

How to setup ArgoCD:

ArgoCD is the deployment tool for Kubernetes. We can store our configuration file in github and if we update any file. AgroCD will automatically deployed it.

Pre-requisite:

We need to have running EKS cluster. EC2 instance with kubectI, Terraform, AWS CLI installed.

Install AWS CLI and Configure

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
```

```
sudo apt install unzip
```

```
unzip awscliv2.zip
```

```
sudo ./aws/install
```

```
Initiating: aws/dist/decoders/parsers/1st/include/1stmscl.txt
You can now run: /usr/local/bin/aws --version
[ec2-user@ip-172-31-9-225 ~]$ aws --version
aws-cli/2.15.7 Python/3.11.6 Linux/6.1.66-91.160.amzn2023.x86_64 exe/x86_64.amzn.2023 prompt/off
[ec2-user@ip-172-31-9-225 ~]$
```

Okay now after installing the AWS CLI, let's configure the **AWS CLI** so that it can authenticate and communicate with the AWS environment.

aws configure

```
[ec2-user@ip-172-31-9-225 ~]$ aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]: ap-south-1
Default output format [None]:
[ec2-user@ip-172-31-9-225 ~]$
```

Install and Setup KubectI

Moving forward now we need to set up the **kubectI** also onto the EC2 instance.

```
curl -LO "https://storage.googleapis.com/kubernetes-release/release/$(curl -s
https://storage.googleapis.com/kubernetes-
release/release/stable.txt)/bin/linux/amd64/kubectI"
```

```
chmod +x ./kubectI
```

```
sudo mv ./kubectI /usr/local/bin
```

kubectl version

```
The connection to the server localhost:8080 was refused - did you specify the right host or port?
[ec2-user@ip-172-31-9-225 ~]$ kubectl version
Client Version: v1.29.0
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
The connection to the server localhost:8080 was refused - did you specify the right host or port?
[ec2-user@ip-172-31-9-225 ~]$
```

Follow below steps to install terraform on AmazonLinux.

```
sudo yum install -y yum-utils shadow-utils
sudo yum-config-manager --add-repo
https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
sudo yum -y install terraform
```

```
Complete!
[ec2-user@ip-172-31-9-225 eks-helm]$ terraform version
Terraform v1.6.6
on linux_amd64
[ec2-user@ip-172-31-9-225 eks-helm]$
```

Creating an Amazon EKS cluster using terraform

Code available in <https://github.com/ksnithya/blue-green.git>

git clone <https://github.com/ksnithya/blue-green.git>

cd blue-green

terraform init

terraform plan

terraform apply

aws eks --region ap-south-1 update-kubeconfig --name eks_cluster_demo

Steps:

1. First we create namespace to install argocd.

kubectl create namespace argocd

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL CODEWHISPERER REFERENCE LOG

[ec2-user@ip-10-0-1-112 ~]$ kubectl create namespace argocd
namespace/argocd created
[ec2-user@ip-10-0-1-112 ~]$
```

2. Let's run the following command to install argocd.

```
service/argocd-metrics created
service/argocd-notifications-controller-metrics created
service/argocd-redis created
service/argocd-repo-server created
service/argocd-server created
service/argocd-server-metrics created
deployment.apps/argocd-applicationset-controller created
deployment.apps/argocd-dex-server created
deployment.apps/argocd-notifications-controller created
deployment.apps/argocd-redis created
deployment.apps/argocd-repo-server created
deployment.apps/argocd-server created
statefulset.apps/argocd-application-controller created
networkpolicy.networking.k8s.io/argocd-application-controller-network-policy created
networkpolicy.networking.k8s.io/argocd-applicationset-controller-network-policy created
networkpolicy.networking.k8s.io/argocd-dex-server-network-policy created
networkpolicy.networking.k8s.io/argocd-notifications-controller-network-policy created
networkpolicy.networking.k8s.io/argocd-redis-network-policy created
networkpolicy.networking.k8s.io/argocd-repo-server-network-policy created
```

- ```
kubectl -n argocd get all
```

