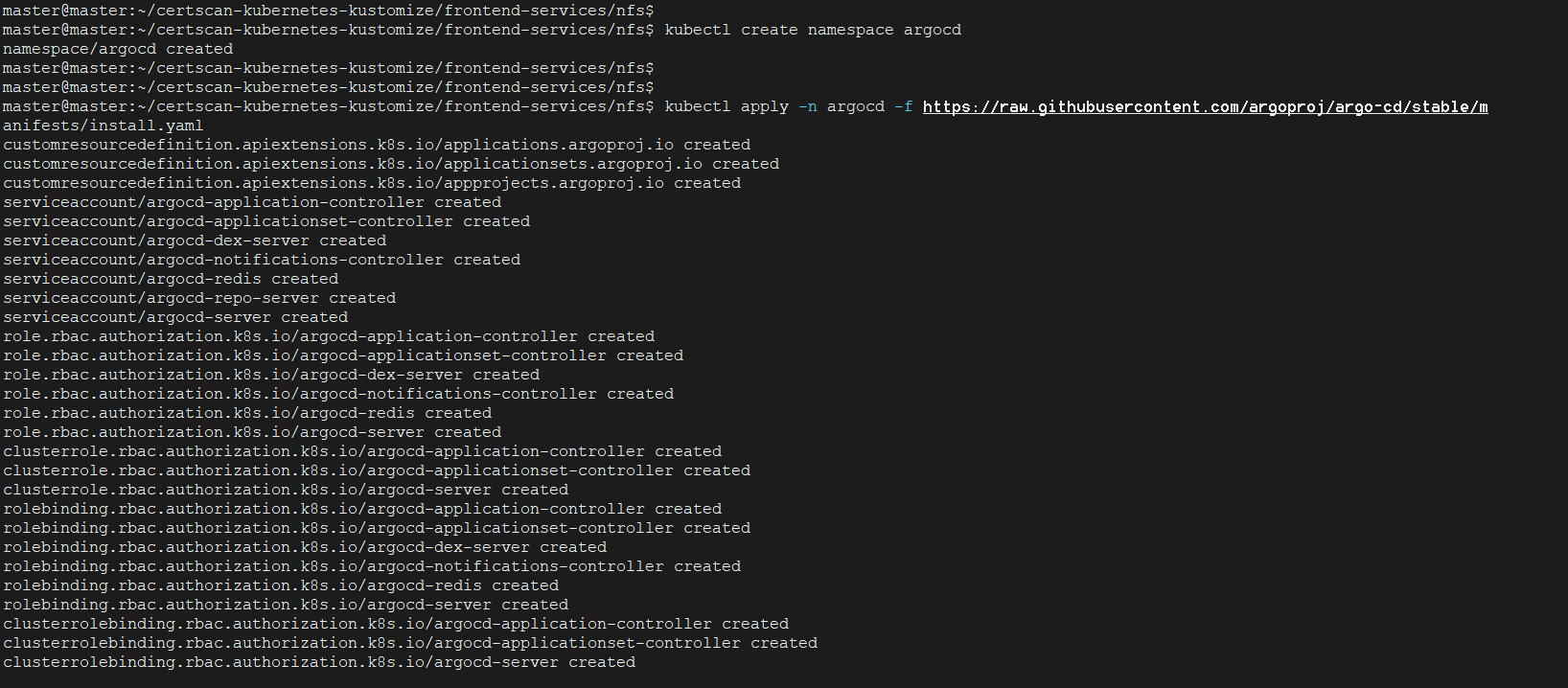
**ArgoCD Setup Guide**

Go to the argocd official site to install argocd on your kubernetes cluster.  
<https://argo-cd.readthedocs.io/en/stable/getting_started/>

Or follow the below steps to install it.

