DevOps

GitOps: Deploying Tetris on EKS Using ArgoCD

mrcloudbook.com · 8 January 2024

Inspired From Cloud Champ's Tutorial

Github: https://github.com/N4si/tetris-game.git

Welcome to an exhilarating journey where we explore the cutting-edge world of GitOps, all while preparing to indulge in one of the most iconic and addictive games of all time: Tetris! In this blog, we'll unravel the mysteries of GitOps, witness its

transformational power, and set the stage for a thrilling gaming experience—all within a Kubernetes cluster deployed on Amazon's Elastic Kubernetes Service (EKS).

Buckle up, as we embark on a fascinating ride that will not only introduce you to GitOps but also allow you to enjoy the classic Tetris game on a Kubernetes platform. Let's get started and see how DevOps and gaming collide in this exciting adventure!

Contents [hide]

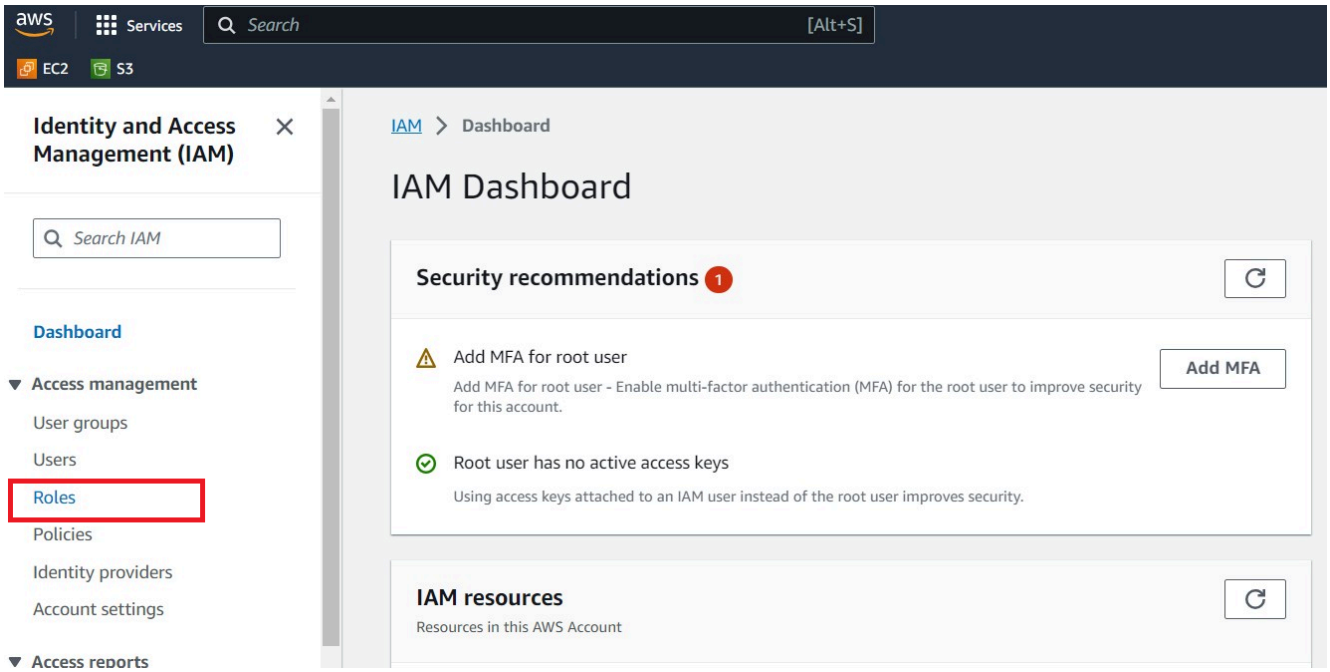
STEP 1: Create IAM Roles

Let's First start creating two IAM roles one is for Cluster and another is for Nodegroup

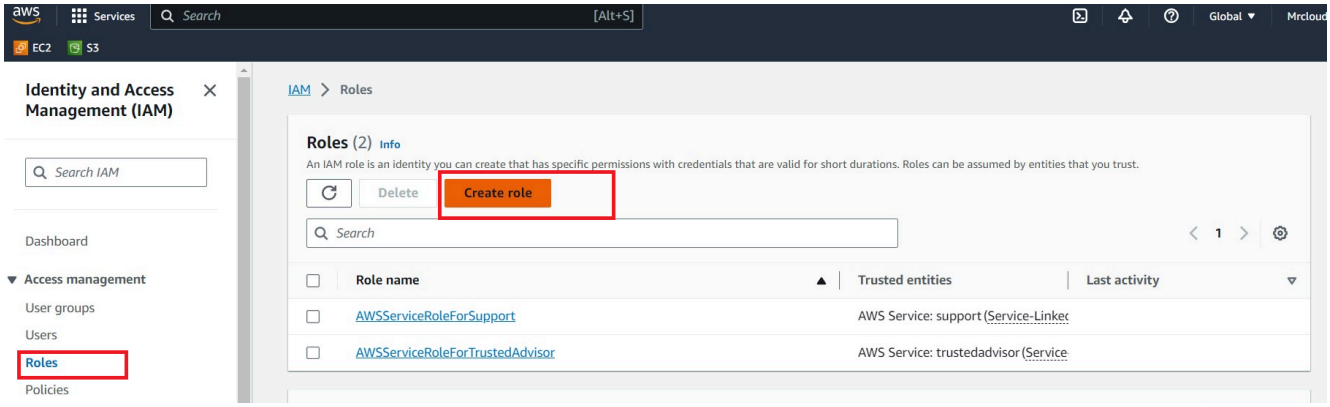
Go to Aws console and search for IAM

You will be redirected to the IAM dashboard

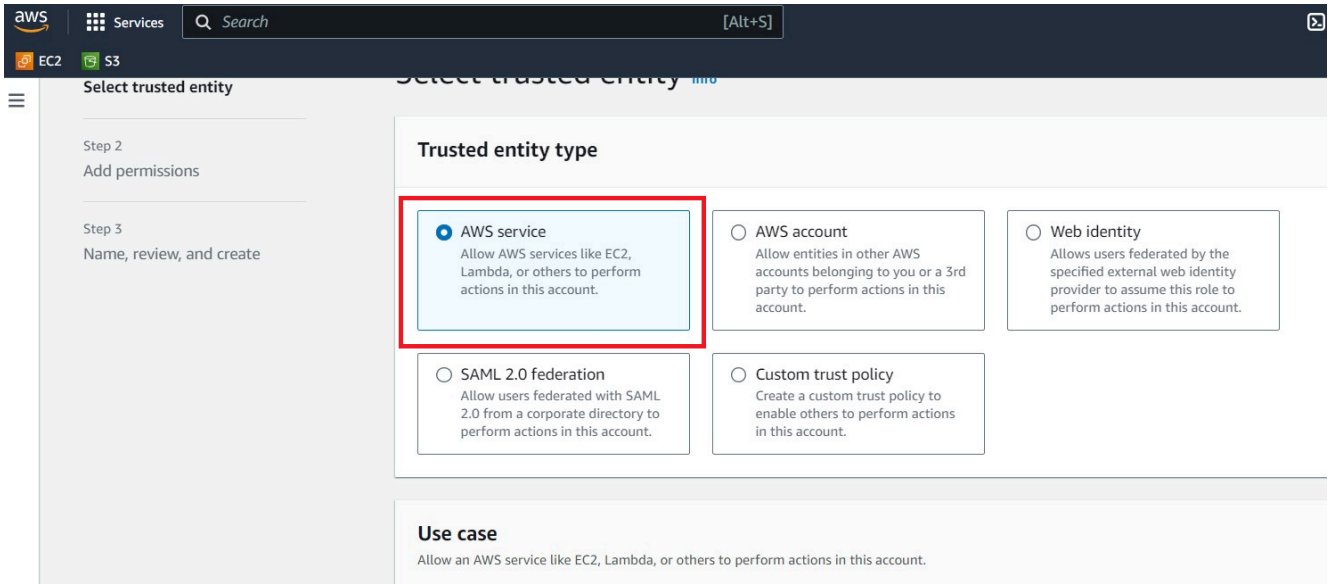
Click "Roles"