|                                                             |       |            |                |        |          |
|-------------------------------------------------------------|-------|------------|----------------|--------|----------|
| service/argocd-server-metrics                               |       | ClusterIP  | 172.20.123.129 | <none> | 8083/TCP |
| 5m38s                                                       |       |            |                |        |          |
| NAME                                                        | READY | UP-TO-DATE | AVAILABLE      | AGE    |          |
| deployment.apps/argocd-applicationset-controller            | 1/1   | 1          | 1              | 5m38s  |          |
| deployment.apps/argocd-dex-server                           | 1/1   | 1          | 1              | 5m38s  |          |
| deployment.apps/argocd-notifications-controller             | 1/1   | 1          | 1              | 5m38s  |          |
| deployment.apps/argocd-redis                                | 1/1   | 1          | 1              | 5m38s  |          |
| deployment.apps/argocd-repo-server                          | 1/1   | 1          | 1              | 5m38s  |          |
| deployment.apps/argocd-server                               | 1/1   | 1          | 1              | 5m38s  |          |
| NAME                                                        |       | DESIRED    | CURRENT        | READY  | AGE      |
| replicaset.apps/argocd-applicationset-controller-7786cb7547 |       | 1          | 1              | 1      | 5m38s    |
| replicaset.apps/argocd-dex-server-58574dff5f                |       | 1          | 1              | 1      | 5m38s    |
| replicaset.apps/argocd-notifications-controller-7764bb774d  |       | 1          | 1              | 1      | 5m38s    |
| replicaset.apps/argocd-redis-77bf5b886                      |       | 1          | 1              | 1      | 5m38s    |
| replicaset.apps/argocd-repo-server-5b9977b575               |       | 1          | 1              | 1      | 5m38s    |
| replicaset.apps/argocd-server-6485ccb9c9                    |       | 1          | 1              | 1      | 5m38s    |
| NAME                                                        | READY | AGE        |                |        |          |

- We see the what service mapped to argocd server.

```
kubectl get svc -n argocd
```

| NAME                                                        | DESIRED | CURRENT | READY | AGE   |
|-------------------------------------------------------------|---------|---------|-------|-------|
| replicaset.apps/argocd-applicationset-controller-7786cb7547 | 1       | 1       | 1     | 5m38s |
| replicaset.apps/argocd-dex-server-58574dff5f                | 1       | 1       | 1     | 5m38s |
| replicaset.apps/argocd-notifications-controller-7764bb774d  | 1       | 1       | 1     | 5m38s |
| replicaset.apps/argocd-redis-77bf5b886                      | 1       | 1       | 1     | 5m38s |
| replicaset.apps/argocd-repo-server-5b9977b575               | 1       | 1       | 1     | 5m38s |
| replicaset.apps/argocd-server-6485ccb9c9                    | 1       | 1       | 1     | 5m38s |

| NAME                                           | READY | AGE   |
|------------------------------------------------|-------|-------|
| statefulset.apps/argocd-application-controller | 1/1   | 5m38s |

```
[ec2-user@ip-10-0-1-83 ~]$ kubectl get svc -n argocd
```

| NAME                                    | TYPE      | CLUSTER-IP     | EXTERNAL-IP | PORT(S)                    |
|-----------------------------------------|-----------|----------------|-------------|----------------------------|
| argocd-applicationset-controller        | ClusterIP | 172.20.233.80  | <none>      | 7000/TCP,8080/TCP          |
| argocd-dex-server                       | ClusterIP | 172.20.20.250  | <none>      | 5556/TCP,5557/TCP,5558/TCP |
| argocd-metrics                          | ClusterIP | 172.20.175.242 | <none>      | 8082/TCP                   |
| argocd-notifications-controller-metrics | ClusterIP | 172.20.97.31   | <none>      | 9001/TCP                   |
| argocd-redis                            | ClusterIP | 172.20.226.48  | <none>      | 6379/TCP                   |
| argocd-repo-server                      | ClusterIP | 172.20.138.232 | <none>      | 8081/TCP,8084/TCP          |
| argocd-server                           | ClusterIP | 172.20.203.87  | <none>      | 80/TCP,443/TCP             |
| argocd-server-metrics                   | ClusterIP | 172.20.123.129 | <none>      | 8083/TCP                   |

```
[ec2-user@ip-10-0-1-83 ~]$
```

- Argocd-server service is mapping to "ClusterIP". We need to change it to "LoadBalancer" or "NodePort" to access the argocd UI from outside world.

```
kubectl patch svc argocd-server -n argocd -p '{"spec": {"type": "LoadBalancer"}}'
```