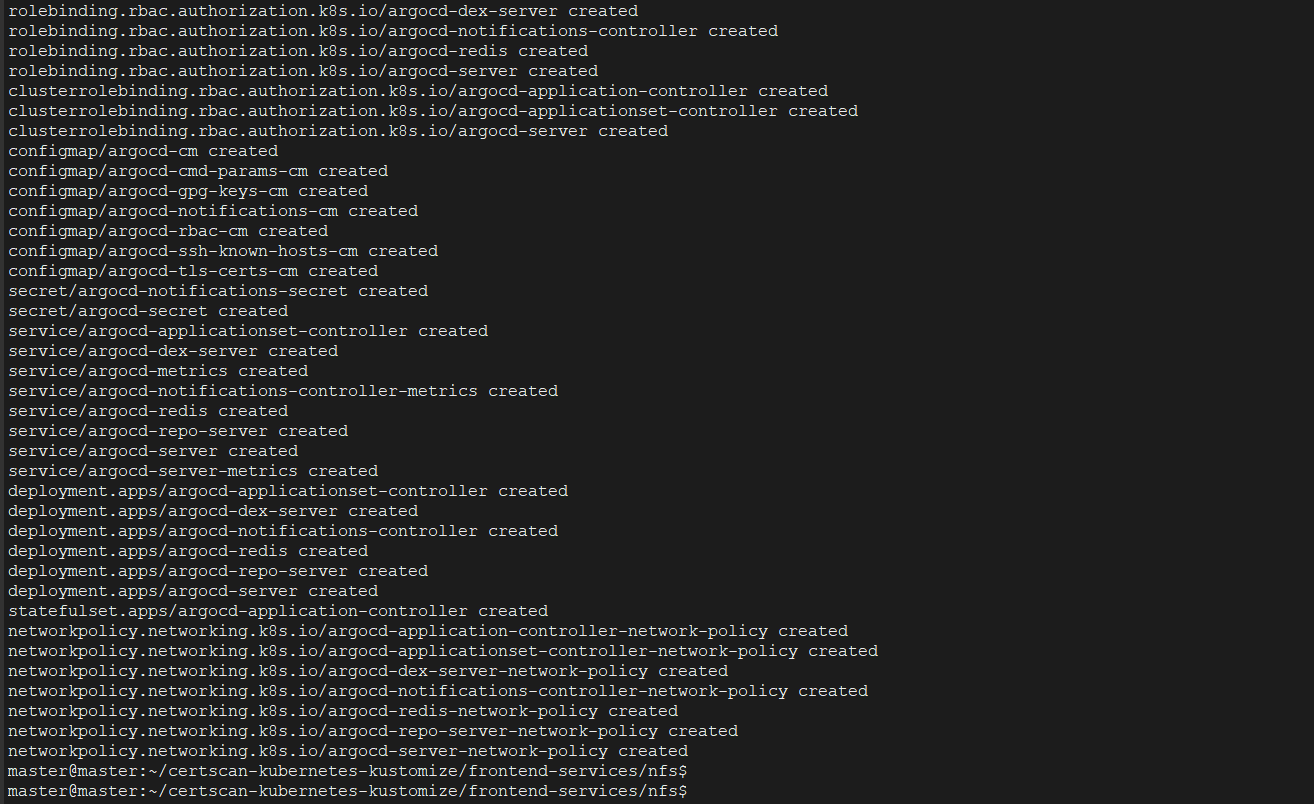
1. Create a namespace for it

**kubectl create namespace argocd**

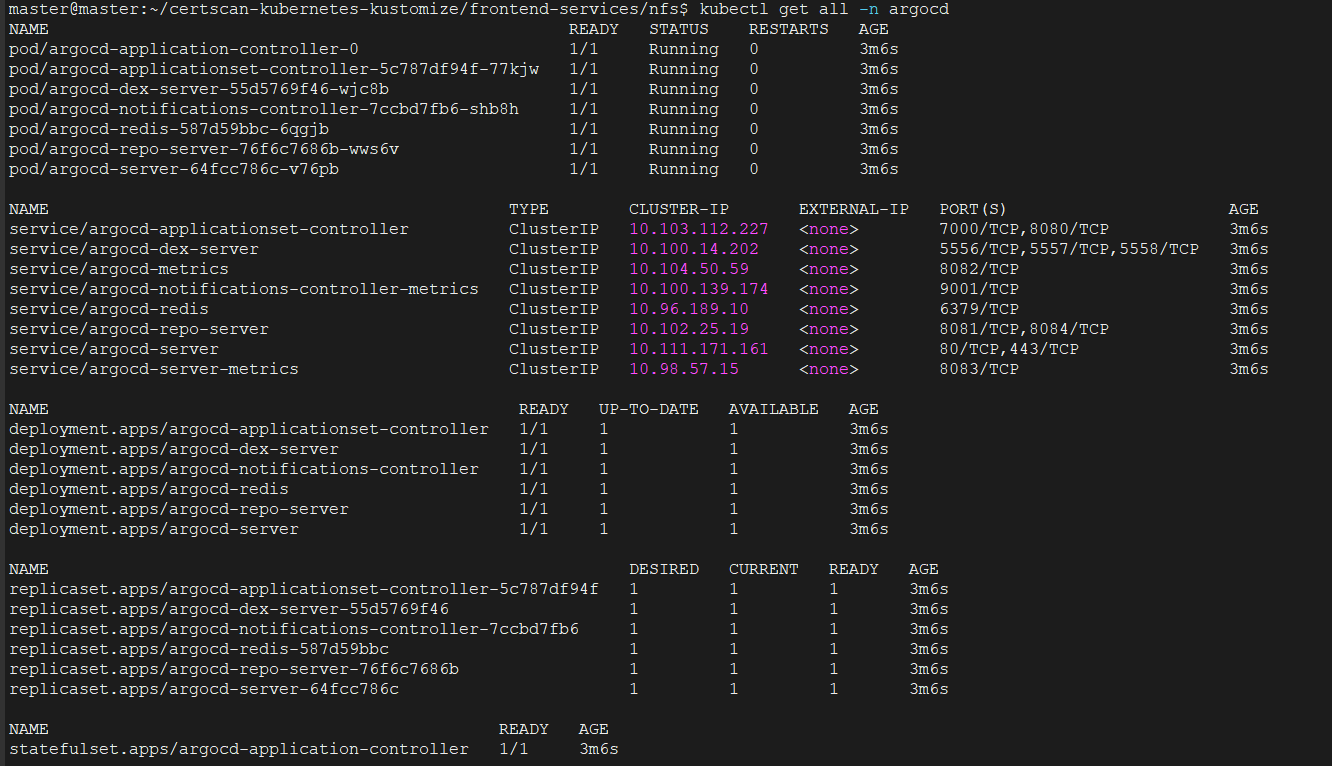
****

1. Now install it using this manifest file

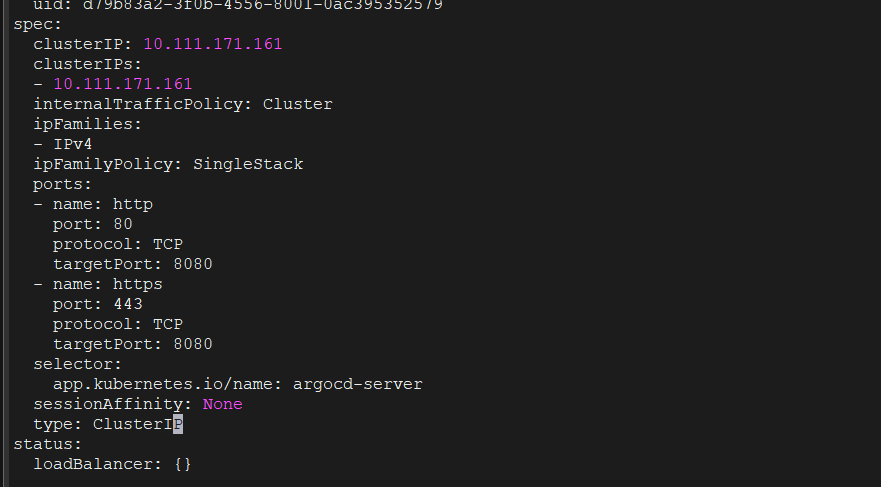
**kubectl apply -n argocd -f** [**https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml**](https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml)

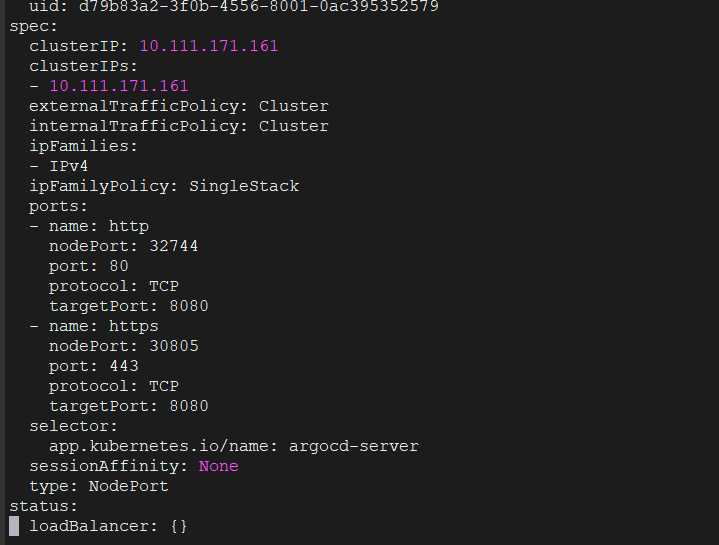
****

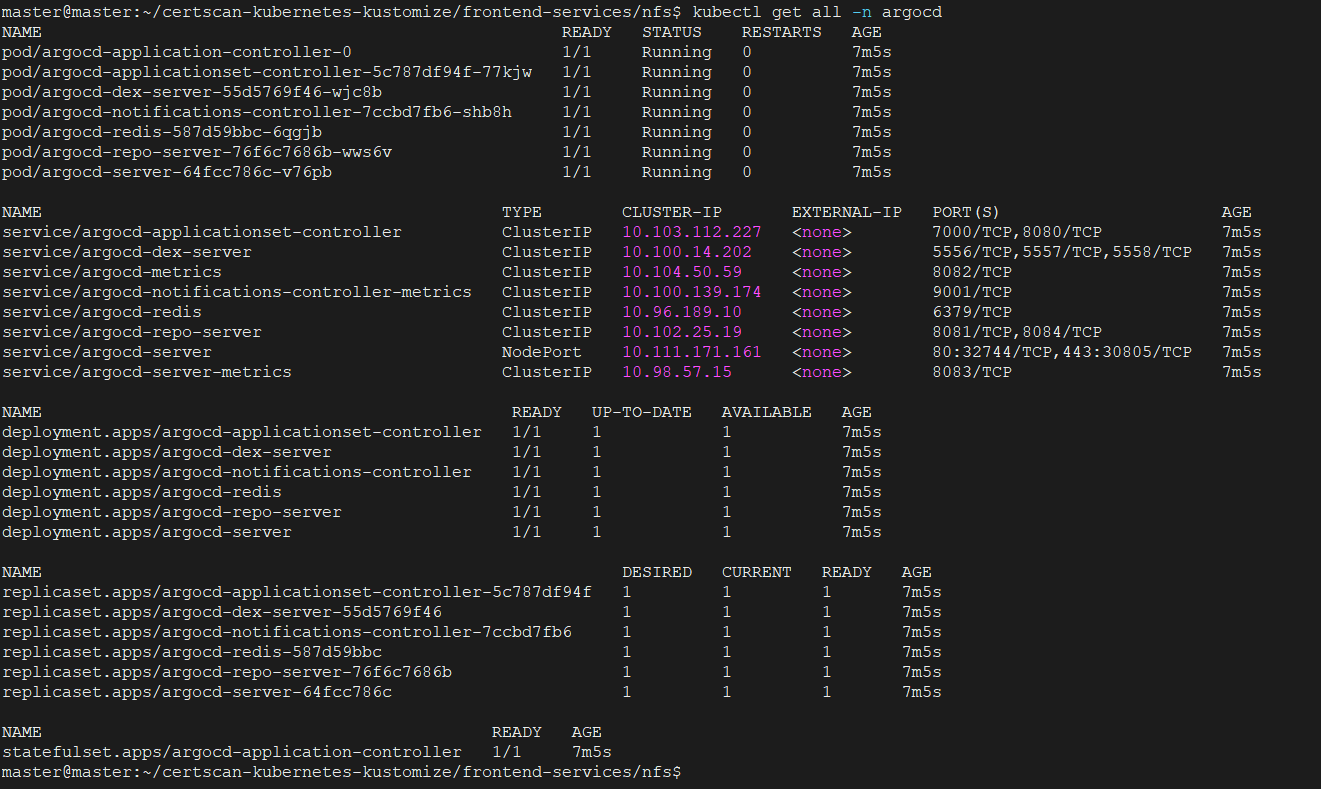
1. Fetch the pods and services and Verify Deployment:  
   **kubectl get all -n argocd**

****

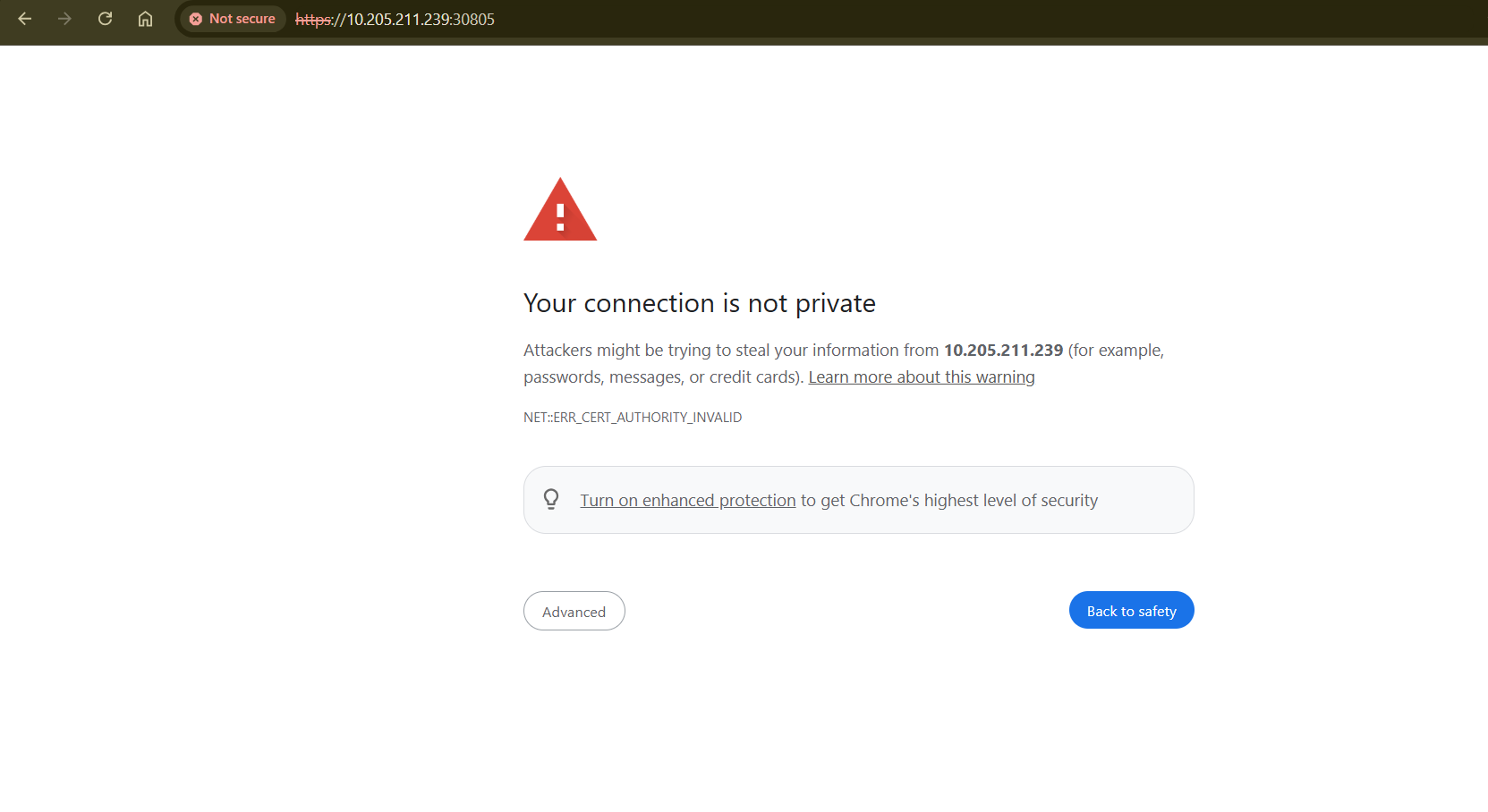
1. Now, open the service/arogcd-server file and change service type from ClusterIP to NodePort:  
   **kubectl edit service/argocd-server -n argocd**