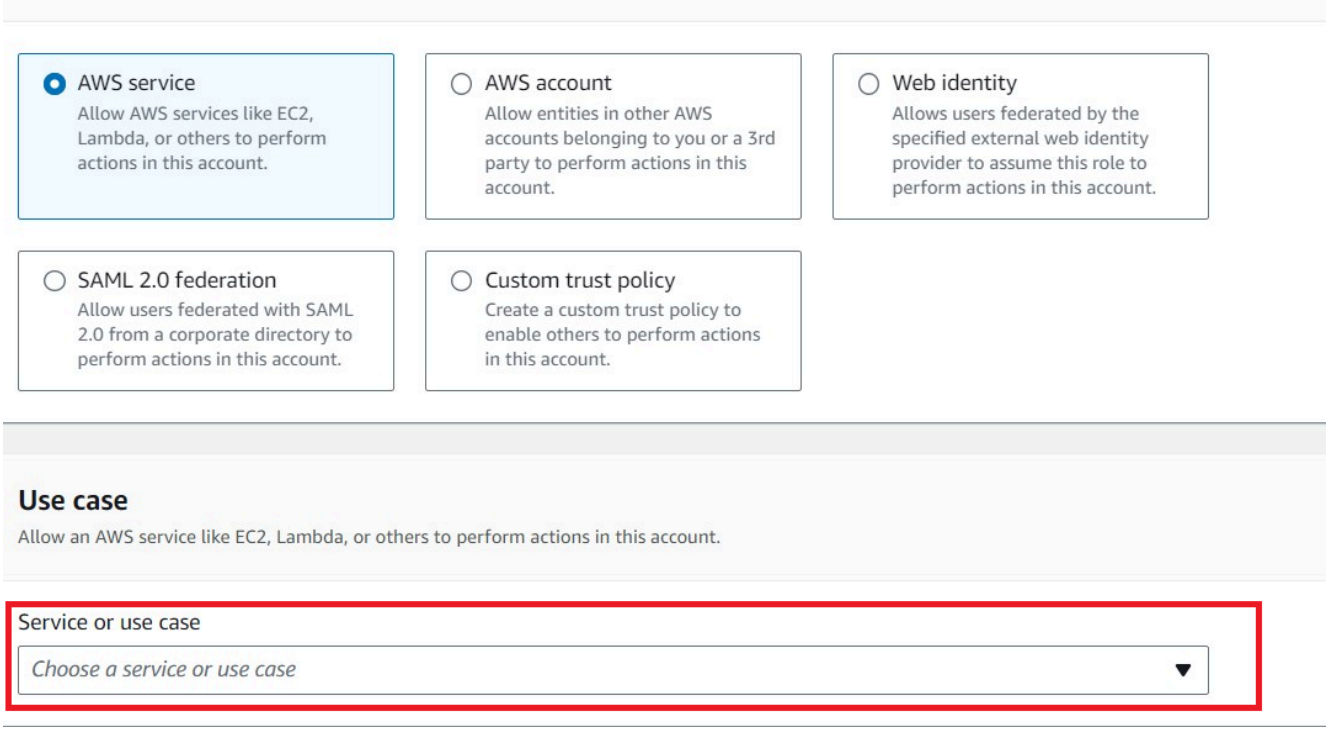
Click “Create role”



Click “Allow AWS services like EC2, Lambda, or others to perform actions in this account.”

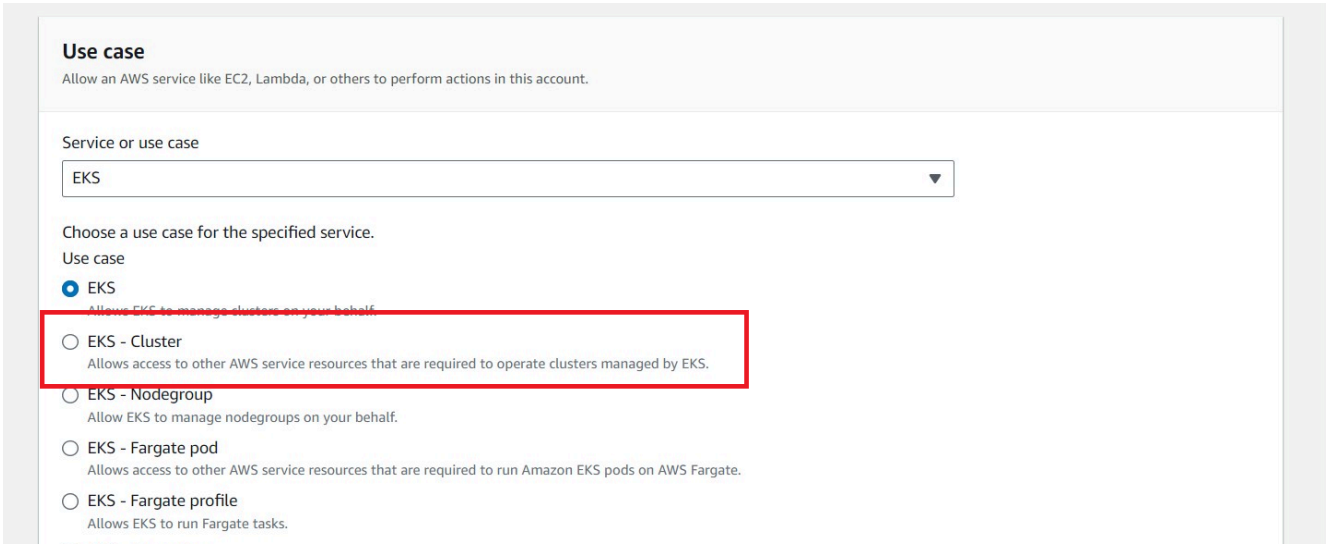


Click “Choose a service or use case”

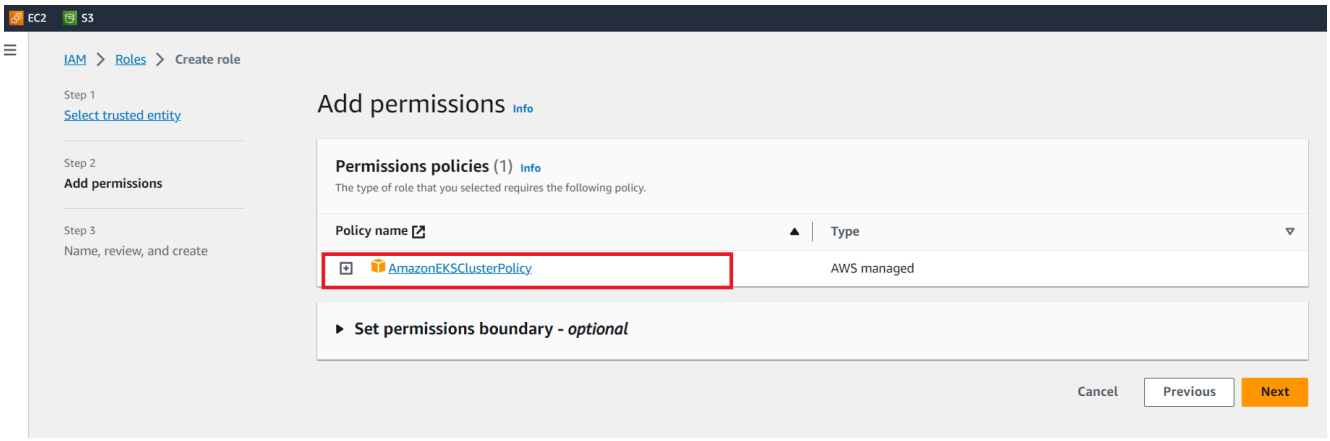


Type “EKS”

