

1. Setup ARGO CD

Login into your k8s cluster and install ArgoCD

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

```
[root@master ~]# kubectl create namespace argocd
namespace/argocd created
[root@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
role.rbac.authorization.k8s.io/argocd-server created
clusterrole.rbac.authorization.k8s.io/argocd-application-controller created
clusterrole.rbac.authorization.k8s.io/argocd-applicationset-controller created
```

- **kubectl get pods -n argocd**

```
[root@master ~]# kubectl get pods -n argocd
```

NAME	READY	STATUS	RESTARTS	AGE
argocd-application-controller-0	1/1	Running	0	3m12s
argocd-applicationset-controller-6dd89769b9-cq8sc	1/1	Running	0	3m12s
argocd-dex-server-5d887d57b9-jq9tk	1/1	Running	0	3m12s
argocd-notifications-controller-f9847886b-pm7c4	1/1	Running	0	3m12s
argocd-redis-7c76ccfd48-dxzmd	1/1	Running	0	3m12s
argocd-repo-server-5bf876bdfb-jrkkp	1/1	Running	0	3m12s
argocd-server-69d5b9f8d5-6rm2d	1/1	Running	0	3m12s

```
[root@master ~]#
```

- **kubectl port-forward svc/argocd-server -n argocd**
8080:443
- **kubectl patch svc argocd-server -n argocd **
-p '{"spec": {"type": "NodePort"}}'

```
[root@master ~]# kubectl port-forward svc/argocd-server -n argocd 8080:443
Forwarding from 127.0.0.1:8080 -> 8080
Forwarding from [::1]:8080 -> 8080
kubectl get svc -n argocd
^C[root@master ~]# kubectl get svc -n argocd
NAME                                TYPE            CLUSTER-IP      EXTERNAL-IP
argocd-applicationset-controller    ClusterIP       10.106.172.215   <none>
argocd-dex-server                   ClusterIP       10.103.109.224   <none>
argocd-metrics                      ClusterIP       10.101.158.47    <none>
argocd-notifications-controller-metrics ClusterIP       10.102.124.48    <none>
argocd-redis                        ClusterIP       10.101.83.232    <none>
argocd-repo-server                  ClusterIP       10.107.147.159   <none>
argocd-server                       ClusterIP       10.97.232.66     <none>
argocd-server-metrics               ClusterIP       10.107.153.5     <none>
[root@master ~]# kubectl patch svc argocd-server -n argocd \
-p '{"spec": {"type": "NodePort"}}'
service/argocd-server patched
```

- **kubectl get svc argocd-server -n argocd**

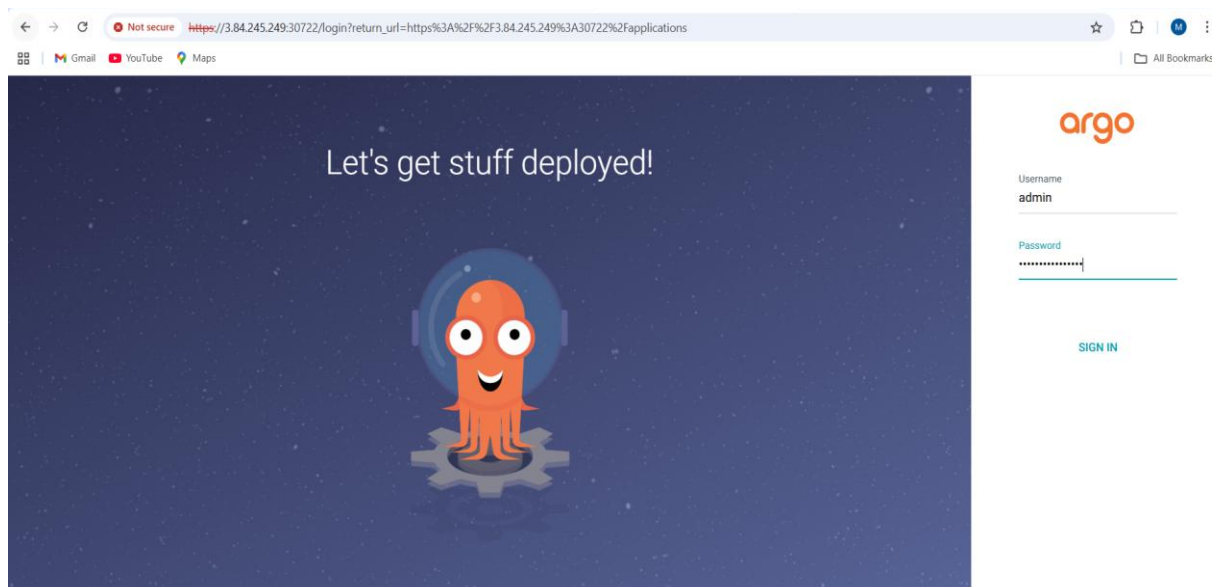
```
[root@master ~]# kubectl get svc argocd-server -n argocd
NAME            TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)                                     AGE
argocd-server   NodePort    10.97.232.66   <none>          80:30722/TCP,443:32003/TCP               11m
```