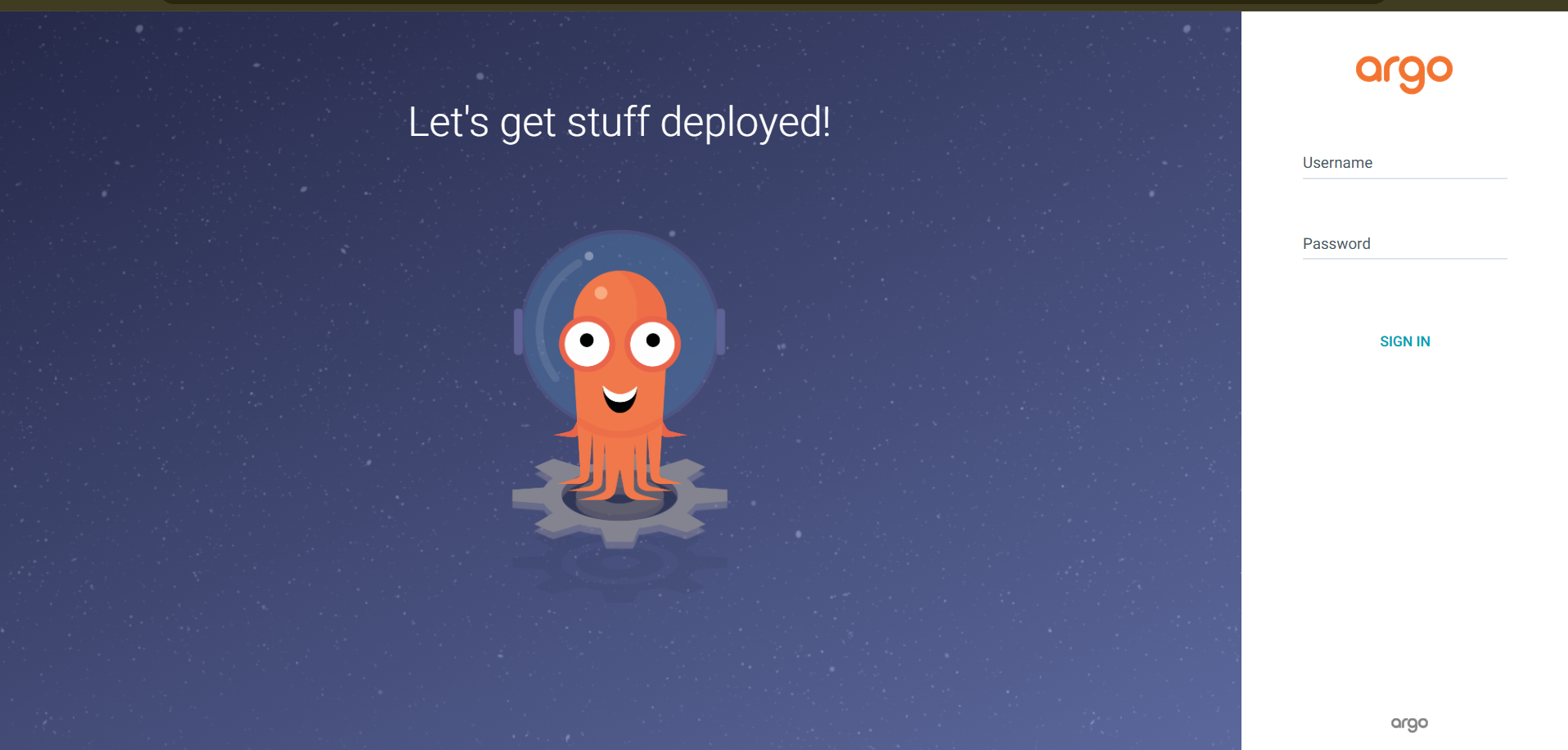
  




  
  
5) Note down the port on which argocd-server is listening. Access the ArgoCD UI using the link **https://<Node-IP>:<Port-of-ArgoCD-Server>**

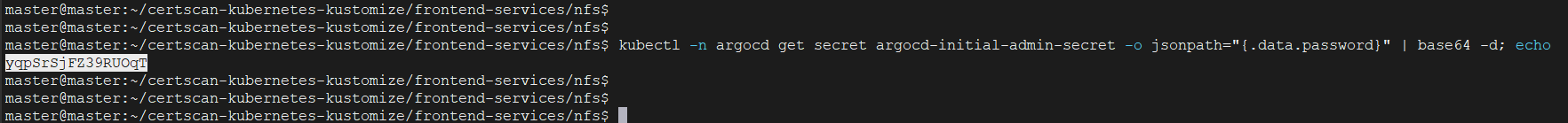
Example: [**https://10.205.211.239:30805**](https://10.205.211.239:30805/)