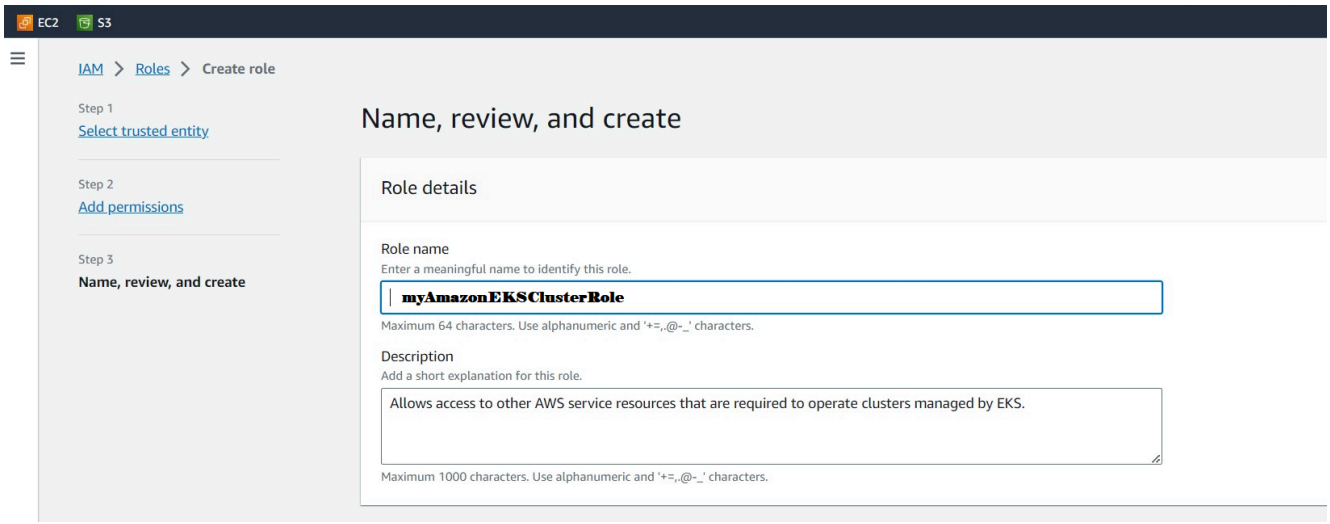
Click this radio button with EKS-Cluster



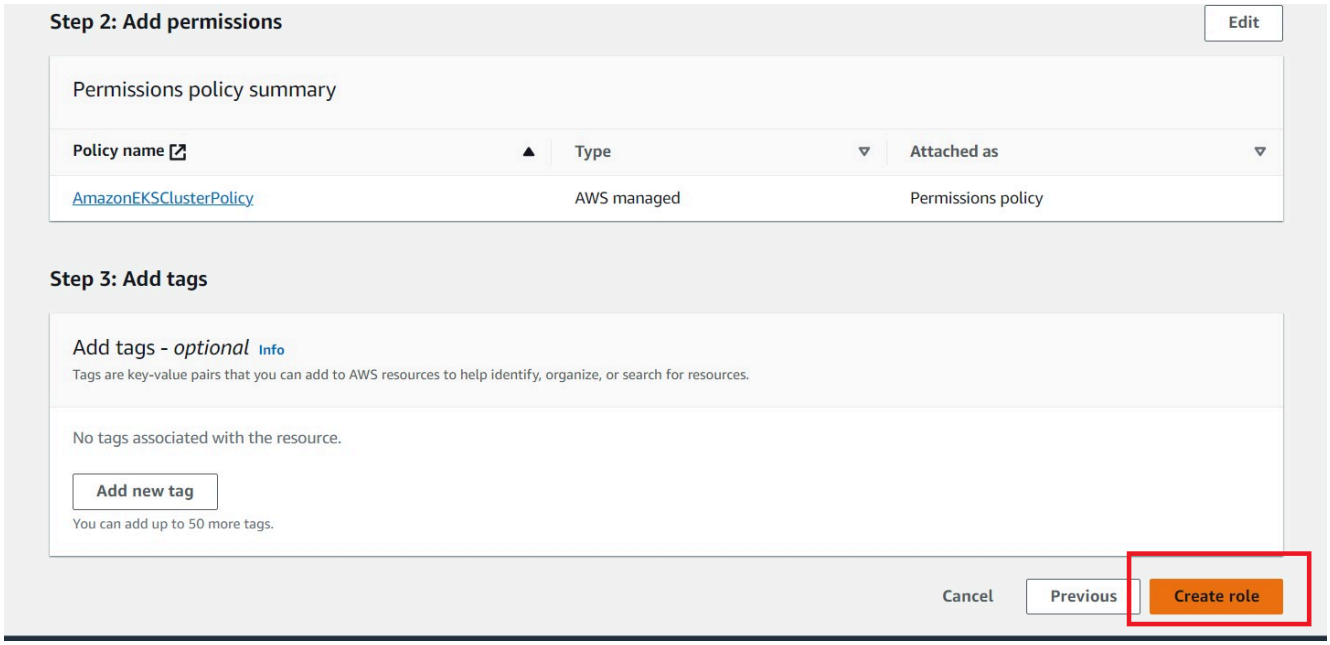
Click “Next” and you will directly redirect to policy and click Next (we have only one policy for it and it selects by default for EKS) that is `AmazonEKSClusterPolicy`



Click the “Role name” field and provide the name (`myAmazonEKSClusterRole`)



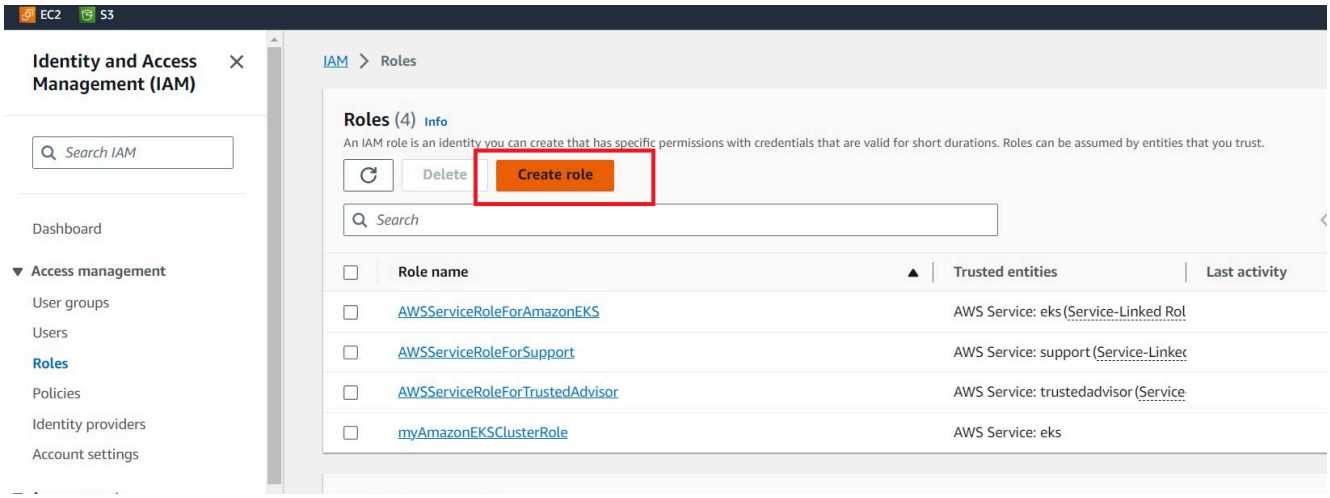
Click “Create role”



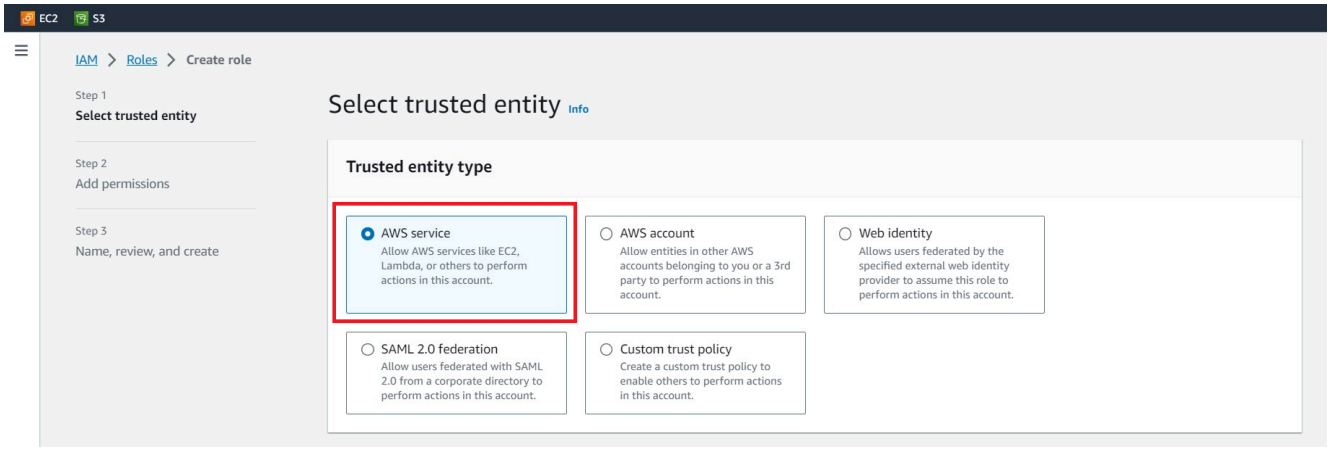
A cluster role is created.