- Now service will be changed to "LoadBalancer"

```
[ec2-user@ip-10-0-1-83 ~]$ kubectl patch svc argocd-server -n argocd -p '{"spec": {"type": "LoadBalancer"}}'
```

```
service/argocd-server patched
```

```
[ec2-user@ip-10-0-1-83 ~]$ kubectl get svc -n argocd
```

| NAME                                    | PORT(S)                    | TYPE         | CLUSTER-IP     | EXTERNAL-IP                             |
|-----------------------------------------|----------------------------|--------------|----------------|-----------------------------------------|
| argocd-applicationset-controller        | 7000/TCP,8080/TCP          | ClusterIP    | 172.20.233.80  | <none>                                  |
| argocd-dex-server                       | 5556/TCP,5557/TCP,5558/TCP | ClusterIP    | 172.20.20.250  | <none>                                  |
| argocd-metrics                          | 8082/TCP                   | ClusterIP    | 172.20.175.242 | <none>                                  |
| argocd-notifications-controller-metrics | 9001/TCP                   | ClusterIP    | 172.20.97.31   | <none>                                  |
| argocd-redis                            | 6379/TCP                   | ClusterIP    | 172.20.226.48  | <none>                                  |
| argocd-repo-server                      | 8081/TCP,8084/TCP          | ClusterIP    | 172.20.138.232 | <none>                                  |
| argocd-server                           | 80:31087/TCP,443:30947/TCP | LoadBalancer | 172.20.203.87  | aeb1d7b98f31441dca402f7f0e031256-388578 |
| argocd-server-metrics                   | 8083/TCP                   | ClusterIP    | 172.20.123.129 | <none>                                  |

```
[ec2-user@ip-10-0-1-83 ~]$
```

```
[ec2-user@ip-172-31-42-60 helm]$ kubectl get svc argocd-server -n argocd
```

| NAME          | TYPE         | CLUSTER-IP   | EXTERNAL-IP                                                              |
|---------------|--------------|--------------|--------------------------------------------------------------------------|
| argocd-server | LoadBalancer | 172.20.26.38 | ac56ae5e4ee004f588e14968903ea1b2-1457899865.ap-south-1.elb.amazonaws.com |

80:31894/TCP,443:31830/TCP 9m11s

Now we can login to argocd using Loadbalancer.

<http://ac56ae5e4ee004f588e14968903ea1b2-1457899865.ap-south-1.elb.amazonaws.com>

- Default username is "admin" and password we can get it from secret.

```
[ec2-user@ip-172-31-42-60 helm]$ kubectl get secret -n argocd
```

| NAME | TYPE | DATA | AGE |
|------|------|------|-----|
|------|------|------|-----|

```

argocd-initial-admin-secret Opaque 1 20m
argocd-notifications-secret Opaque 0 21m
argocd-secret Opaque 5 21m
[ec2-user@ip-172-31-42-60 helm]$
[ec2-user@ip-172-31-42-60 helm]$ kubectl get secret -n argocd argocd-initial-admin-secret -o yaml
apiVersion: v1
data:
 password: Vld1UG5yQ21hT2FqczJtVQ==
kind: Secret
metadata:
 creationTimestamp: "2023-12-28T11:30:26Z"
 name: argocd-initial-admin-secret
 namespace: argocd
 resourceVersion: "3173"
 uid: cb246043-44d6-499e-bfa0-7a80857ea2aa
type: Opaque
[ec2-user@ip-172-31-42-60 helm]$
[ec2-user@ip-172-31-42-60 helm]$ echo "Vld1UG5yQ21hT2FqczJtVQ==" | base64 -d
VWuPnrCmaOajs2mU[ec2-user@ip-172-31-42-60 helm]$
VWuPnrCmaOajs2mU[ec2-user@ip-172-31-42-60 helm]$ kubectl get secret -n argocd

```

| NAME                        | TYPE   | DATA | AGE |
|-----------------------------|--------|------|-----|
| argocd-initial-admin-secret | Opaque | 1    | 20m |
| argocd-notifications-secret | Opaque | 0    | 21m |
| argocd-secret               | Opaque | 5    | 21m |