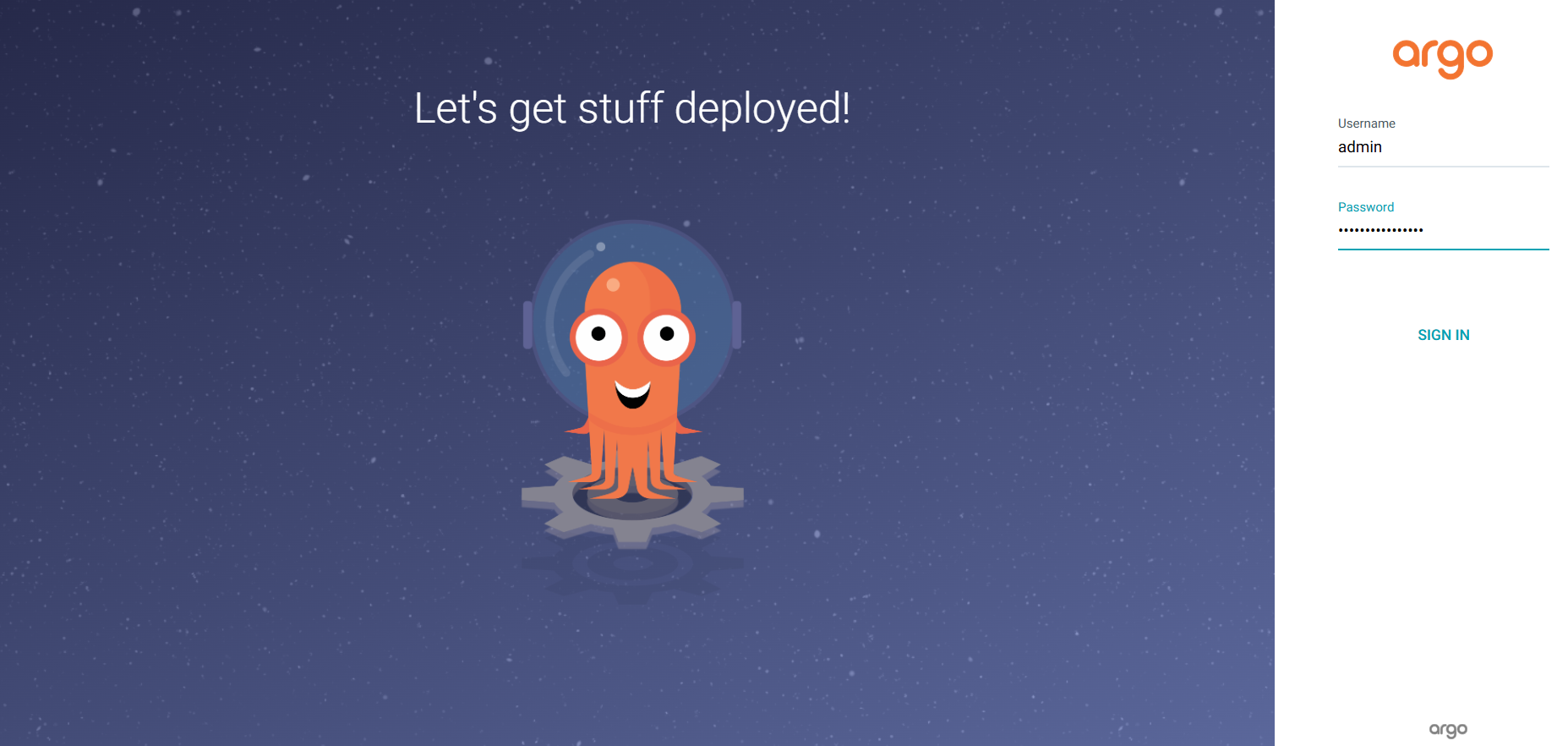




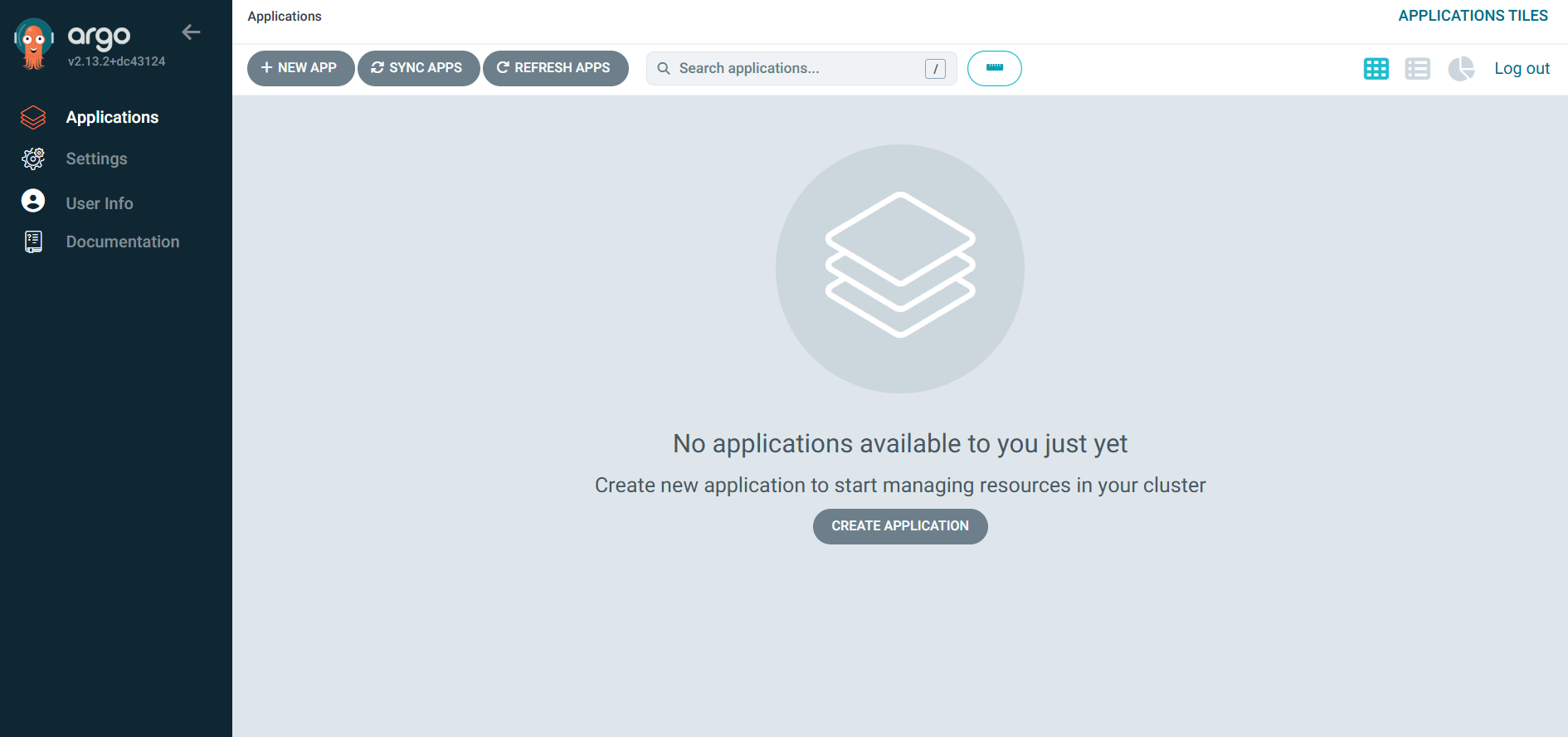
6) Get the password from Initial-admin-password secret using this command:  
 **kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath="{.data.password}" | base64 -d; echo**

And username is **“admin”**

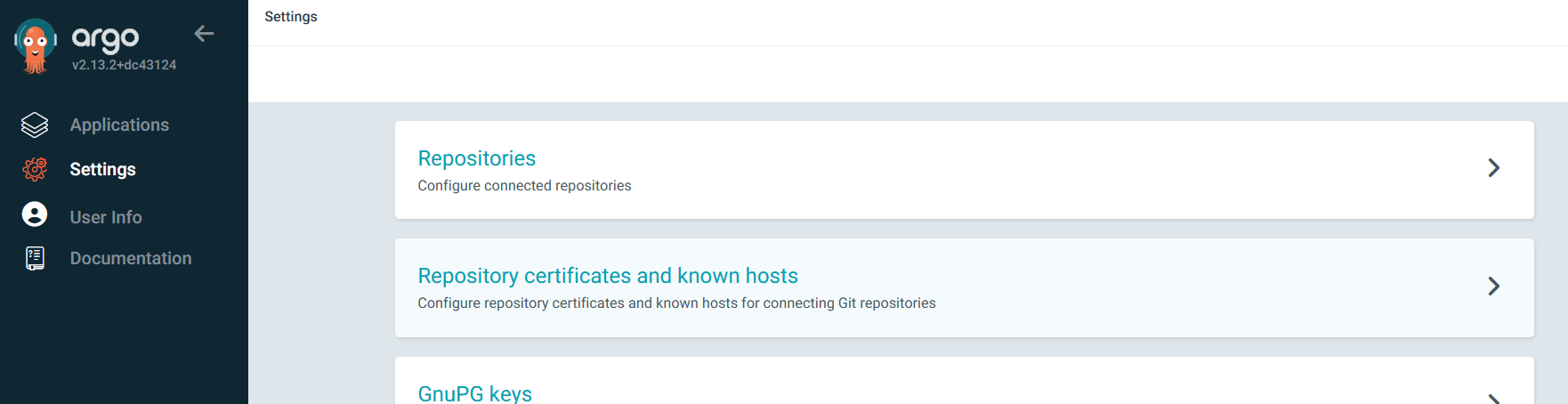


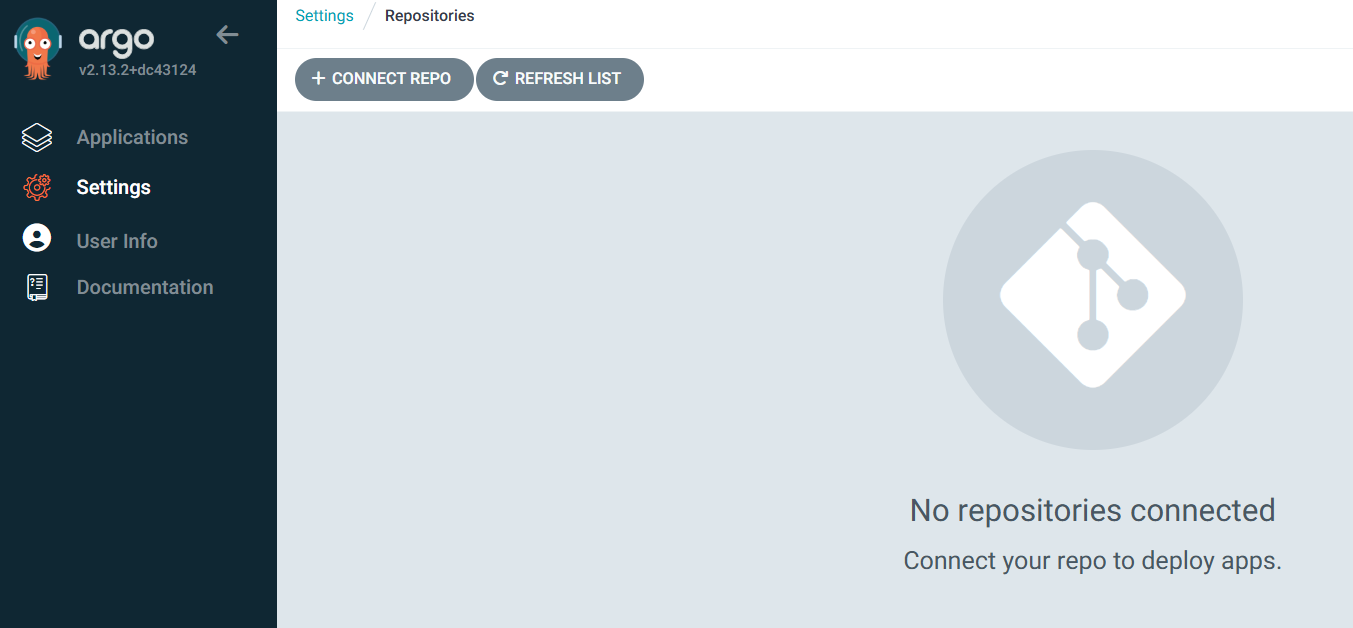


7) Now you can see the ArgoCD dashboard.