Now Create a Role for NodeGroup

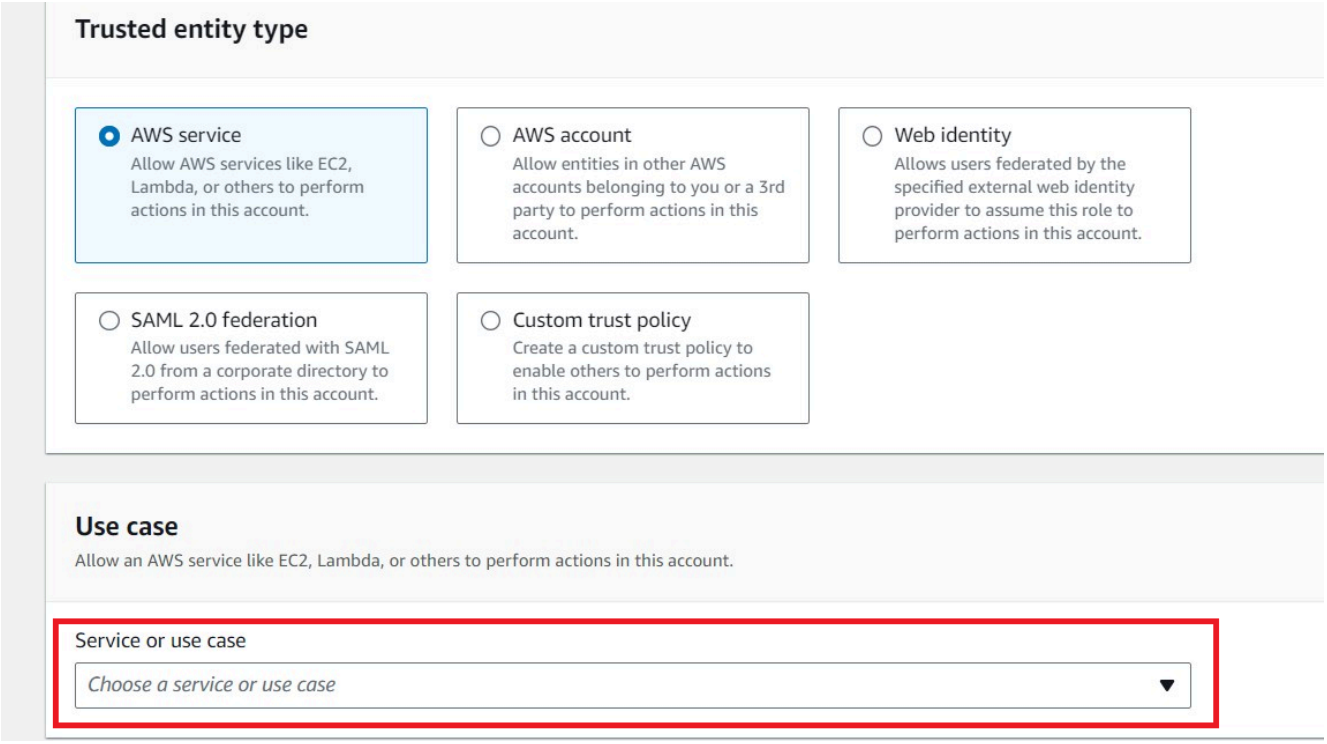
Click “Create role”



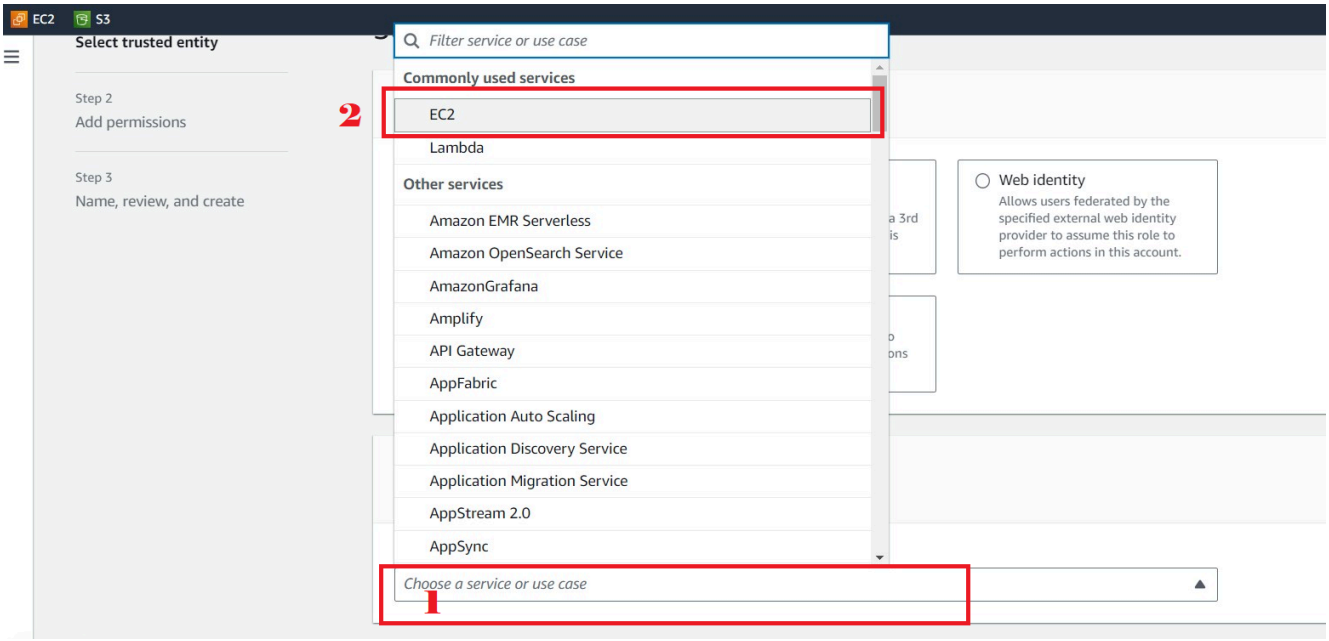
Click “Allow AWS services like EC2, Lambda, or others to perform actions in this account.”



Click “Choose a service or use case”

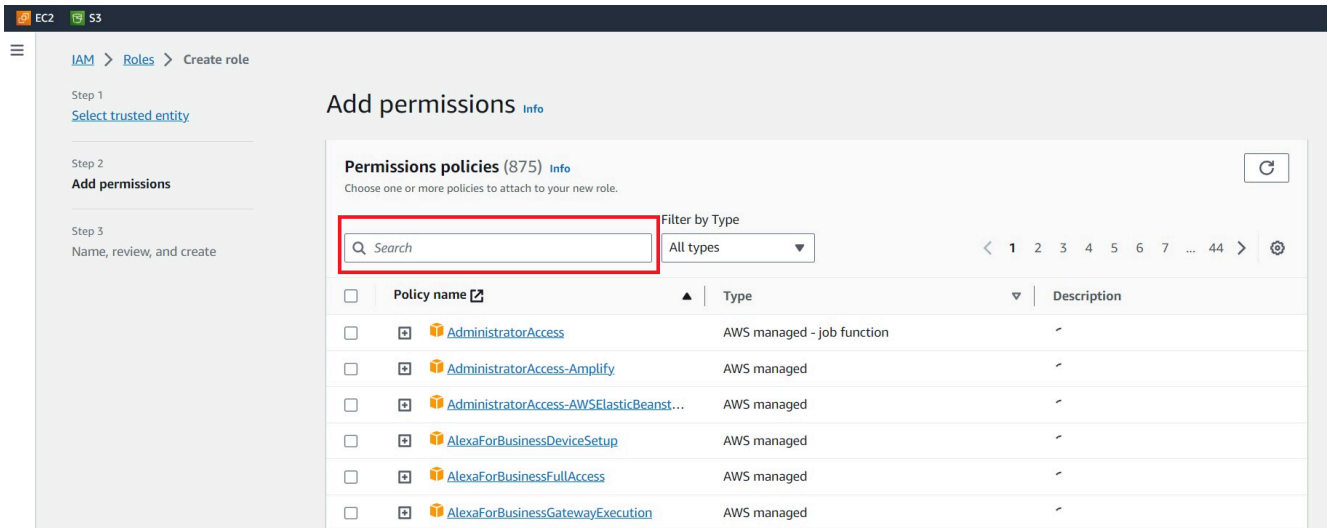


Click “EC2”



Click “Next”

Click the “Search” field.