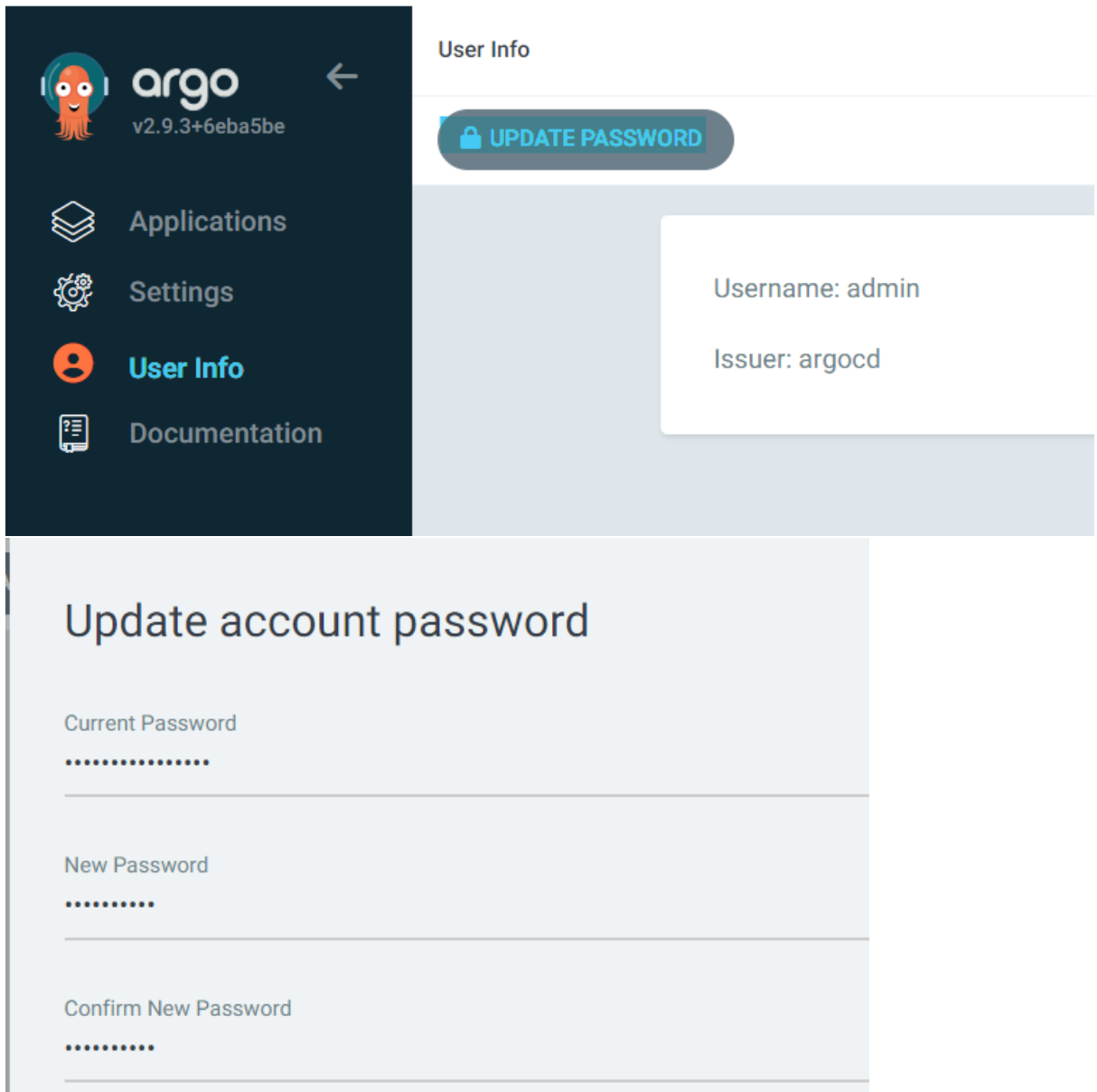
```

[ec2-user@ip-172-31-42-60 helm]$
[ec2-user@ip-172-31-42-60 helm]$ kubectl get secret -n argocd argocd-initial-admin-secret -o yaml
apiVersion: v1
data:
 password: Vld1UG5yQ21hT2FqczJtVQ==
kind: Secret
metadata:
 creationTimestamp: "2023-12-28T11:30:26Z"
 name: argocd-initial-admin-secret
 namespace: argocd
 resourceVersion: "3173"
 uid: cb246043-44d6-499e-bfa0-7a80857ea2aa
type: Opaque
[ec2-user@ip-172-31-42-60 helm]$ echo "Vld1UG5yQ21hT2FqczJtVQ==" | base64 -d
VWuPnrCmaOajs2mU[ec2-user@ip-172-31-42-60 helm]$

```

We can use this password to login. After login it is recommended to change the password.