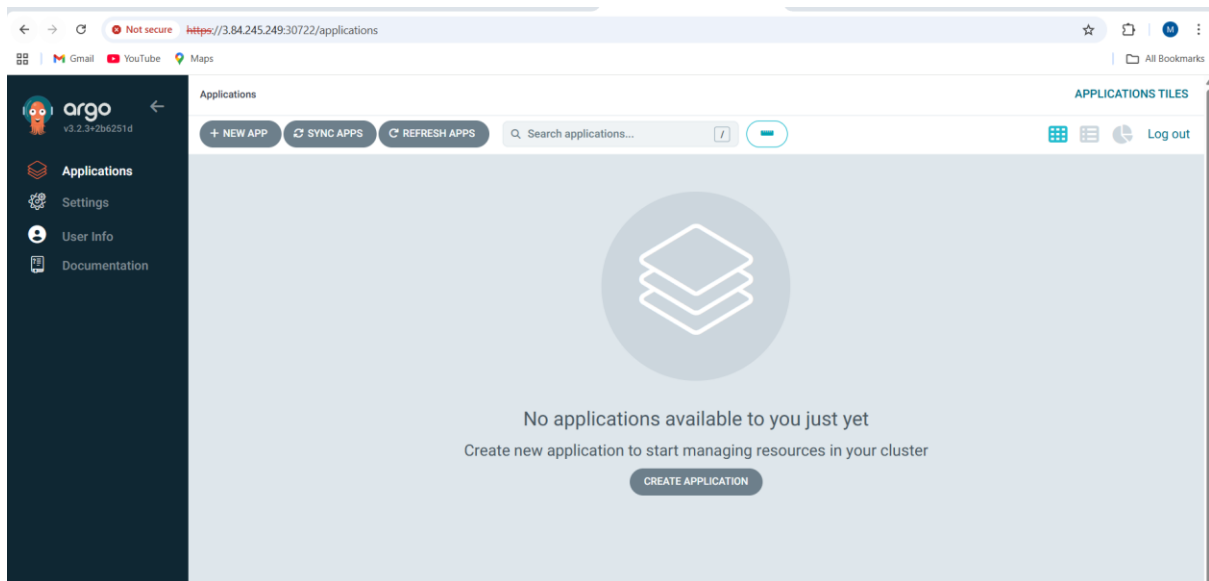
Access in browser with the ip and port number:30722

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

```
[root@master ~]# kubectl get secret argocd-initial-admin-secret -n argocd \
-o jsonpath="{.data.password}" | base64 --decode
9VgucHD3U7-Xk4Az[root@master ~]#
```

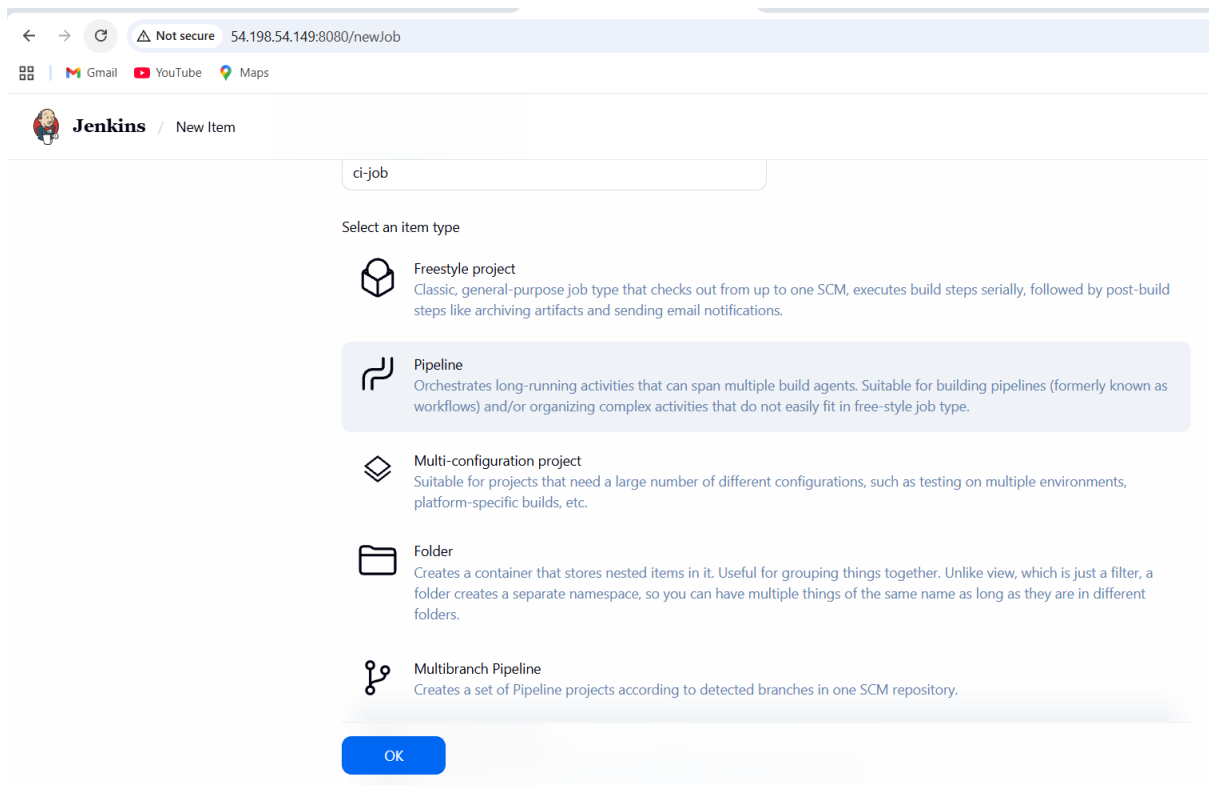
Username:admin password given in the above command we can access the argocd





2. Create Jenkins job for CI.

Login into Jenkins server and build a new job with the name ci-job and select type as pipeline.



Create repository in github and add Jenkins file.

New repository

Type to search

Create a new repository

Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository.](#)
Required fields are marked with an asterisk (*).

1

General

Owner *

mujaheed00

Repository name *

ci-job

ci-job is available.

Great repository names are short and memorable. How about [supreme-train](#)?

Description

0 / 350 characters

2

Configuration

Choose visibility *

Public

Choose who can see and commit to this repository

Add README

READMEs can be used as longer descriptions. [About READMEs](#)

On ☒

Add .gitignore

No .gitignore

github.com/mujaheed00/ci-job/new/main

Gmail YouTube Maps

All Bookmarks

mujaheed00 / ci-job

Type to search

+ ↻ 🔍 📄 📧 📧

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

ci-job / Jenkinsfile

in main

Cancel changes

Commit changes...