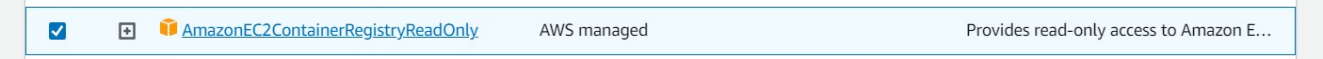
Search these Policy Names and make it check (I already have these in it)

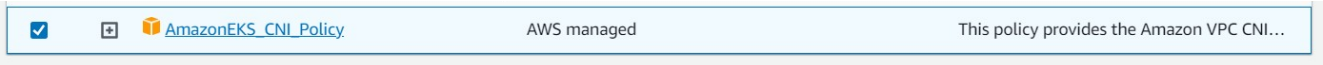
`AmazonEC2ContainerRegistryReadOnly`

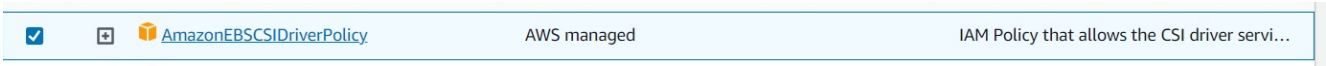
`AmazonEKS_CNI_Policy`

`AmazonEBSCSIDriverPolicy`

`AmazonEKSWorkerNodePolicy`





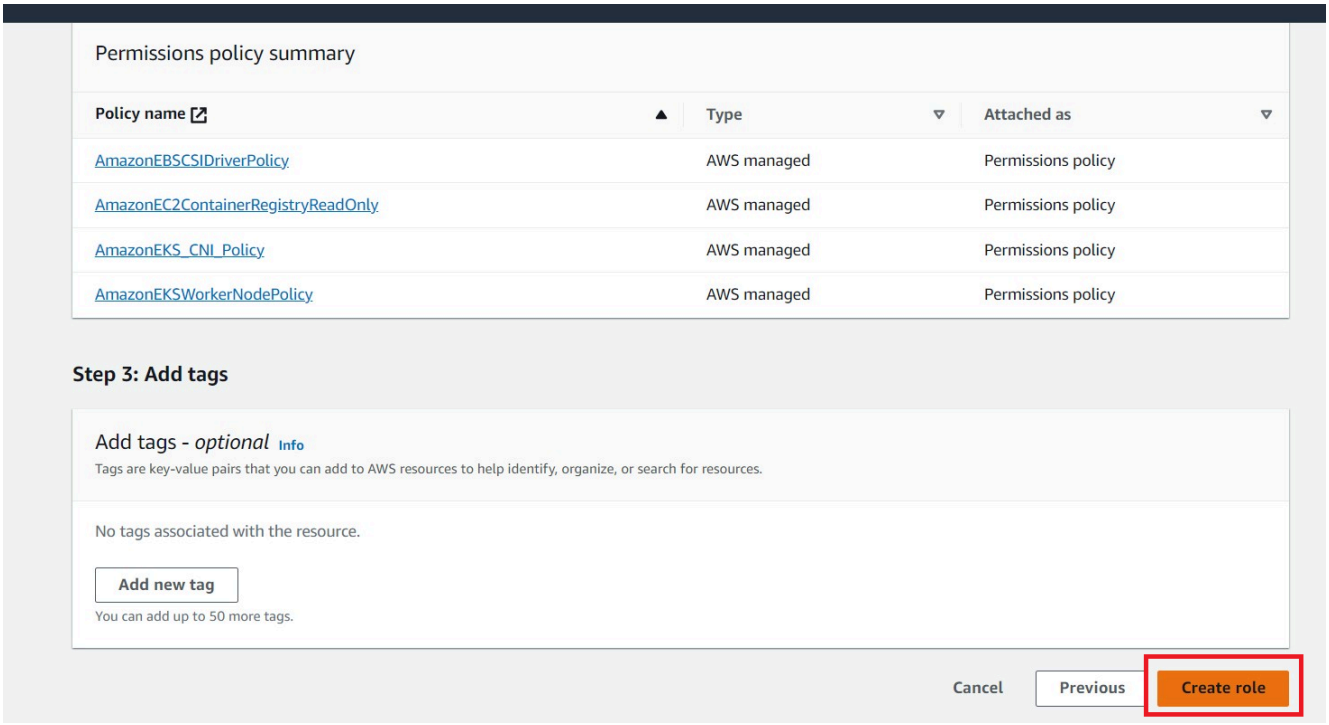


Click “Next”

Click the “Role name” field.

Add Role name as `myAmazonNodeGroupPolicy`

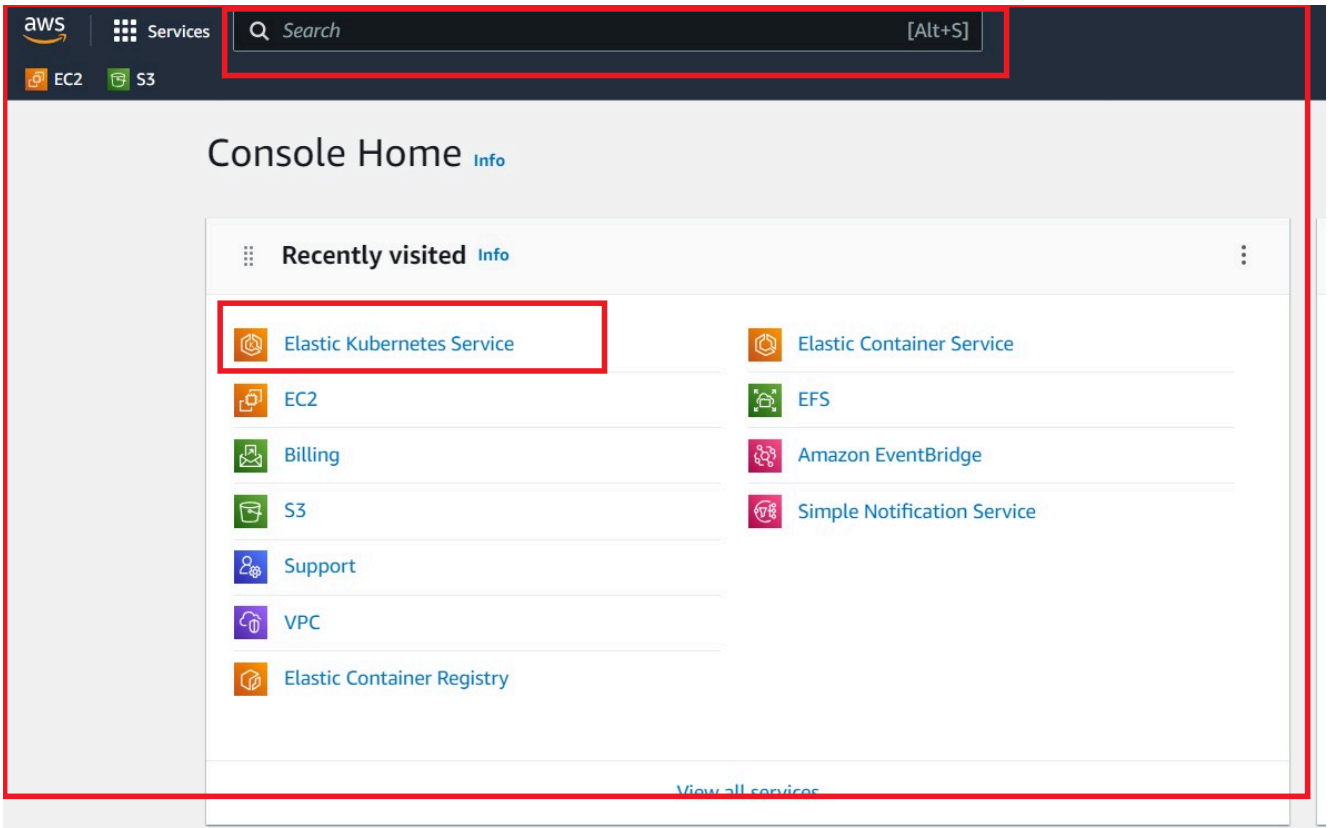
Click “Create role”