<http://ac56ae5e4ee004f588e14968903ea1b2-1457899865.ap-south-1.elb.amazonaws.com>



The image shows the Argo CD web interface. On the left is a dark sidebar with the Argo logo (an orange octopus) and version 'v2.9.3+6eba5be'. Below the logo are links for 'Applications', 'Settings', 'User Info' (highlighted in blue), and 'Documentation'. The main content area has a 'User Info' header with an 'UPDATE PASSWORD' button. Below this, a white box displays 'Username: admin' and 'Issuer: argocd'. The 'Update account password' form contains three password input fields: 'Current Password', 'New Password', and 'Confirm New Password', each with a masked password (dots).

argo v2.9.3+6eba5be

Applications

Settings

User Info

Documentation

User Info

UPDATE PASSWORD

Username: admin

Issuer: argocd

## Update account password

Current Password

.....

New Password

.....

Confirm New Password

.....

Click on "Save password".

8. Now we install argocd CLI.

```
sudo curl -sSL -o /usr/local/bin/argocd
https://github.com/argoproj/argo-cd/releases/latest/download/argocd-
linux-amd64
sudo chmod +x /usr/local/bin/argocd
```



```
[ec2-user@ip-172-31-42-60 helm]$ argocd version
argocd: v2.9.3+6eba5be
 BuildDate: 2023-12-01T23:24:09Z
 GitCommit: 6eba5be864b7e031871ed7698f523336dfe75c7
 GitTreeState: clean
 GoVersion: go1.21.4
 Compiler: gc
 Platform: linux/amd64
FATA[0000] Failed to establish connection to ac56ae5e4ee004f588e14968903ealb2-145789986
localhost, argocd-server, argocd-server.argocd, argocd-server.argocd.svc, argocd-server
m
[ec2-user@ip-172-31-42-60 helm]$
```

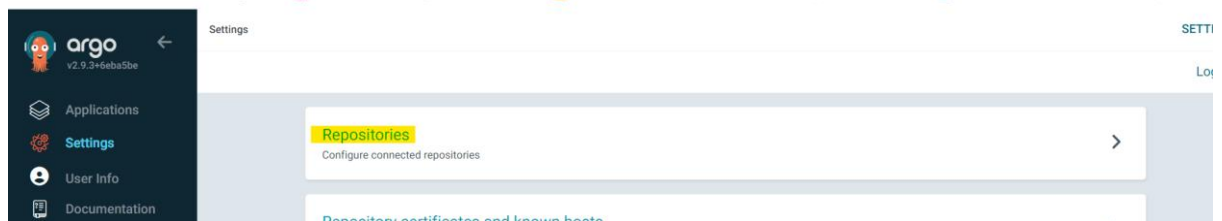
9. Now we login to argocd CLI.

```
argocd login $ARGOCD_SERVER --username admin --password $ARGO_PWD --insecure
```

```
Summary: latest 6e4132d6190 3 weeks ago 72589
[ec2-user@ip-172-31-42-60 ~]$ sudo argocd login ac56ae5e4ee004f588e14968903ealb2-1457899865.ap-south-1.elb.amazonaws.com --username admin --password Venkat@123 --insecure
'admin:login' logged in successfully
Context 'ac56ae5e4ee004f588e14968903ealb2-1457899865.ap-south-1.elb.amazonaws.com' updated
[ec2-user@ip-172-31-42-60 ~]$
```

10. Now we create deployment app using argocd UI.

In the ArgoCD web interface and click on Settings -> Repositories



On the next page click on Connect Repo and fill input the necessary details.

CONNECT SAVE AS CREDENTIALS TEMPLATE CANCEL

Choose your connection method:

VIA HTTPS ▼

CONNECT REPO USING HTTPS

Type  
git

Project  
default

Repository URL  
https://github.com/SeronDevops/ARGO-CICD-KUBERNETES.git

Activate Windows  
Go to Settings to activate Windows.