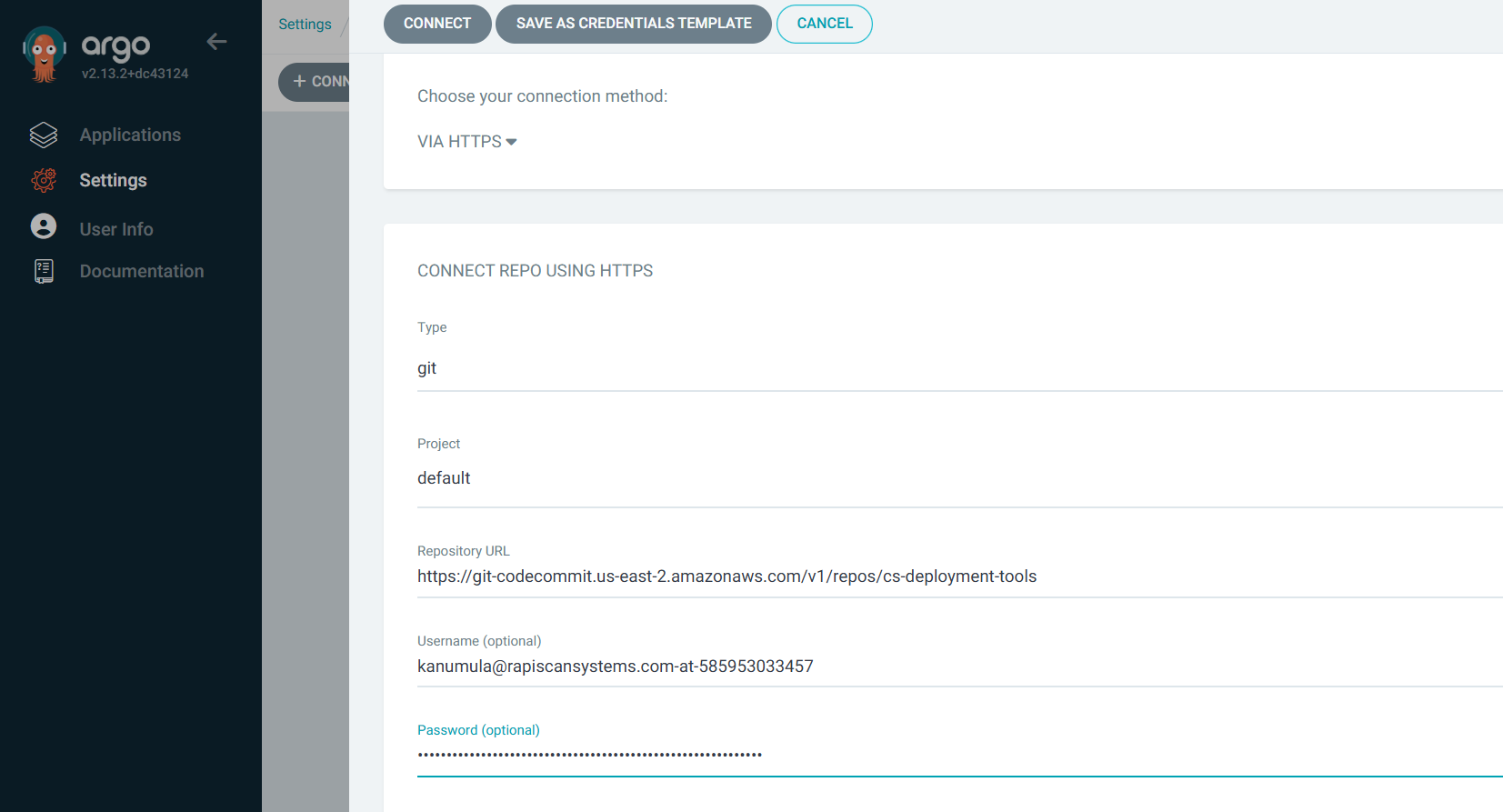


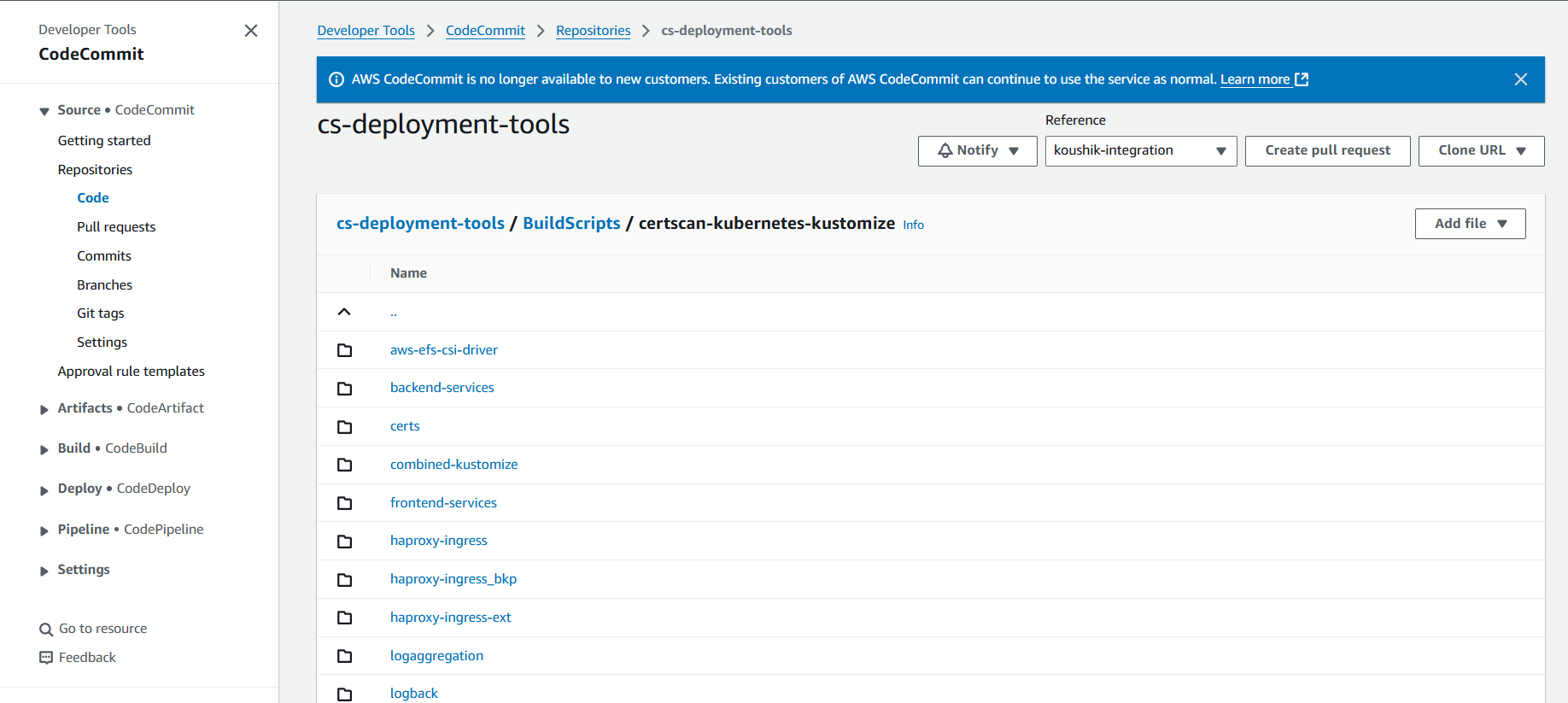
8) To authenticate a private repository, go to **settings -> Repositories -> Connect Repo**



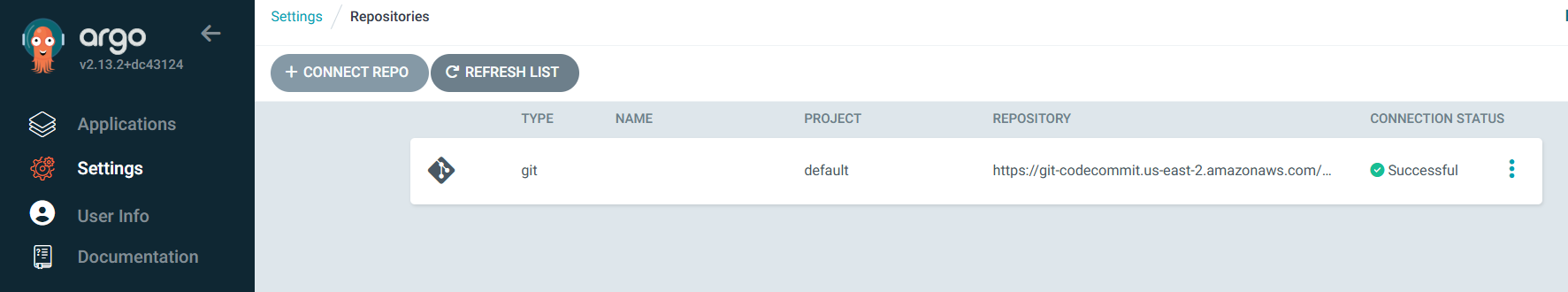


9) Choose the **HTTPS** method and enter the repo URL and add your **AWS CodeCommit credentials** in place of username and password.