Edit Preview


Spaces 2 No wrap

```
1 pipeline {
2   agent any
3
4   options {
5     timestamps()
6     disableConcurrentBuilds()
7   }
8
9   stages {
10
11     stage('Checkout') {
12       steps {
13         checkout scm
14       }
15     }
16
17     stage('Build') {
18       steps {
19         sh 'mvn clean compile'
20       }
21     }
22   }
23 }
```

Select git scm and give your repository

← → ↻ ⚠ Not secure 54.198.54.149:8080/job/ci-job/configure

📦 Gmail YouTube Maps

 **Jenkins** / ci-job ▾ / Configuration

Configure

- ⚙ General
- 🕒 Triggers
- 📄 Pipeline**
- 🔧 Advanced

Define your Pipeline using [Groovy](#) directly or pull it from source control.

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

`https://github.com/mujaheed00/ci-job.git`

Credentials ?


- none -

Advanced ▾

Save **Apply**

← → ↻ ⚠ Not secure 54.198.54.149:8080/job/ci-job/configure

📦 Gmail YouTube Maps

 **Jenkins** / ci-job ▾ / Configuration

Configure

- ⚙ General
- 🕒 Triggers
- 📄 Pipeline**
- 🔧 Advanced

Repository URL ?

`https://github.com/mujaheed00/ci-job.git`

Credentials ?

- none -

Advanced ▾

+ Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

`*/main`

+ Add Branch

Click on build now

The screenshot shows the Jenkins web interface for a job named 'ci-job'. The left sidebar contains navigation links: Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Stages, Rename, and Pipeline Syntax. The main area displays the job's status as 'Success' with a green checkmark. Below this, there's a section for 'Last Successful Artifacts' showing 'ci-artifact.tar.gz' (233 B). The 'Stage View' section shows a table of stages and their durations for the current build (#4).

Stage	Duration
Declarative: Checkout SCM	161ms
Checkout	146ms
Build	324ms
Test	330ms
Package	332ms
Archive Artifact	72ms
Declarative: Post Actions	80ms

The 'Permalinks' section lists recent builds: Last build (#4), 17 sec ago; Last stable build (#4), 17 sec ago; Last successful build (#4), 17 sec ago; and Last failed build (#3), 2 min 17 sec ago.

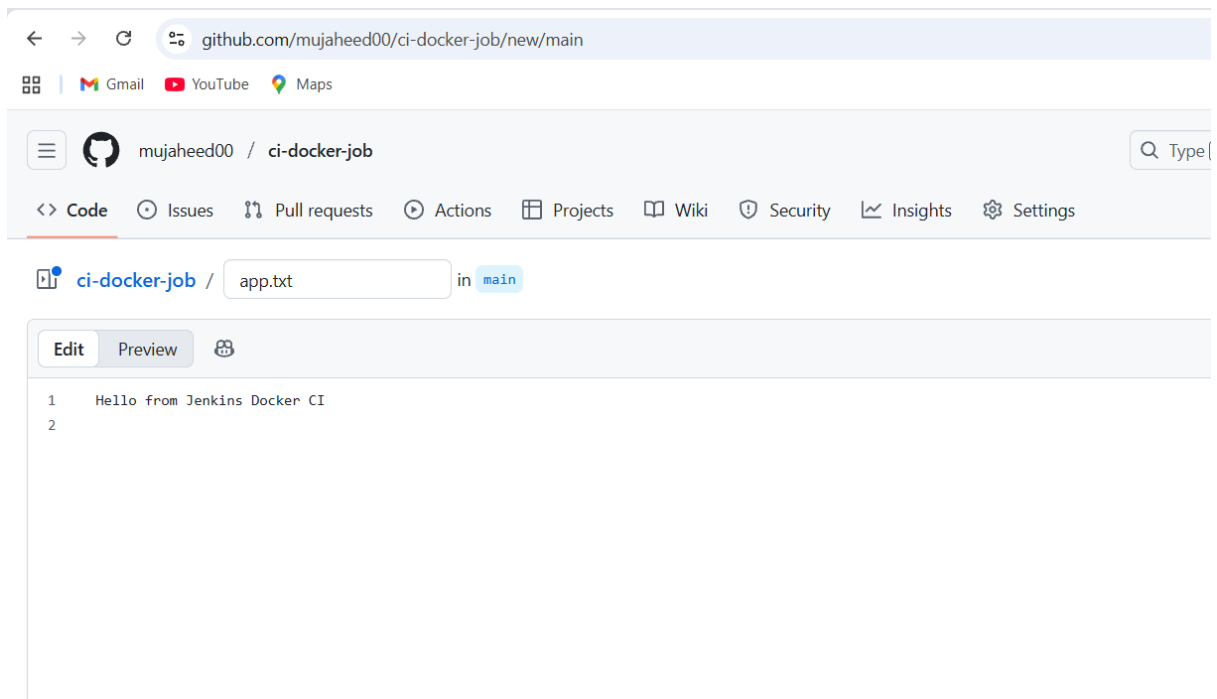
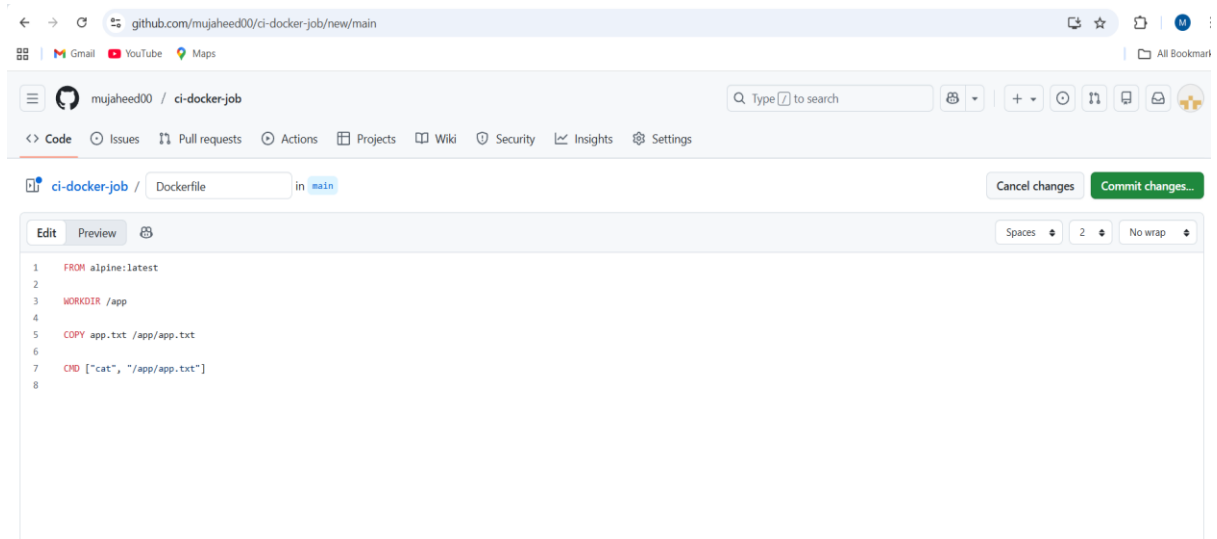
3. Create Jenkins job to create docker image and build image.