Under “Repository URL” we can give the github link where our code exist.  
Repo: https://github.com/ksnithya/blue-green.git

Fill out just these three places and leave the rest options. Head up and click on Connect

Settings / Repositories REPOSITORIES

+ CONNECT REPO REFRESH LIST Log out

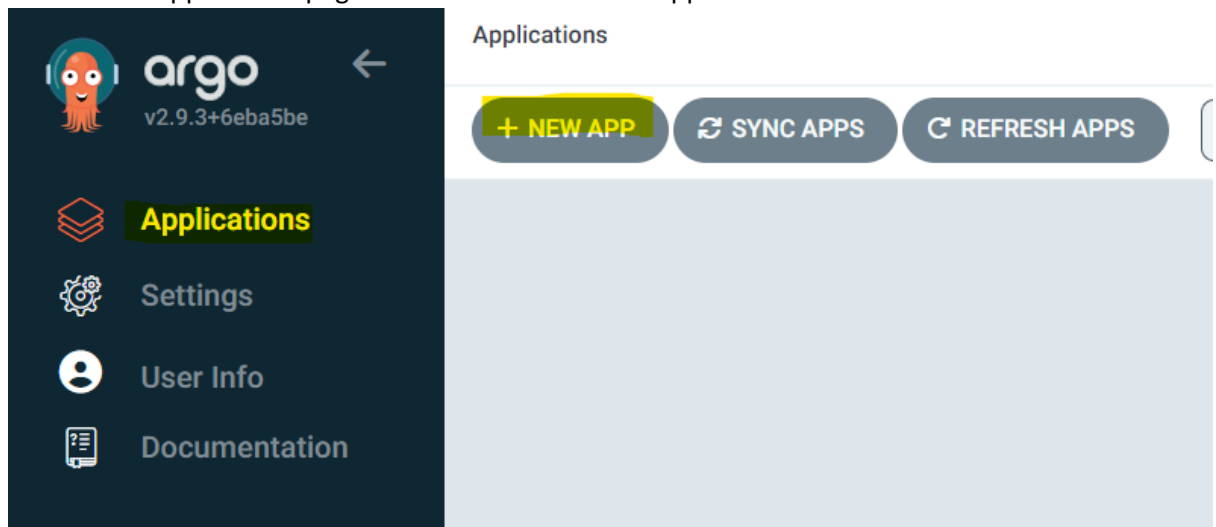
| TYPE | NAME       | REPOSITORY                                 | CONNECTION STATUS |
|------|------------|--------------------------------------------|-------------------|
| git  | blue-green | https://github.com/ksnithya/blue-green.git | Successful        |

Activate Windows  
Go to Settings to activate Windows.



Now we have connected our GitHub repository with ArgoCD. Let's go build our application.

11. Click on the Applications page and click on create new app



Fill the details as above.

This is a 'CREATE' form for a new application. At the top are 'CREATE' and 'CANCEL' buttons. The form is divided into sections. The 'GENERAL' section includes 'Application Name' (filled with 'python-app'), 'Project Name' (filled with 'default'), and 'SYNC POLICY' (set to 'Automatic'). There are checkboxes for 'PRUNE RESOURCES' and 'SELF HEAL', both of which are currently unchecked. An 'EDIT AS YAML' button is in the top right corner of the form area.

Fill the source details. Code is under blue directory so we need to give the directory path where our code exist.



This section shows the 'SOURCE' details. It includes a 'Repository URL' field filled with 'https://github.com/ksnithya/blue-green.git', a 'Revision' field filled with 'HEAD', and a 'Path' field filled with 'blue'. To the right of the repository URL is a 'GIT' icon with a checkmark. A 'Branches' dropdown menu is visible on the right side of the revision field.

Select the cluster URL and Namespace. Now click on create. It will create the app.

DESTINATION



Cluster URL  
https://kubernetes.default.svc

Namespace  
default

 **python-app** 

Project: default

Labels:

Status:  Healthy  Synced

Repository: https://github.com/ksnithya/blue-green.git

Target Re... HEAD




Path: blue

Destinati... in-cluster

Namespa... default

Created At: 12/28/2023 19:08:43 (a few seconds ago)

Last Sync: 12/28/2023 19:08:44 (a few seconds ago)

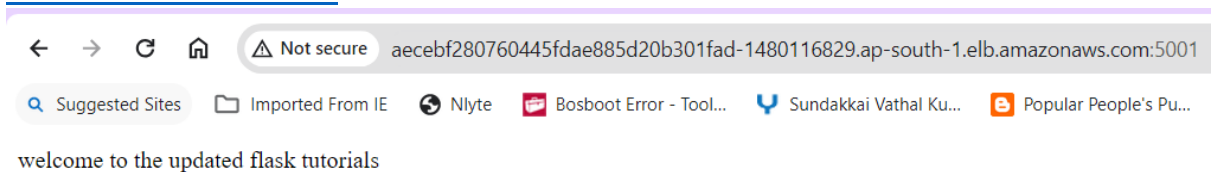
 SYNC  REFRESH  DELETE

We can see our app deployed in our cluster server.