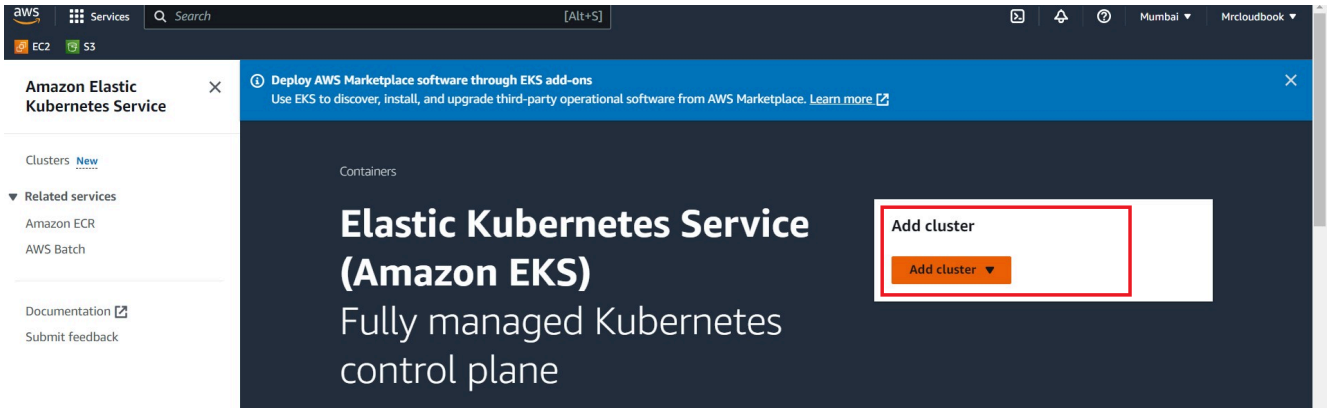
NodeGroup Role is created.

Step 2: Create EKS Cluster

Click the “Search” field and search For EKS or select directly Elastic Kubernetes Service on the Recently visited tab



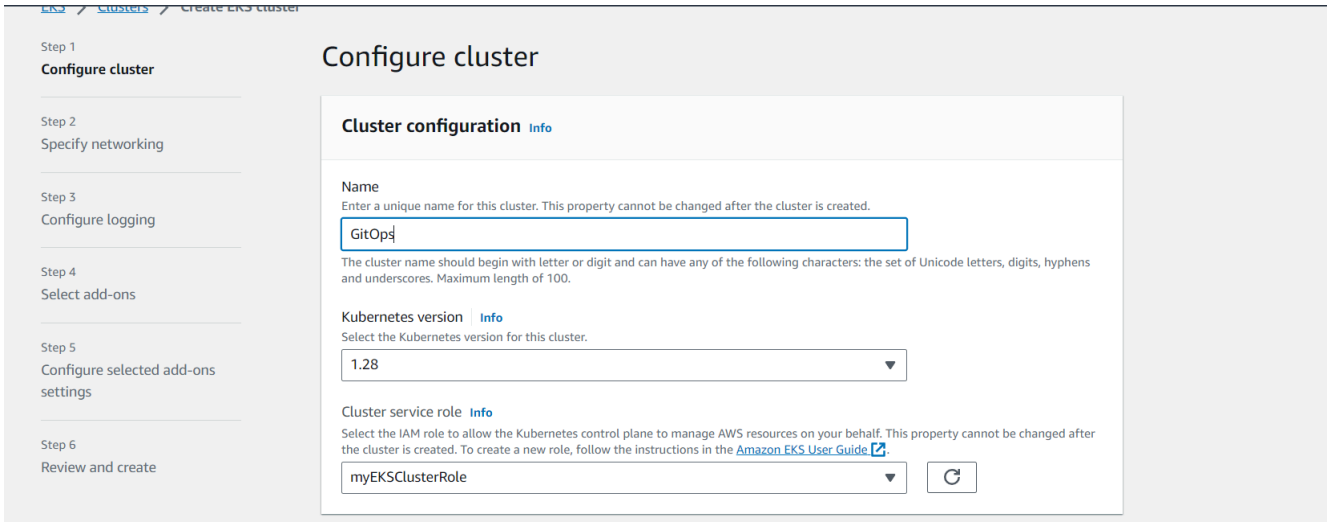
Click “Add cluster”



Click “Create”

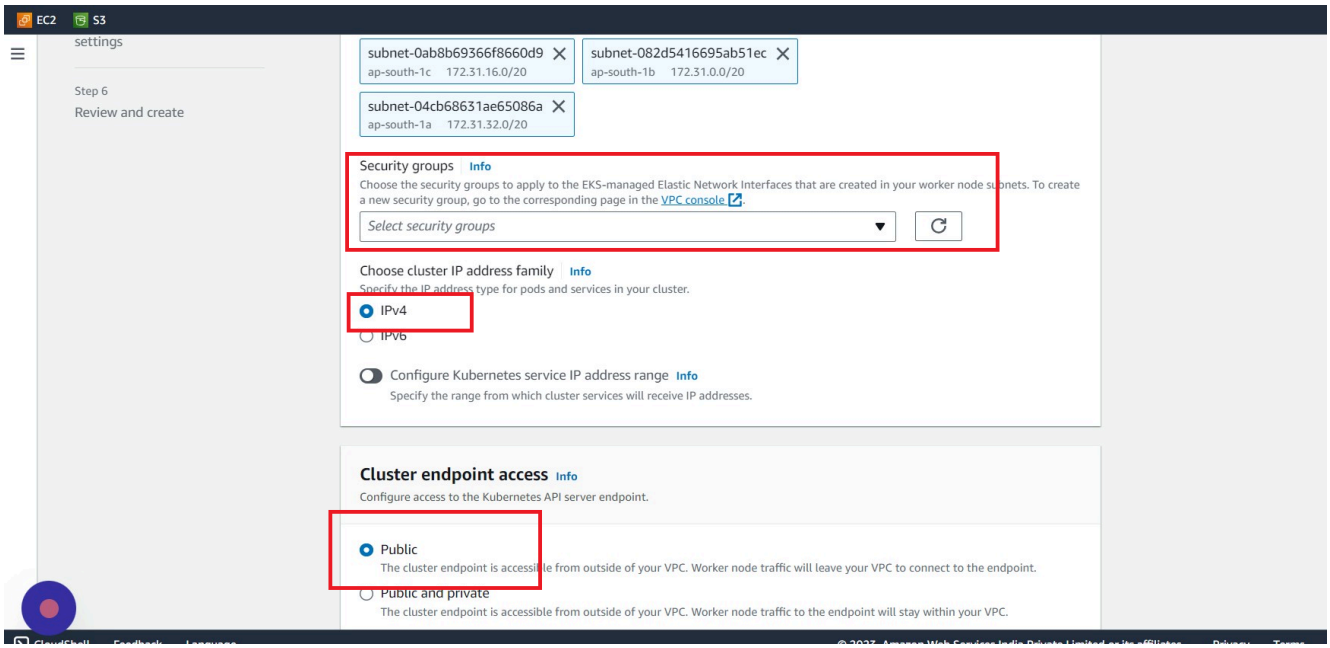
Click the “Name” field and enter a unique name for the cluster that is anything you want. For example, I used Cloud and version 1.28

Click “myAmazonEKSClusterRole” which is created in step 1.