Make sure that your Jenkins server will have docker service up and running.

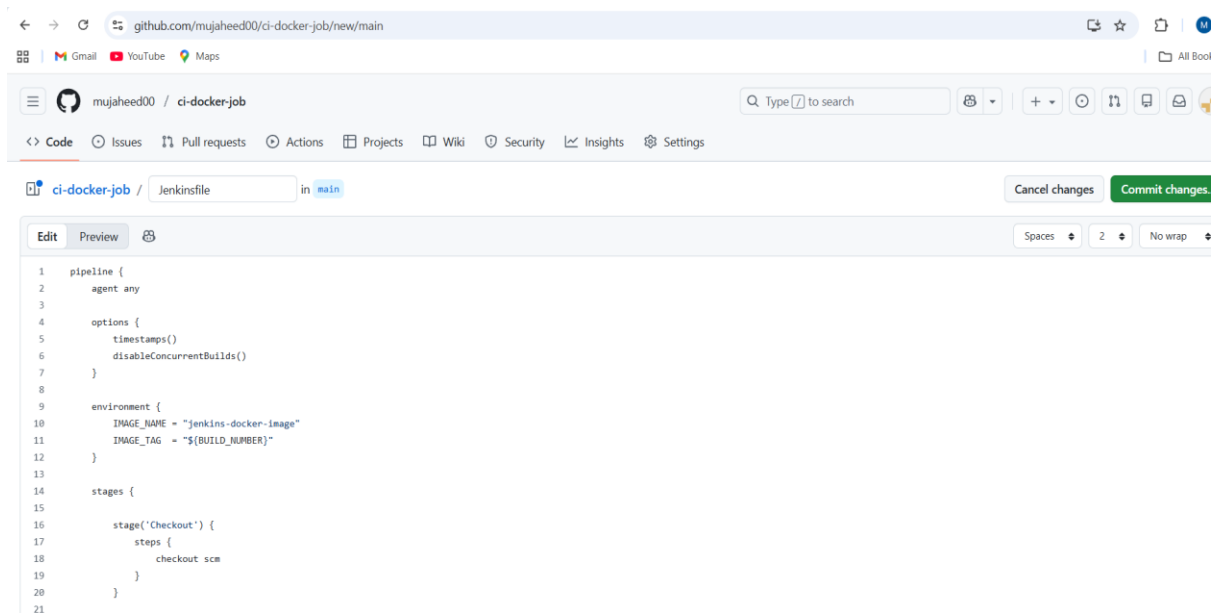
Create a new repository in github with the name ci-docker-job

The screenshot shows the GitHub 'Create a new repository' form. The 'General' section is active, showing the repository name 'ci-docker-job' and the owner 'mujaheed00'. A message indicates 'ci-docker-job is available'. The 'Description' field is empty. The 'Configuration' section shows the visibility set to 'Public', the 'Add README' checkbox is checked, and the 'Add .gitignore' dropdown is set to 'No .gitignore'.

Create a Dockerfile in that repository.