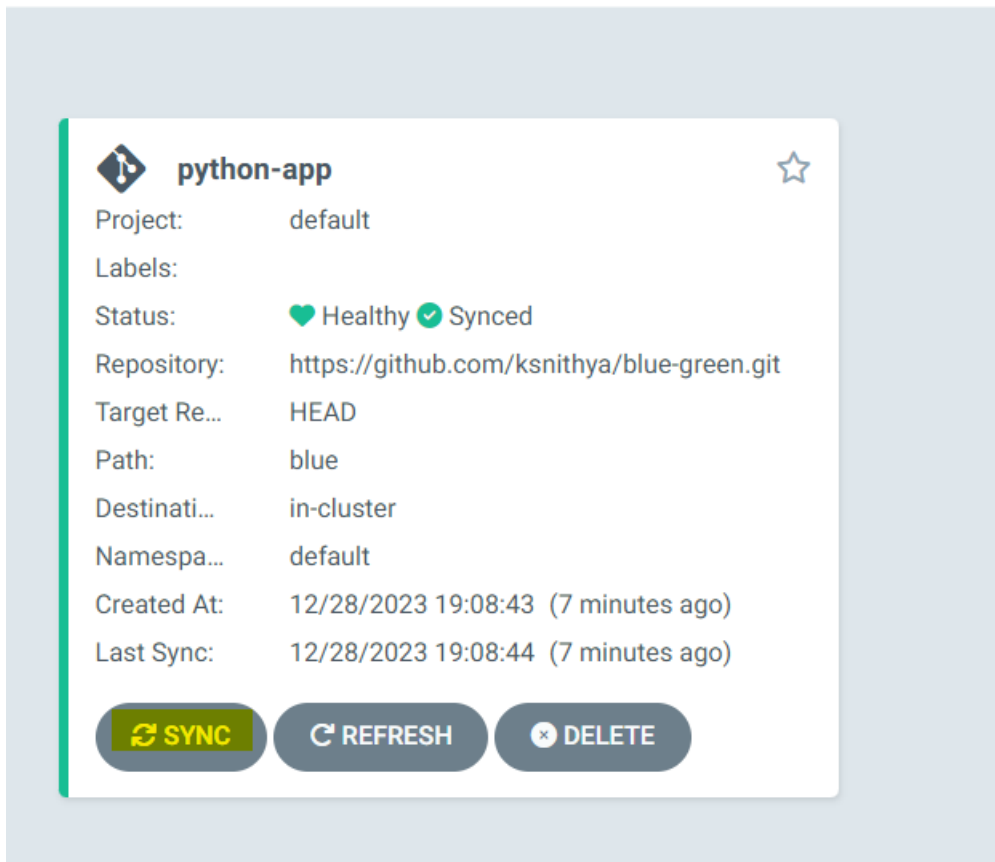
```
[ec2-user@ip-172-31-42-60 ~]$ kubectl get deploy
NAME READY UP-TO-DATE AVAILABLE AGE
python-flask 1/1 1 1 36s
[ec2-user@ip-172-31-42-60 ~]$ kubectl get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 172.20.0.1 <none> 443/TCP 145m
python-load LoadBalancer 172.20.10.104 aecebf280760445fdae885d20b301fad-1480116829.ap-south-1.elb.amazonaws.com 5001:32010/TCP 42s
[ec2-user@ip-172-31-42-60 ~]$
```

We can access the app using Loadbalancer URL.

<http://aecebf280760445fdae885d20b301fad-1480116829.ap-south-1.elb.amazonaws.com:5001>



12. Now we can update our python app and create new image and push to dockerhub. Update that image details in our deploy.yml file in our repo and click on sync in app.