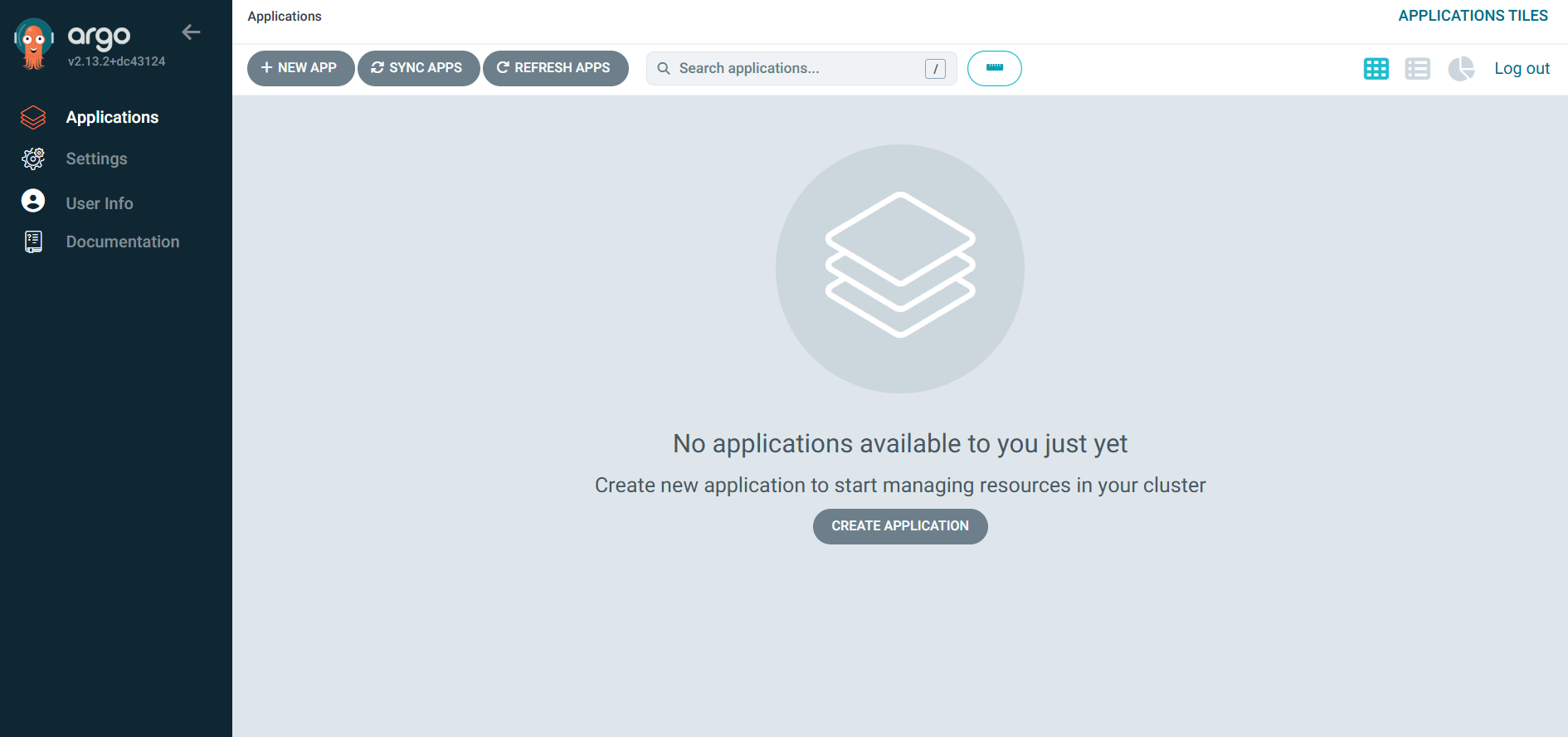




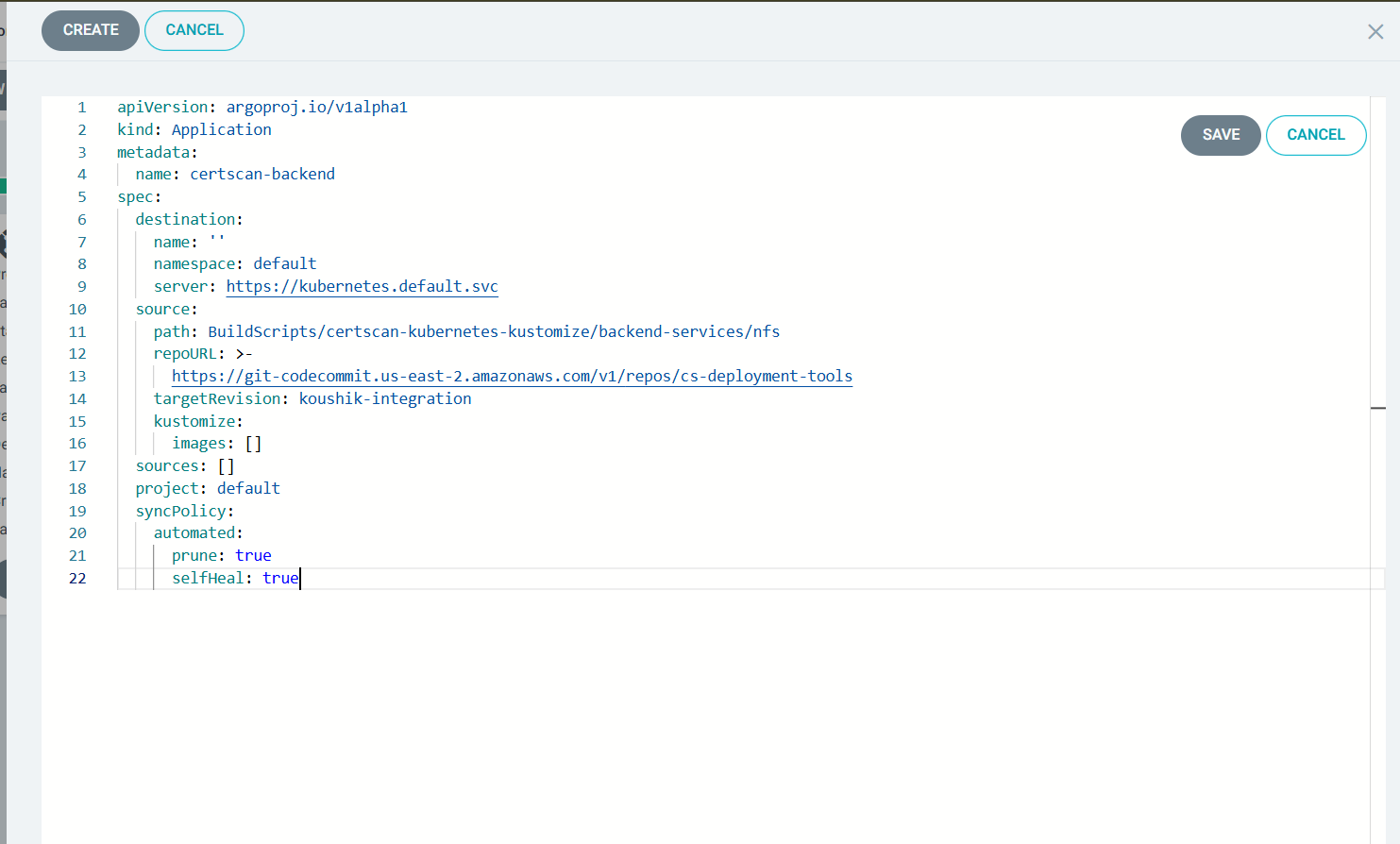
10) You can see the repo has been added successfully.



11) Now, go to the ArgoCD dashboard to add an argoCD application. Click on **“New App”.**



12) Now configure an application. Enter the configuration provided below. Create two applications. One for backend applications and one for frontend applications.