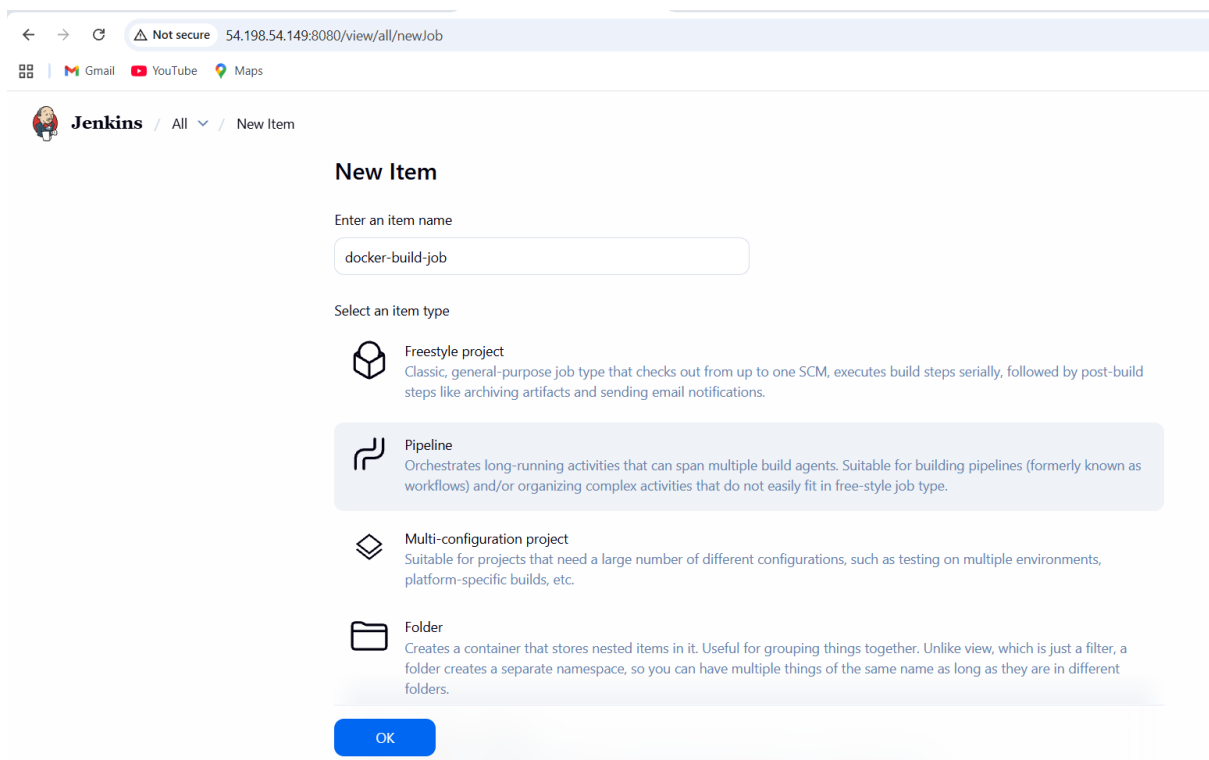
Add another file Jenkinsfile



The screenshot shows a GitHub web interface for a repository named 'ci-docker-job' by user 'mujaheed00'. The file 'Jenkinsfile' is selected in the 'main' branch. The Jenkinsfile content is as follows:

```
1 pipeline {
2   agent any
3
4   options {
5     timestamps()
6     disableConcurrentBuilds()
7   }
8
9   environment {
10    IMAGE_NAME = "jenkins-docker-image"
11    IMAGE_TAG = "${BUILD_NUMBER}"
12  }
13
14  stages {
15    stage('Checkout') {
16      steps {
17        checkout scm
18      }
19    }
20  }
21 }
```

Build a new job with the name docker-build-job and type as pipeline



The screenshot shows the Jenkins 'New Item' configuration page. The browser address bar indicates a local address: 54.198.54.149:8080/view/all/newJob. The page title is 'Jenkins / All / New Item'. The 'Enter an item name' field contains 'docker-build-job'. Under 'Select an item type', four options are listed:

- Freestyle project**: Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type. (This option is highlighted with a blue background.)
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

An 'OK' button is located at the bottom of the configuration area.

Give the repo which you have jenkinsfile and dockerfile and app.txt give that in git scm

The screenshot shows the Jenkins Configuration page for a job named 'docker-build-job'. The left sidebar has a 'Configure' section with options: General, Triggers, Pipeline (selected), and Advanced. The main area is titled 'Configuration' and contains the following fields:

- Repositories**:
 - Repository URL**: `https://github.com/mujaheed00/ci-docker-job.git`
 - Credentials**: `- none -`
 - Advanced**: `Advanced` (dropdown)
 - + Add Repository** button
- Branches to build**:
 - Branch Specifier (blank for 'any')**: `*/main`

At the bottom are **Save** and **Apply** buttons.

Click on build now.

The screenshot shows the Jenkins 'docker-build-job' Status page. The left sidebar has a 'Status' section with options: Status (selected), Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Stages, Rename, and Pipeline Syntax. The main area is titled 'docker-build-job' and contains the following elements:

- Stage View**: A table showing stage times for a build.
- Permalinks**: A section for build links.

Stage View Table:

	Declarative: Checkout SCM	Checkout	Build Docker Image	Verify Image	Declarative: Post Actions
Average stage times: (full run time: ~5s)	271ms	170ms	1s	321ms	117ms
#1 Jan 09 13:35 No Changes	271ms	170ms	1s	321ms	117ms

Permalinks

Builds: #1 8:05 AM

4. Create ArgoCD job to deploy on ek8s cluster.

Go to github and click on new repository and name it as argocd-eks-deploy

github.com/new

New repository

Search Type to search

Create a new repository

Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository](#).
Required fields are marked with an asterisk (*).

1 General

Owner * mujaheed00 / Repository name * argocd-eks-deploy
✔ argocd-eks-deploy is available.

Great repository names are short and memorable. How about [laughing-octo-carnival](#)?

Description
0 / 350 characters

2 Configuration

Choose visibility * Public
Choose who can see and commit to this repository

Add README On
READMEs can be used as longer descriptions. [About READMEs](#)

Add .gitignore No .gitignore
.gitignore tells git which files not to track. [About ignoring files](#)

In that repository add manifest files.

Namespace.yml:

github.com/mujaheed00/argocd-eks-deploy/new/main

mujaheed00 / argocd-eks-deploy

<> Code Issues Pull requests Actions Projects Wiki Security Insights

argocd-eks-deploy / namespace.yaml in main

Edit Preview

```
1 apiVersion: v1
2 kind: Namespace
3 metadata:
4   name: demo-app
5 |
```