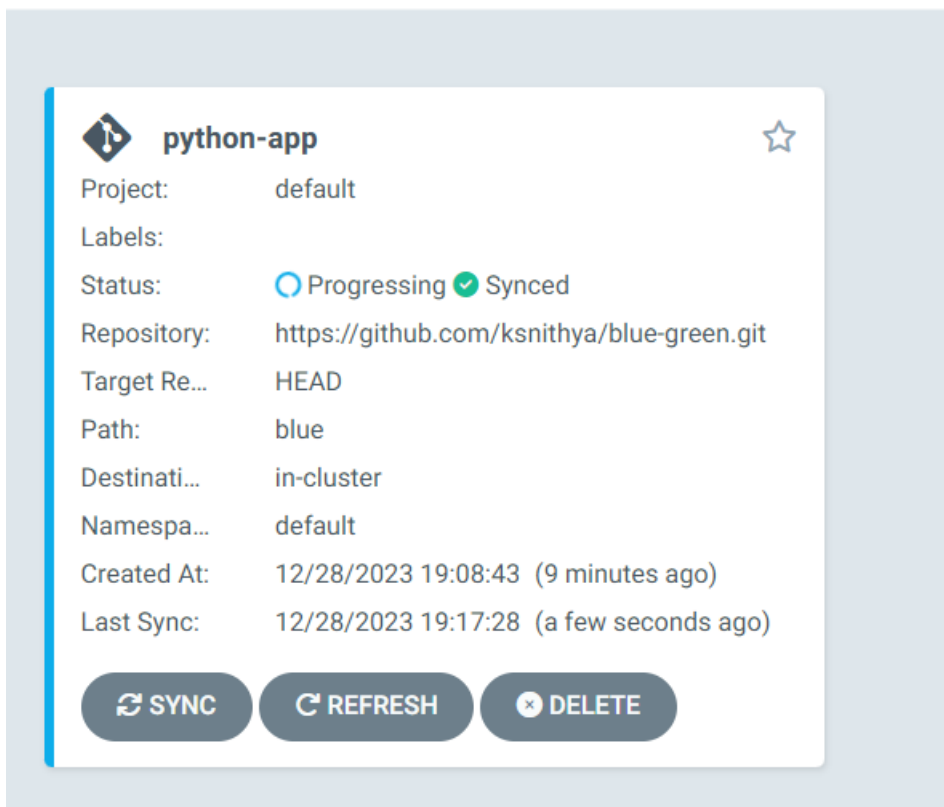


The screenshot shows a deployment card for 'python-app'. It includes a star icon in the top right corner. The card lists the following details:

- Project: default
- Labels:
- Status: ♥ Healthy ✓ Synced
- Repository: <https://github.com/ksnithya/blue-green.git>
- Target Re...: HEAD
- Path: blue
- Destinati...: in-cluster
- Namespa...: default
- Created At: 12/28/2023 19:08:43 (7 minutes ago)
- Last Sync: 12/28/2023 19:08:44 (7 minutes ago)

At the bottom, there are three buttons: **SYNC** (with a circular arrow icon), **REFRESH** (with a circular arrow icon), and **DELETE** (with a trash can icon).



It will sync and redeploy it.



The screenshot shows the same deployment card for 'python-app', but the status has changed to 'Progressing'. The details are as follows:

- Project: default
- Labels:
- Status: ○ Progressing ✓ Synced
- Repository: <https://github.com/ksnithya/blue-green.git>
- Target Re...: HEAD
- Path: blue
- Destinati...: in-cluster
- Namespa...: default
- Created At: 12/28/2023 19:08:43 (9 minutes ago)
- Last Sync: 12/28/2023 19:17:28 (a few seconds ago)

The buttons at the bottom remain the same: **SYNC**, **REFRESH**, and **DELETE**.



 **python-app** 

Project:

default

Labels:

Status:

 Healthy  Synced

Repository:

https://github.com/ksnithya/blue-green.git

Target Re...

HEAD

Path:

blue

Destinati...

in-cluster

Namespa...


default


Created At:


12/28/2023 19:08:43 (16 minutes ago)

Last Sync:





12/28/2023 19:22:13 (2 minutes ago)

 SYNC

 REFRESH


 DELETE


Now app is deployed with new image.


   


Not secure


aecebf280760445fdae885d20b301fad-1480116829.ap-south-1.elb.amazonaws.com:5001


 Suggested Sites


 Imported From IE

 Nlyte

 Bosboot Error - Tool...

 Sundakkai Vathal Ku...

 Popular People's Pu...



welcome to the flask updated tutorials new