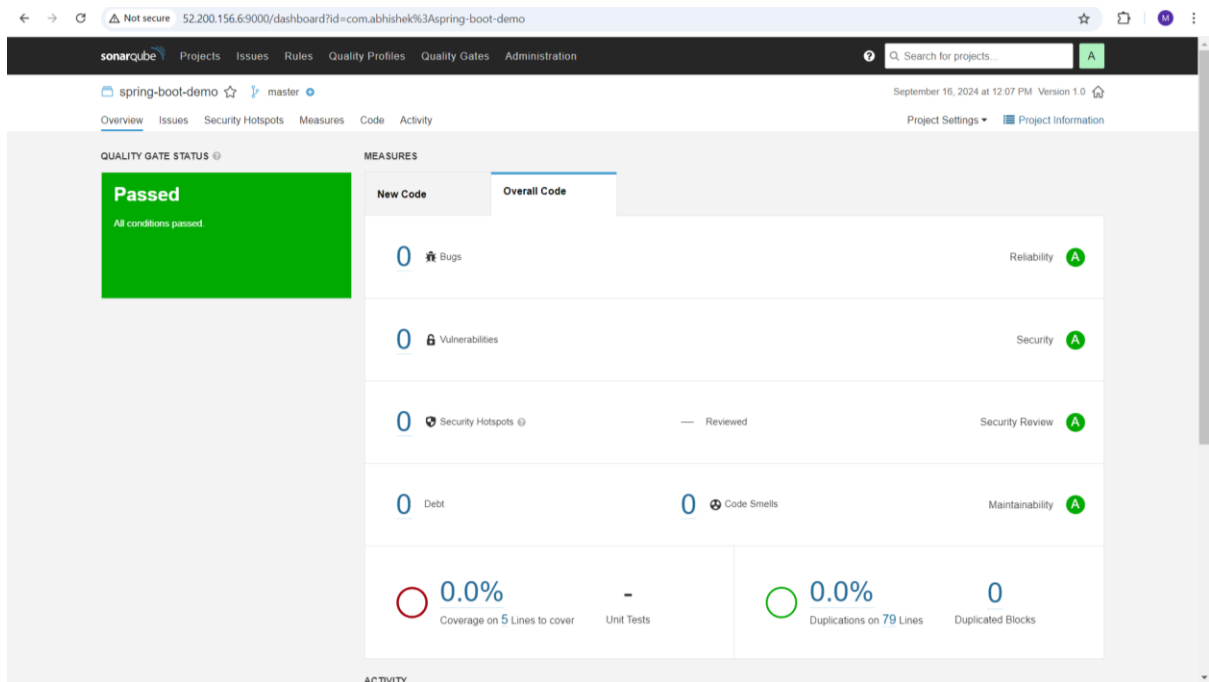
Build the pipeline



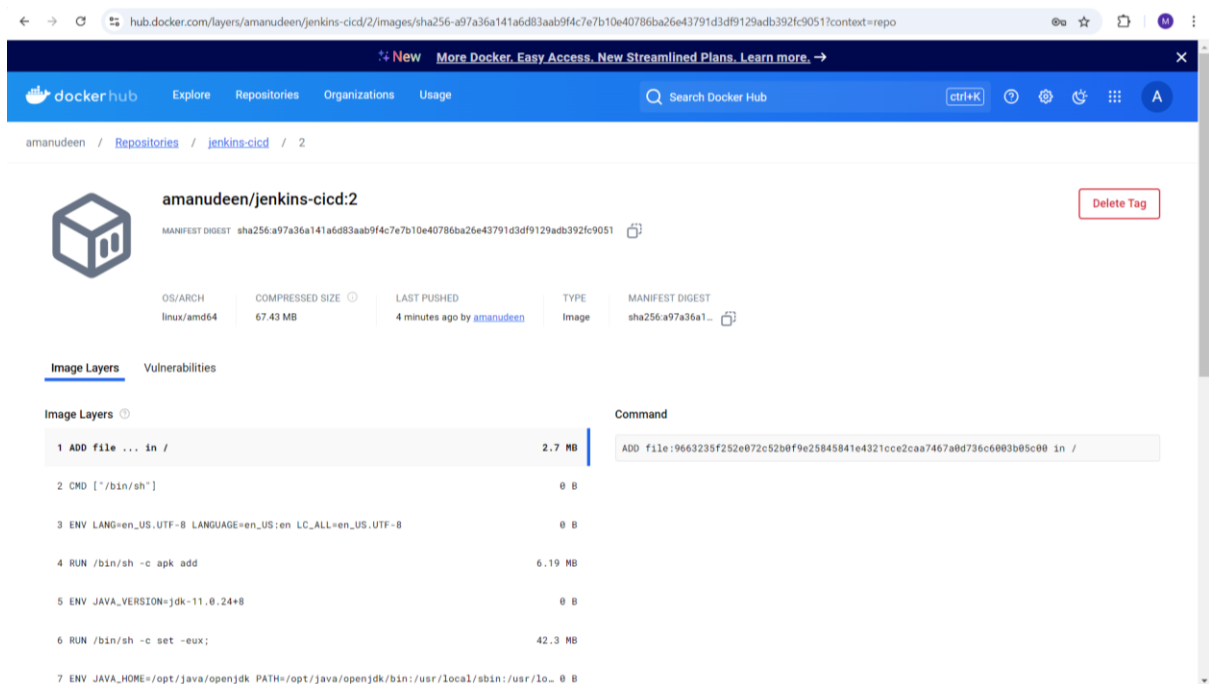
The screenshot shows the Jenkins web interface at the URL `52.200.156.6:8080/job/jenkins-cicd/`. The page title is "jenkins-cicd". The "Stage View" is displayed, showing a table of stages and their durations. The stages are: Declarative: Checkout SCM, Checkout, Build and Test, Static Code Analysis, Build and Push Docker Image, and Update Deployment File. The table shows the average stage times and the actual times for the last two builds. The "Builds" section on the left shows two builds: #2 (6:36 AM) and #1 (6:35 AM). The "Permalinks" section on the right lists various links for the builds.

Stage	Declarative: Checkout SCM	Checkout	Build and Test	Static Code Analysis	Build and Push Docker Image	Update Deployment File
Average stage times:	637ms	420ms	15s	25s	11s	1s