deployment.yaml

← → ↻ github.com/mujaheed00/argocd-eks-deploy/new/main

📁 | 📧 Gmail 📺 YouTube 📍 Maps

☰ mujaheed00 / argocd-eks-deploy 🔍 Type /

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

📁 argocd-eks-deploy / deployment.yaml in main

Edit Preview 🗑

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: demo-app
5    namespace: demo-app
6  spec:
7    replicas: 2
8    selector:
9      matchLabels:
10       app: demo
11    template:
12      metadata:
13        labels:
14          app: demo
15      spec:
16        containers:
17        - name: demo-container
18          image: nginx:latest
19          ports:
20            - containerPort: 80
21
```

Service.yaml

← → ↻ github.com/mujaheed00/argocd-eks-deploy/new/main

📁 | 📧 Gmail 📺 YouTube 📍 Maps

☰ mujaheed00 / argocd-eks-deploy 🔍 Type / to

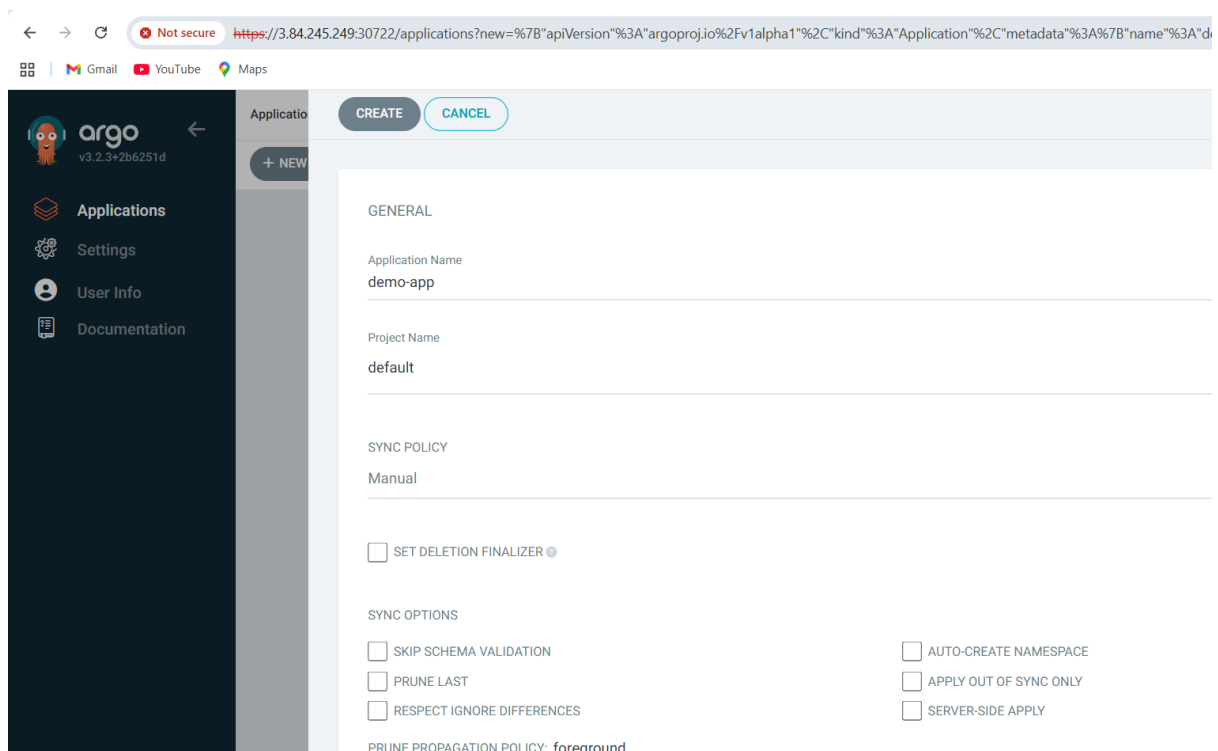
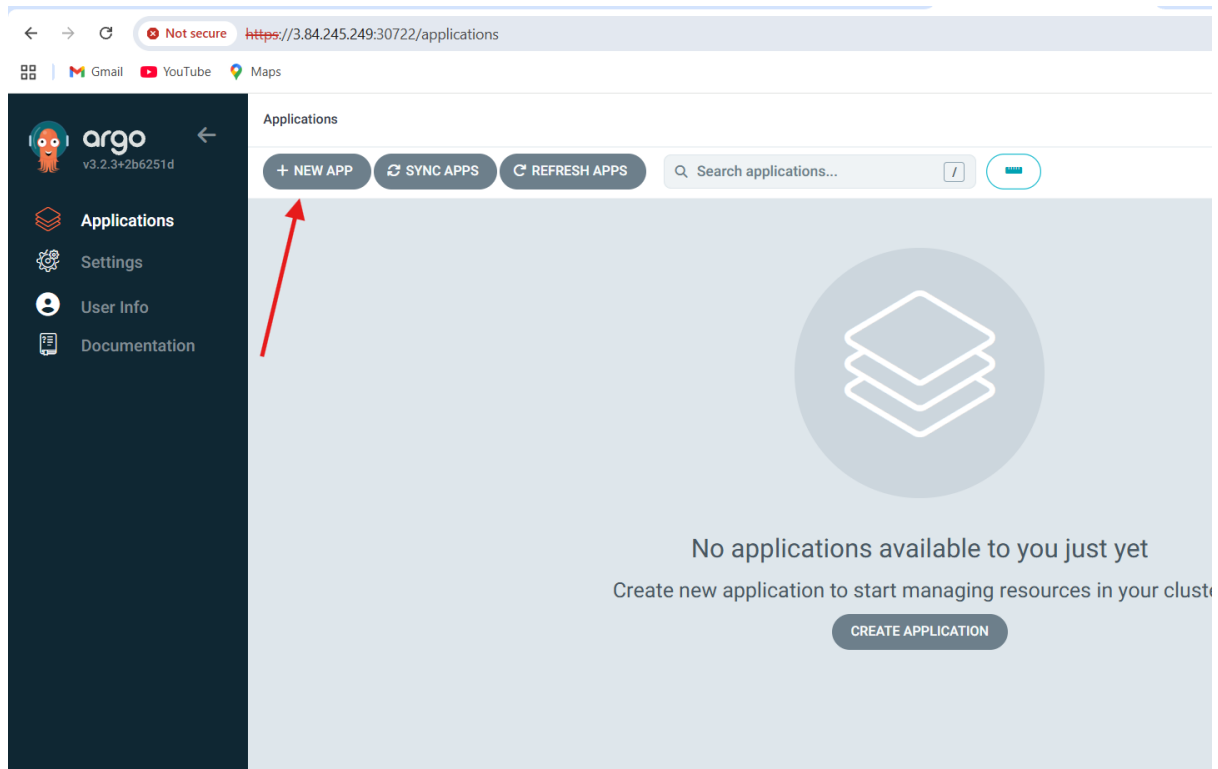
<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

📁 argocd-eks-deploy / service.yaml in main

Edit Preview 🗑

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: demo-service
5    namespace: demo-app
6  spec:
7    type: LoadBalancer
8    selector:
9      app: demo
10   ports:
11   - port: 80
12     targetPort: 80
13
```

Go to argocd GUI and click on new app




Source: which you had created for manifest files

Branch: main

Path: .