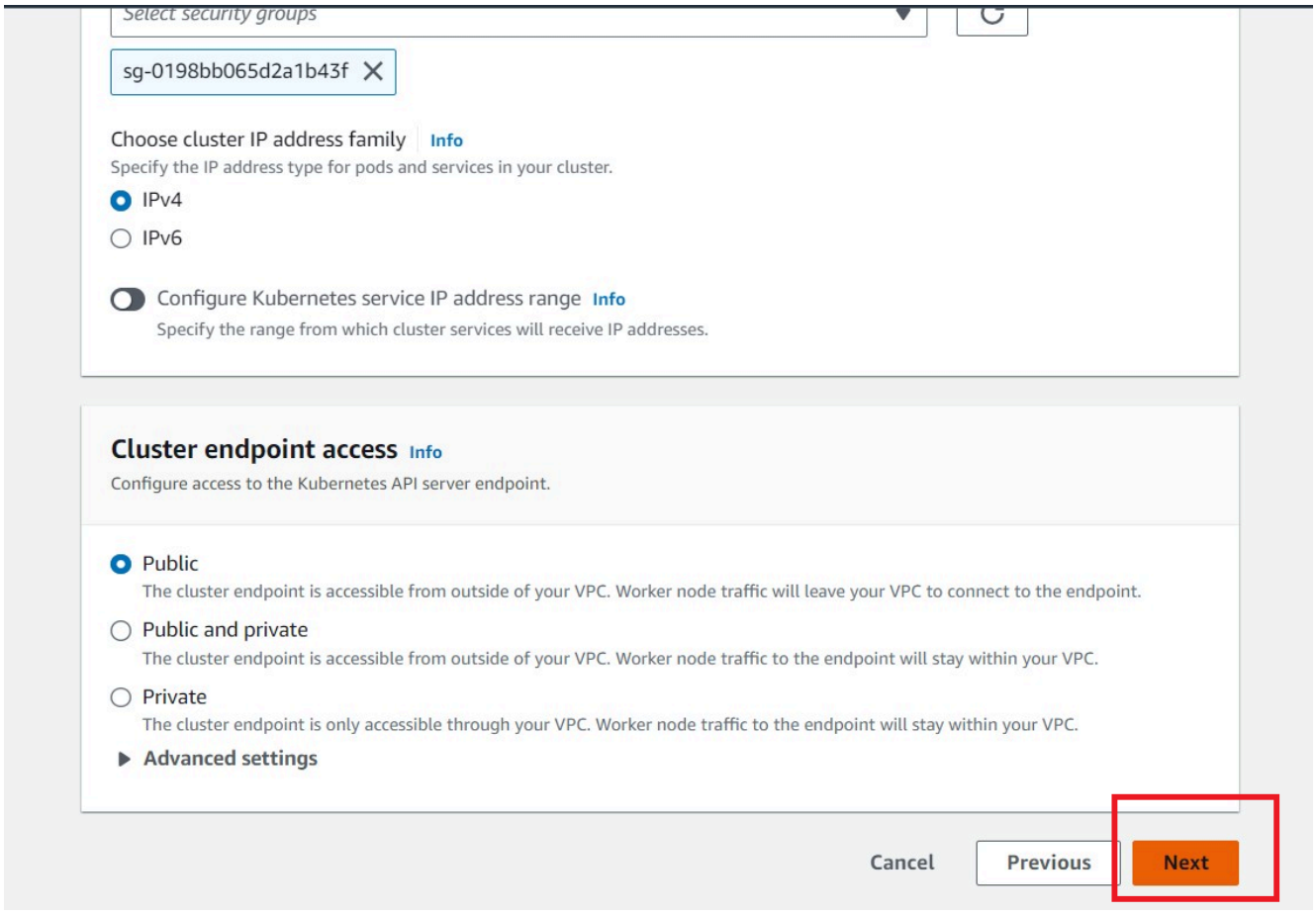


Click “Next”

Click “Select security groups” and Use the existing security group or create a new security Group



Click “Next”



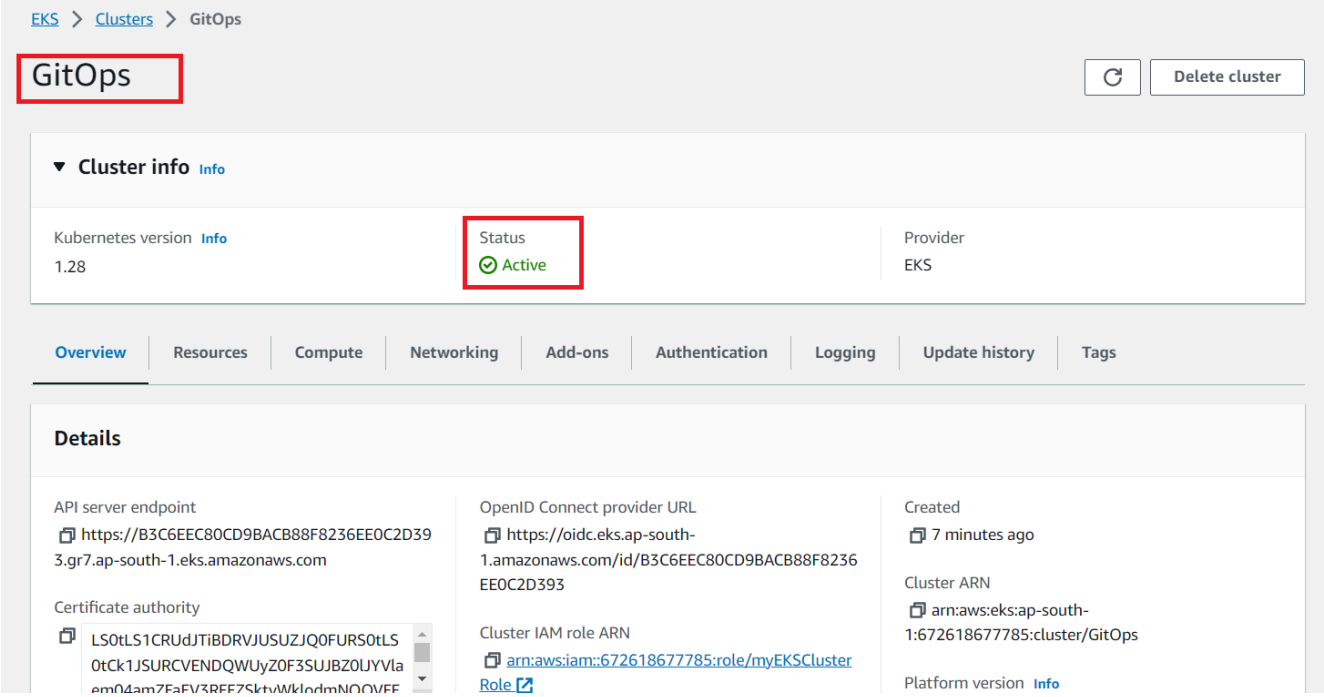
Click “Next”

No changes Click “Next” (Default no need to change anything)

No changes Click “Next” (Default no need to change anything)

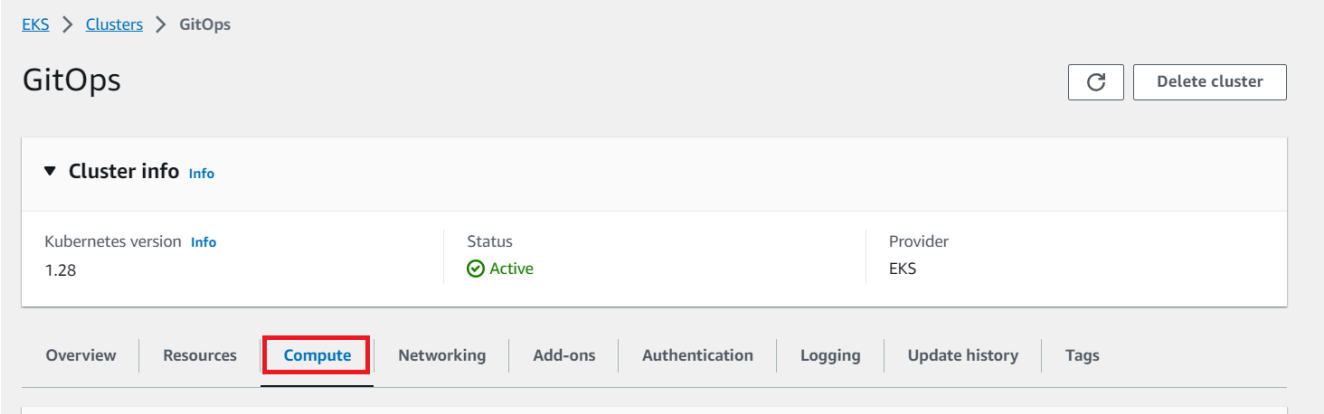
Click “Create”

It takes 15 minutes to create.