(Average full run time: ~1min 32s)						
Build #2	637ms	420ms	15s	25s	11s	1s
Build #1						

JenkinsCI Pipeline is successfully running.



Sonarqube code analysis is passed



New Dockerimage has been pushed with new tag

CI part is done

Install Minikube and Argocd

sudo apt update

curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

```
sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64
```

```
sudo chmod 777 /var/run/docker.sock
```

```
minikube start --memory=4098 --driver=hyperkit
```

```
sudo snap install kubectl --classic
```

```
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
```

```
sudo install minikube-linux-amd64 /usr/local/bin/minikube
```

```
sudo chmod 777 /var/run/docker.sock
```

```
minikube start --memory=4098 --driver=docker
```

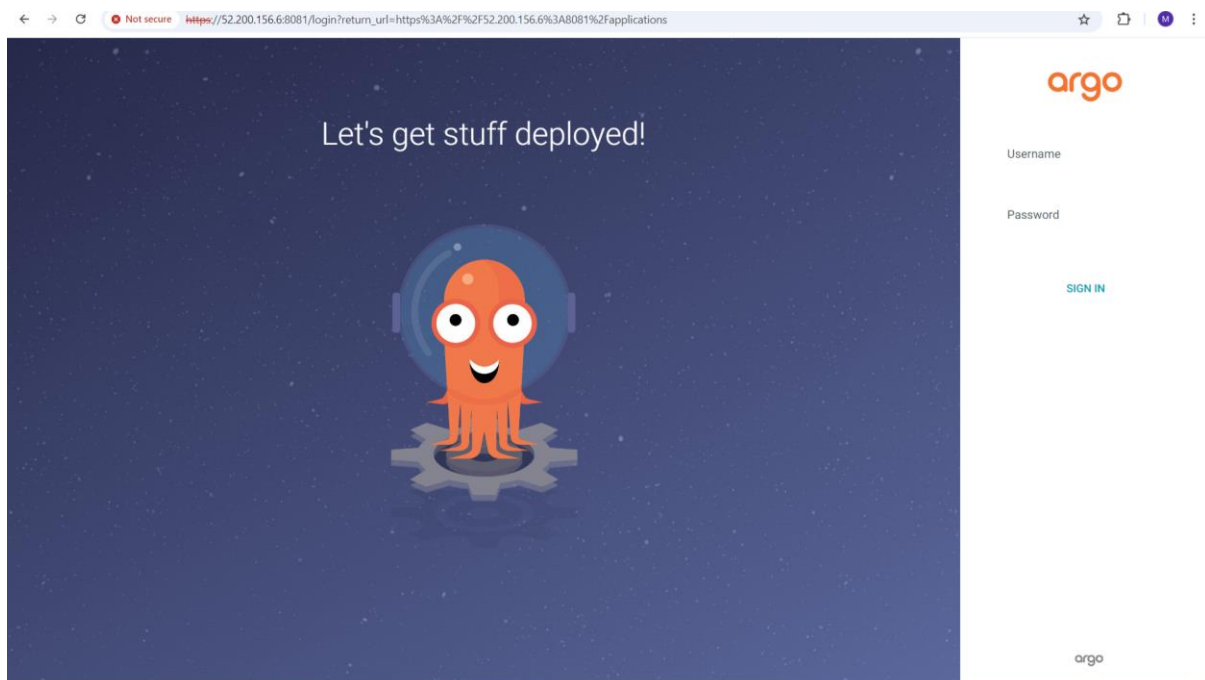
```
sudo snap install kubectl --classic
```