Not secure <https://3.84.245.249:30722/applications?new=%7B%22apiVersion%3A%22Fv1alpha1%22%22kind%3A%22Application%22%22metadata%3A%7B%22name%3A%22demo-app%22%22%7D%7D>

Gmail YouTube Maps

argo
v3.2.3+2b6251d

←

Applications
Settings
User Info
Documentation

Application
+ NEW

CREATE CANCEL

SOURCE

Repository URL
<https://github.com/mujaheed00/argocd-eks-deploy.git> G

Revision
main B


Path
.

DESTINATION

Cluster URL
<https://kubernetes.default.svc> U

Not secure <https://3.84.245.249:30722/applications?new=%7B%22apiVersion%3A%22Fv1alpha1%22%22kind%3A%22Application%22%22metadata%3A%7B%22name%3A%22demo-app%22%22%7D%7D>

Gmail YouTube Maps

argo
v3.2.3+2b6251d

←

Applications
Settings
User Info
Documentation

Application
+ NEW

CREATE CANCEL

DESTINATION

Cluster URL
<https://kubernetes.default.svc>

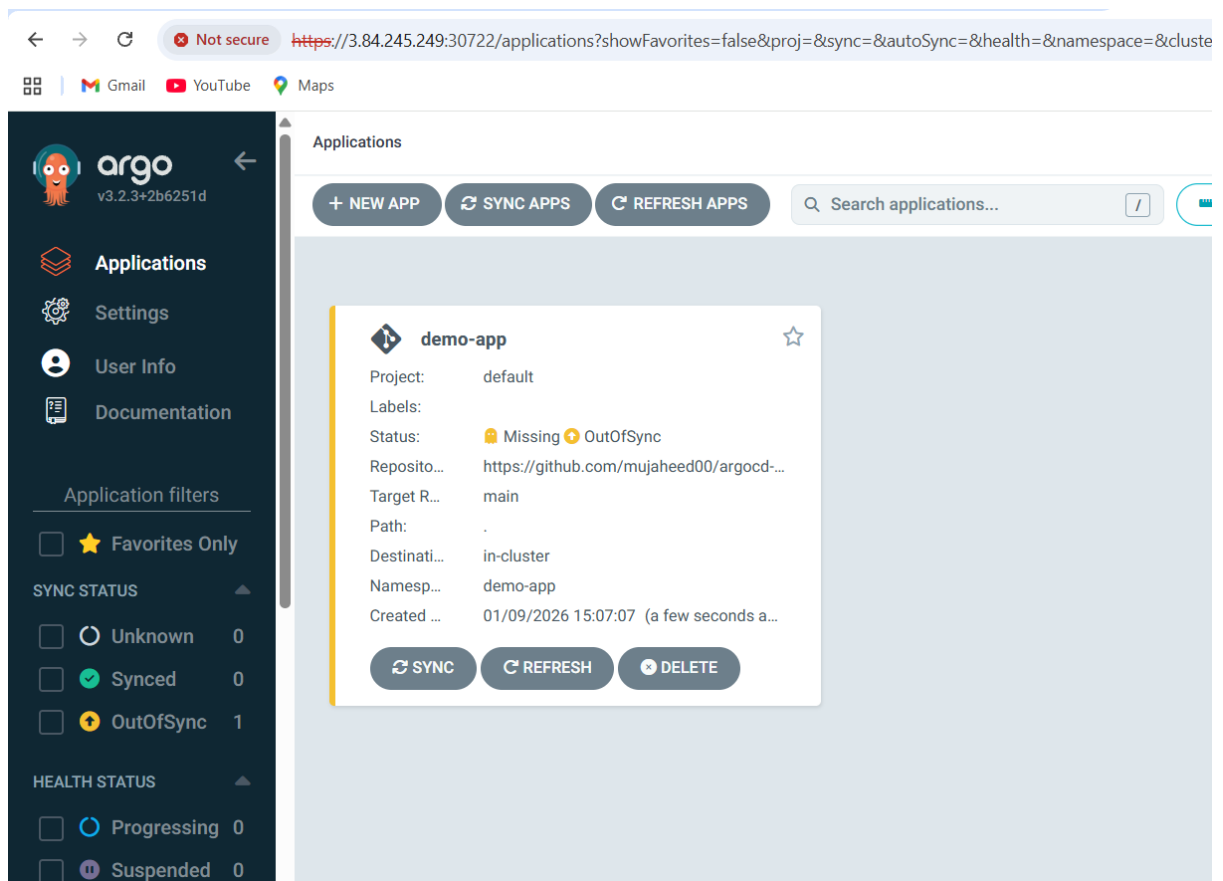
Namespace
demo-app

Directory ▼

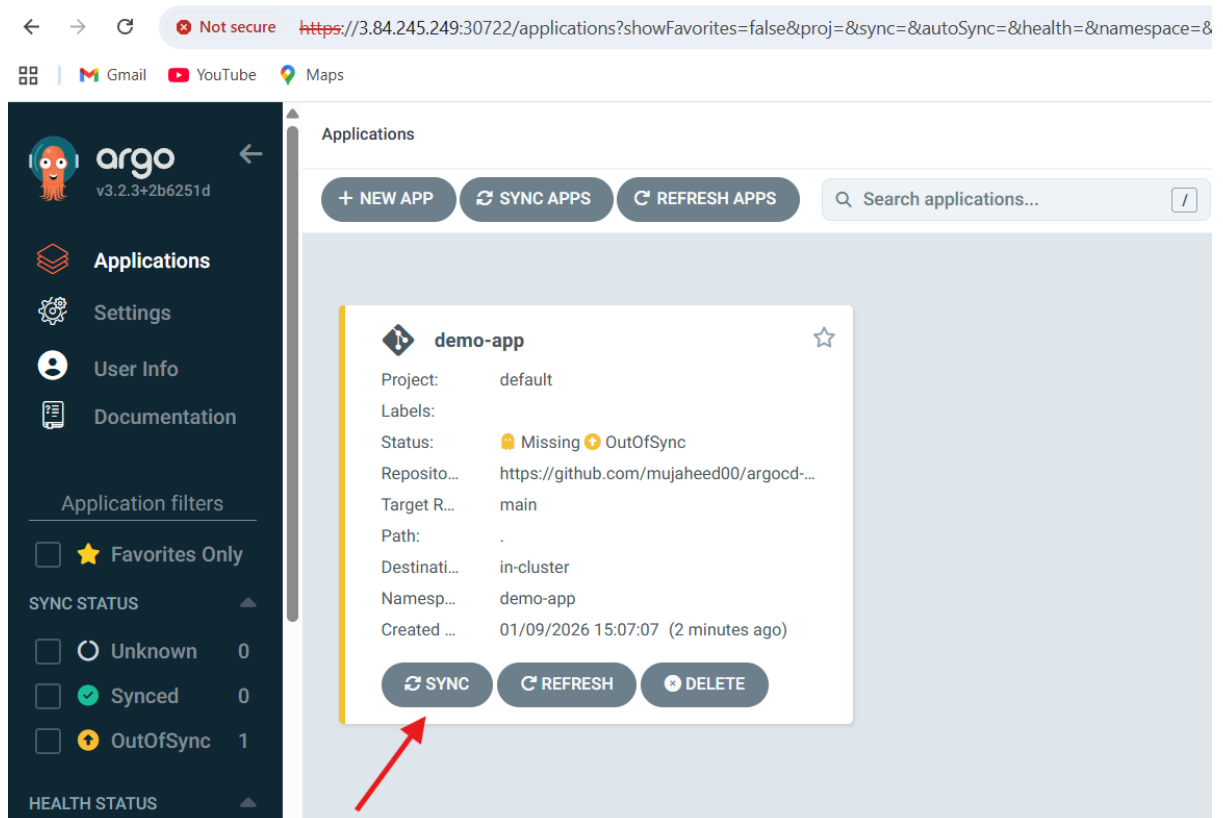
DIRECTORY

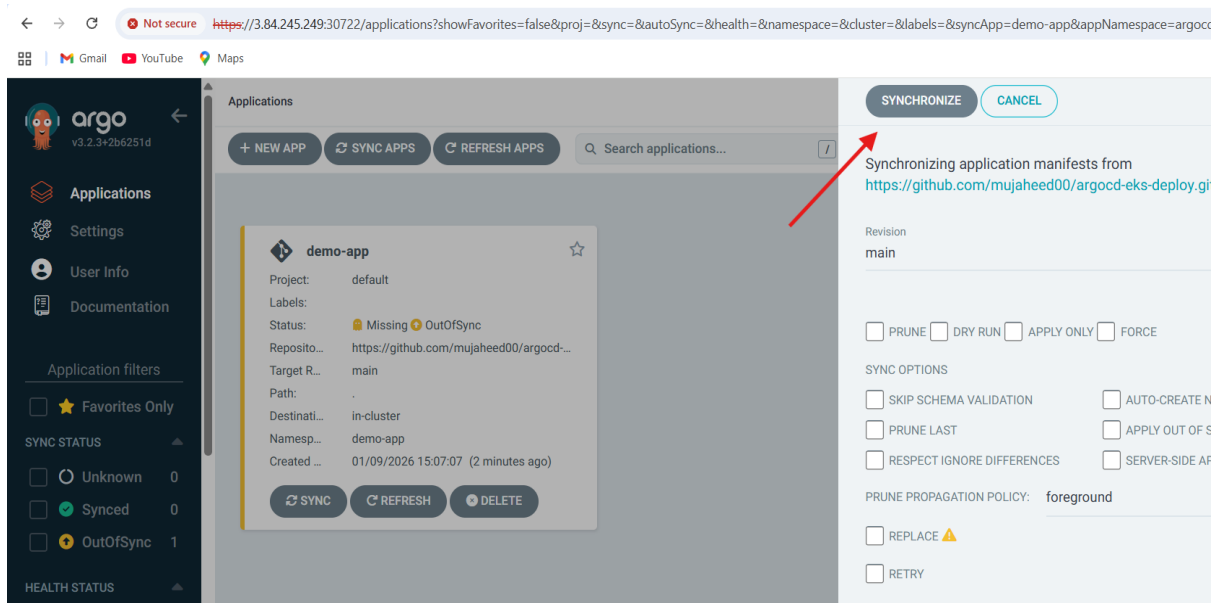
DIRECTORY RECURSE ☐

Click on create



Click on sync



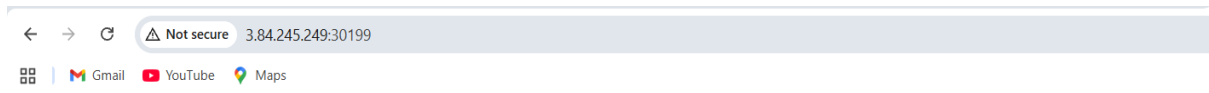


Expose this demo-app in your cluster

- **kubectl get svc -n demo-app**

```
[root@master ~]# kubectl get svc -n demo-app
NAME                TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
demo-service        LoadBalancer  10.98.39.140   <pending>      80:30199/TCP     70s
[root@master ~]#
```

Access in your browser with your ip and port 30199



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.