Add this configuration in this editor:  
For Backed Application  
  
**apiVersion:** [**argoproj.io/v1alpha1**](http://argoproj.io/v1alpha1)

**kind: Application**

**metadata:**

**name: certscan-backend**

**spec:**

**destination:**

**name: ''**

**namespace: default**

**server: https://kubernetes.default.svc**

**source:**

**path: BuildScripts/certscan-kubernetes-kustomize/backend-services/nfs**

**repoURL: >-**

**https://git-codecommit.us-east-2.amazonaws.com/v1/repos/cs-deployment-tools**

**targetRevision: koushik-integration**

**kustomize:**

**images: []**

**sources: []**

**project: default**

**syncPolicy:**

**automated:**

**prune: true**

**selfHeal: true**And for Frontend Application use this:  
  
**apiVersion: argoproj.io/v1alpha1**

**kind: Application**

**metadata:**

**name: certscan-frontend**

**spec:**

**destination:**

**name: ''**

**namespace: default**

**server: https://kubernetes.default.svc**

**source:**

**path: BuildScripts/certscan-kubernetes-kustomize/frontend-services/nfs**

**repoURL: >-**

**https://git-codecommit.us-east-2.amazonaws.com/v1/repos/cs-deployment-tools**

**targetRevision: koushik-integration**

**kustomize:**

**images: []**

**sources: []**

**project: default**

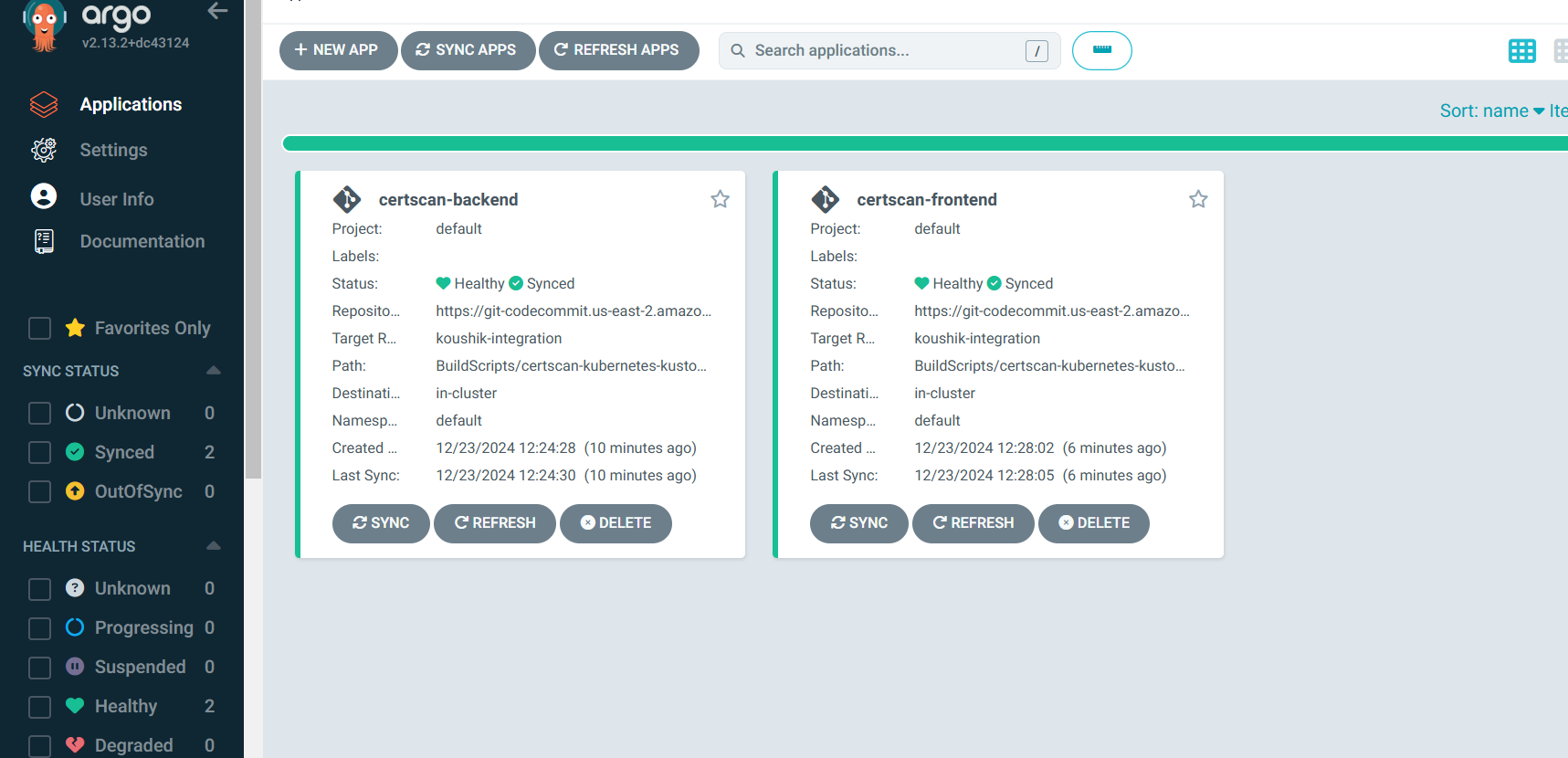
**syncPolicy:**

**automated:**

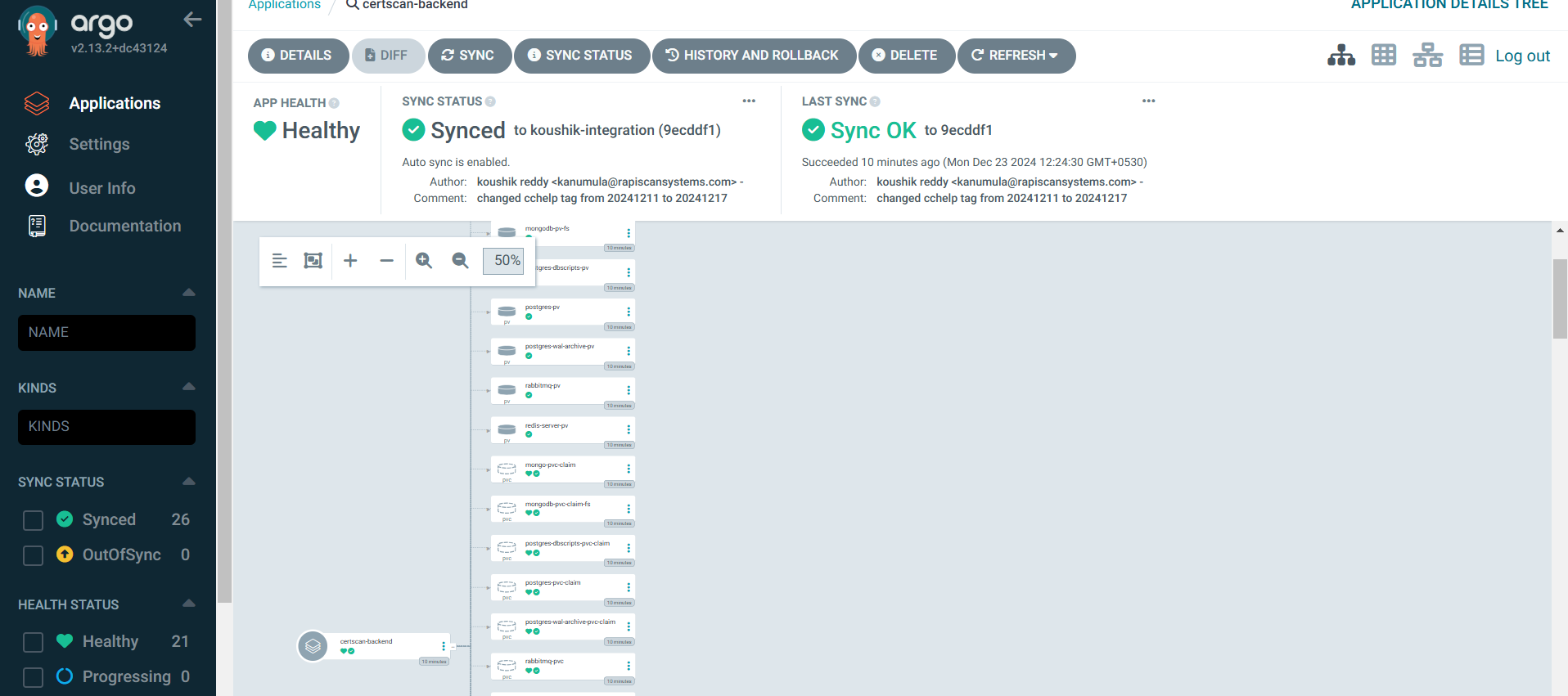
**prune: true**

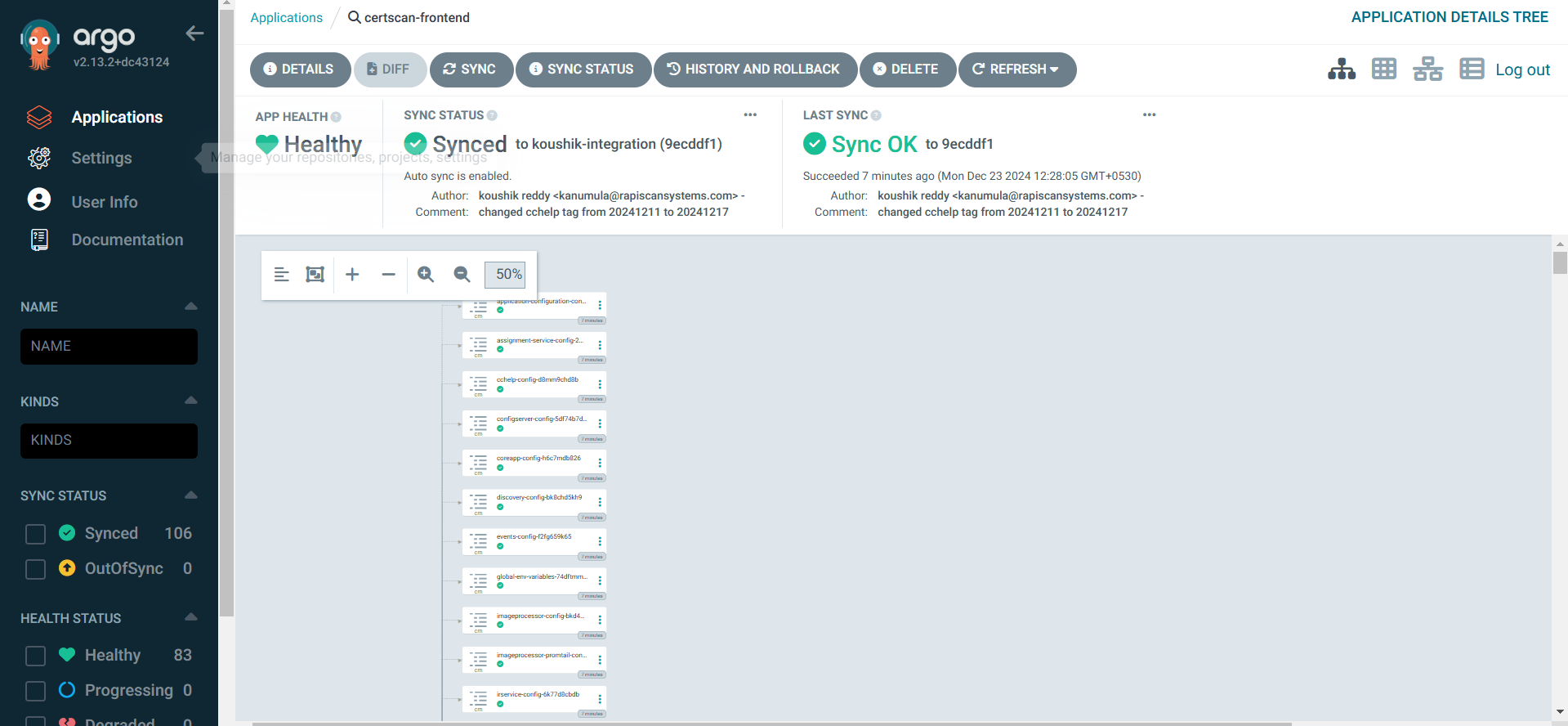
**selfHeal: true**

13) You can see that apps have been created and they are synced properly.

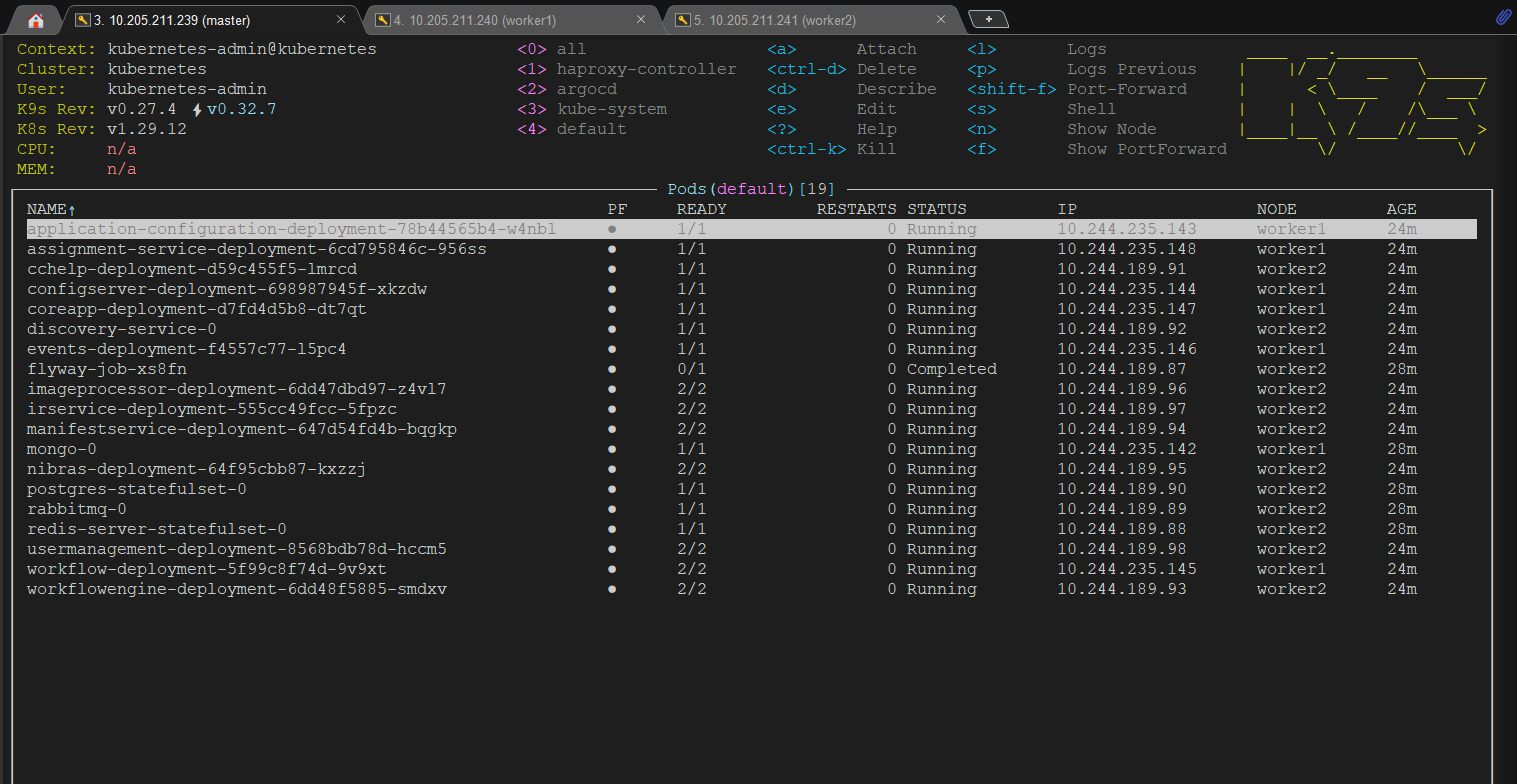


14) You can check all the installed services by that application.

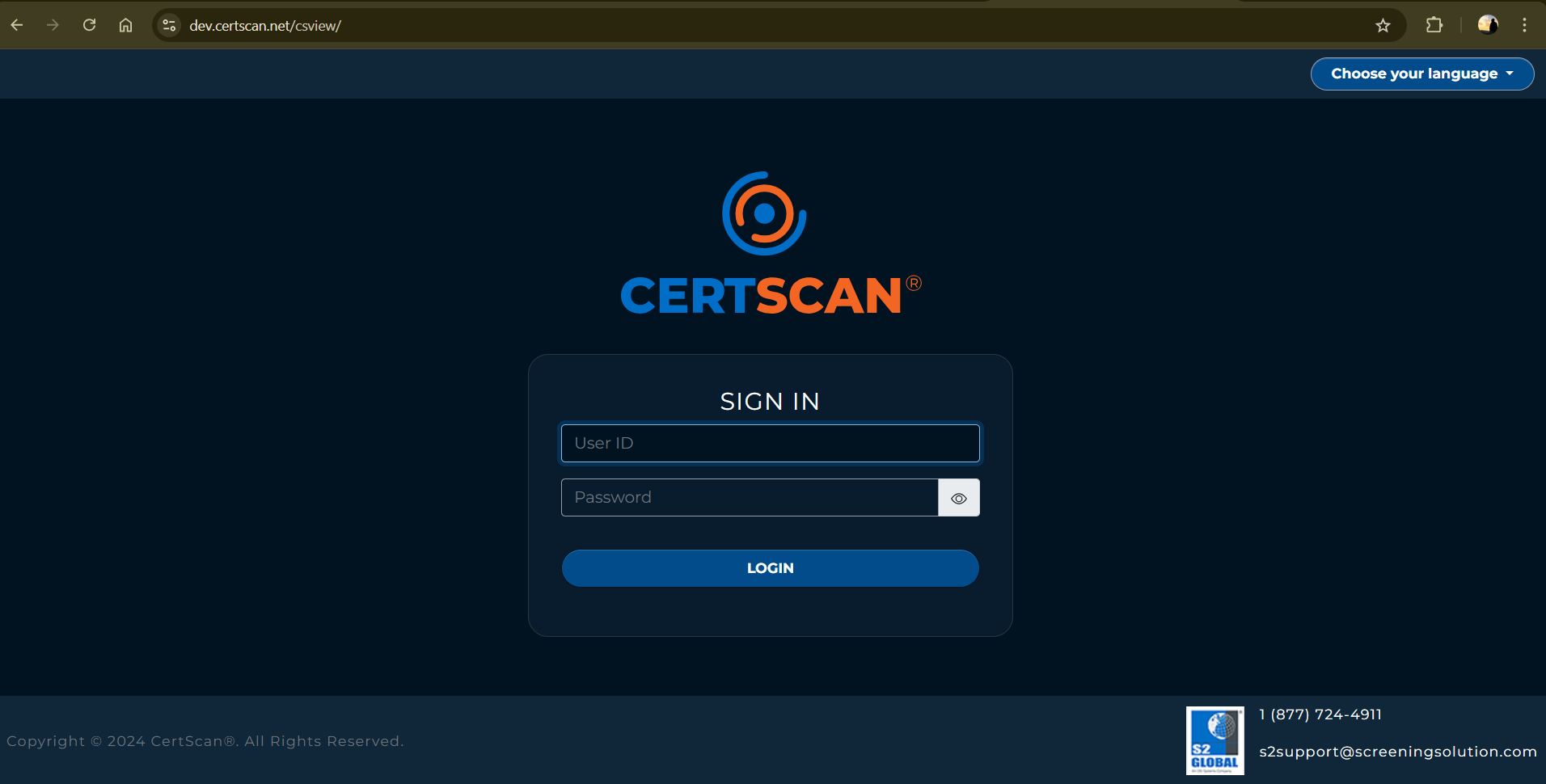




15) You can see the application deployed in the kubernetes cluster



16) Make sure to run haproxy services in the cluster to access the application using domain name.



17) You can access the application.