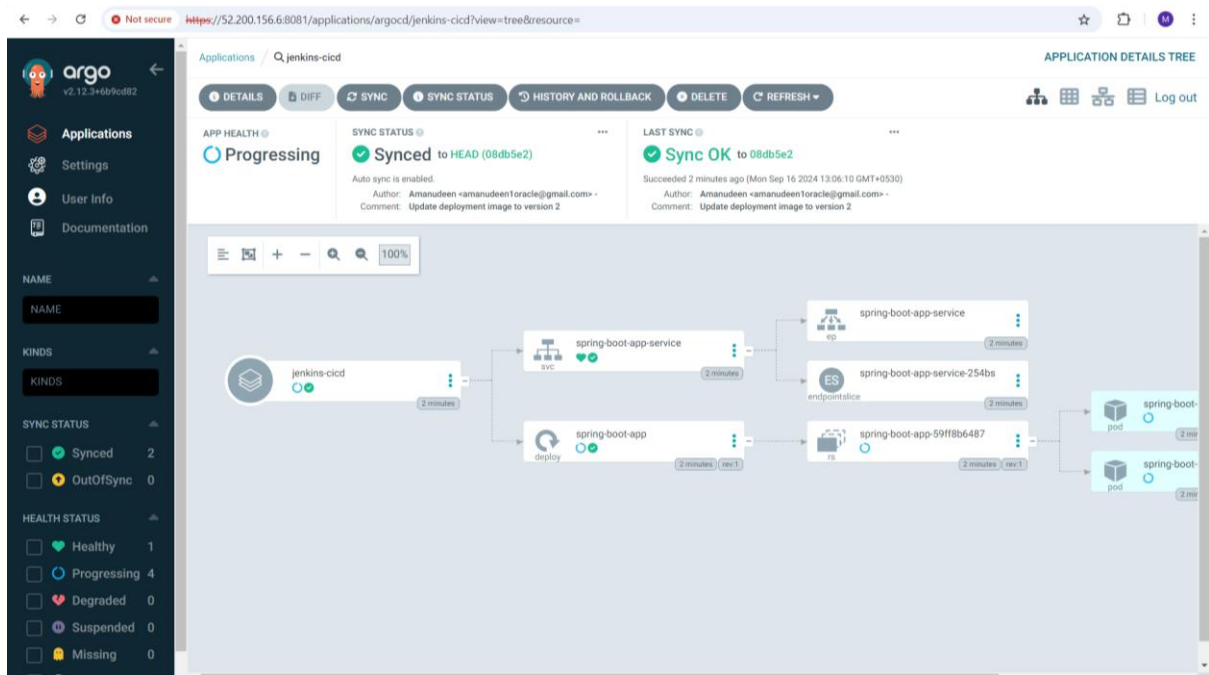
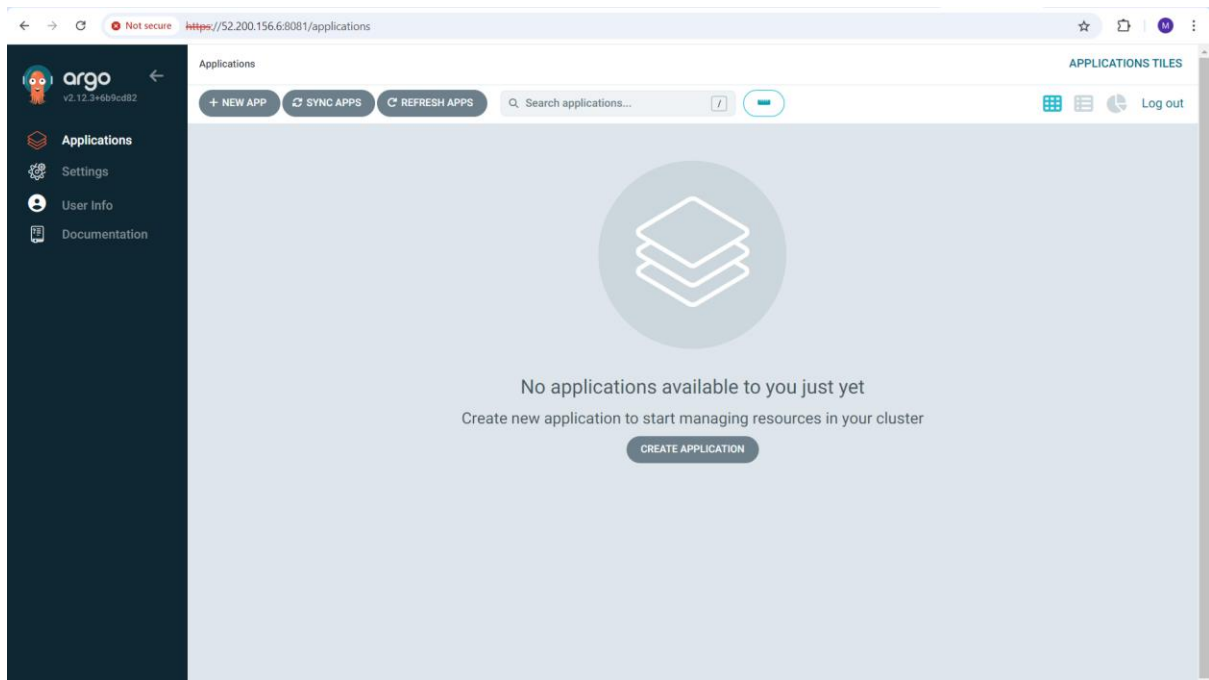
```
kubectl get pods
```

```
kubectl create namespace argocd
```

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

```
watch kubectl get pods -n argocd
```





Not secure https://52.200.156.6:8081/applications/argocd/jenkins-cid?view=tree&resource=&operation=true

argo v2.12.3+64b1cd82

Applications
Settings
User Info
Documentation

NAME
NAME

KINDS
KINDS

SYNC STATUS
Synced 2
OutOfSync 0

HEALTH STATUS
Healthy 1
Progressing 4
Degraded 0
Suspended 0
Missing 0
Unknown 0

OPERATION Sync

PHASE Succeeded

MESSAGE successfully synced (all tasks run)

STARTED AT 4 minutes ago (Mon Sep 16 2024 13:06:10 GMT+0530)

DURATION 00:00 min

FINISHED AT 4 minutes ago (Mon Sep 16 2024 13:06:10 GMT+0530)

REVISION 08db5e2

INITIATED BY automated sync policy

RESULT

KIND	NAMESPACE	NAME	STATUS	HOOK	MESSAGE
v1/Service	argocd	spring-boot-app-service	Synced		service/spring-boot-app-service created
apps/v1/...	argocd	spring-boot-app	Synced		deployment.apps/spring-boot-app created

Not secure https://52.200.156.6:8081/applications/argocd/jenkins-cid?view=tree&operation=false&resource=

Applications / jenkins-cid

APPLICATION DETAILS TREE

DETAILS DIFF SYNC SYNC STATUS HISTORY AND ROLLBACK DELETE REFRESH

APP HEALTH @ Healthy

SYNC STATUS @ Synced to HEAD (d64daab)

LAST SYNC @ Sync OK to d64daab

Auto sync is enabled.
Author: Amanudeen <amanudeen1oracle@gmail.com>
Comment: Update deployment image to version 2

Succeeded 3 minutes ago (Mon Sep 16 2024 13:49:12 GMT+0530)
Author: Amanudeen <122690641+Amanudeen@users.norepl...>
Comment: Update deployment.yml

jenkins-cid

spring-boot-app-service

spring-boot-app-service-254bs

spring-boot-app-684ff5d6f5

spring-boot-app-59ff8b6487

spring-boot-app-684ff5d6f5-4...

spring-boot-app-684ff5d6f5-p...

Pods are healthy

ArgoCD is successfully running.