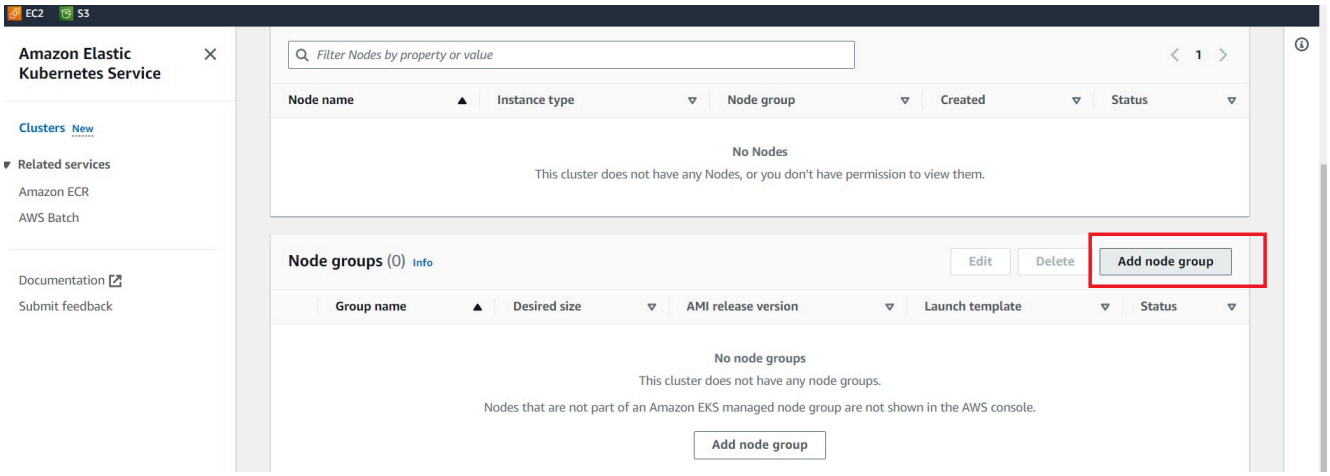


Once your Cluster up to active status

Click “Compute”



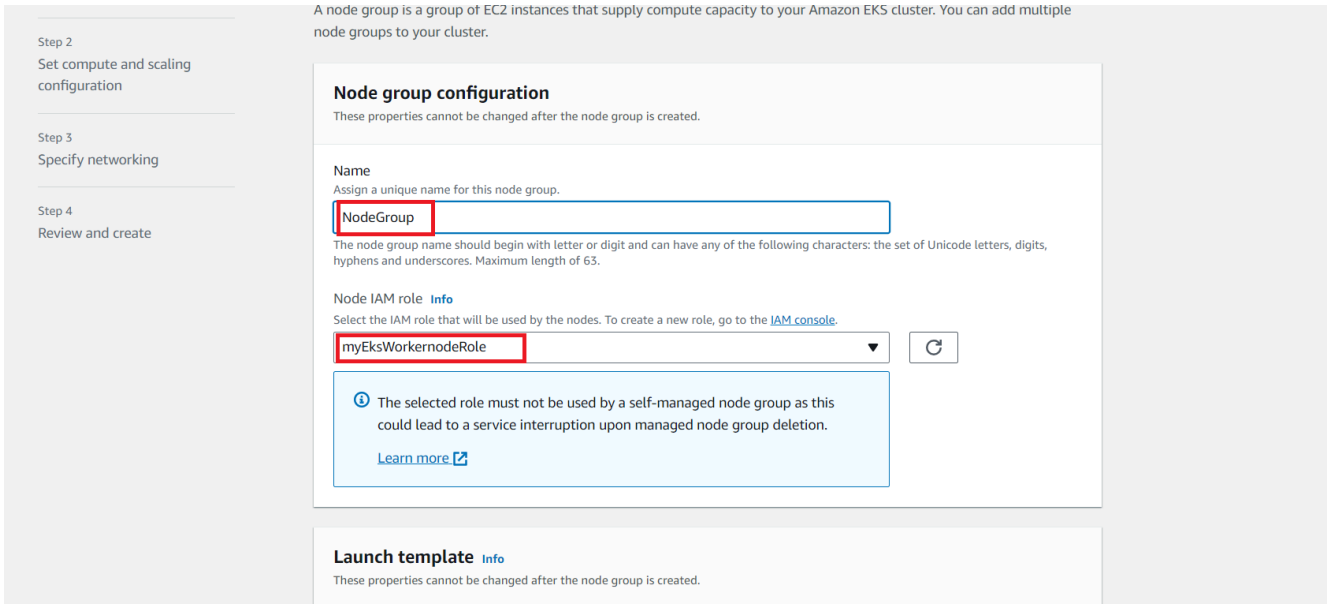
Click on “Add node group”



Click the “Name” field.

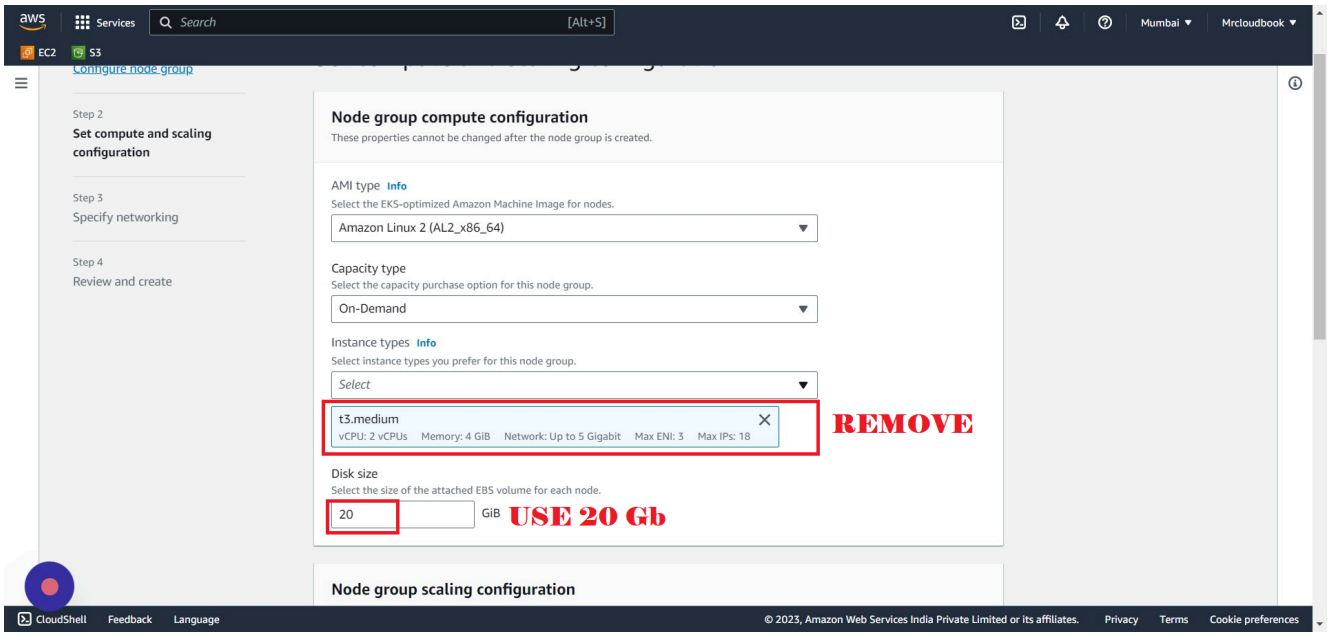
Write any Name you want (`NodeGroup`)

Select the Role that was created for the node Group in Step 1

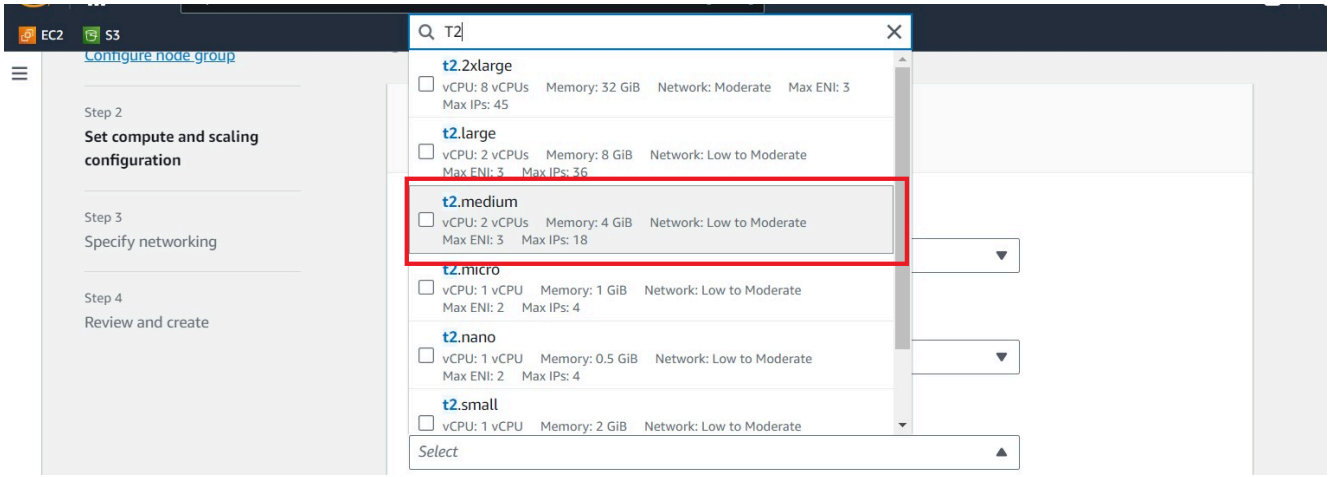


Click “Next”

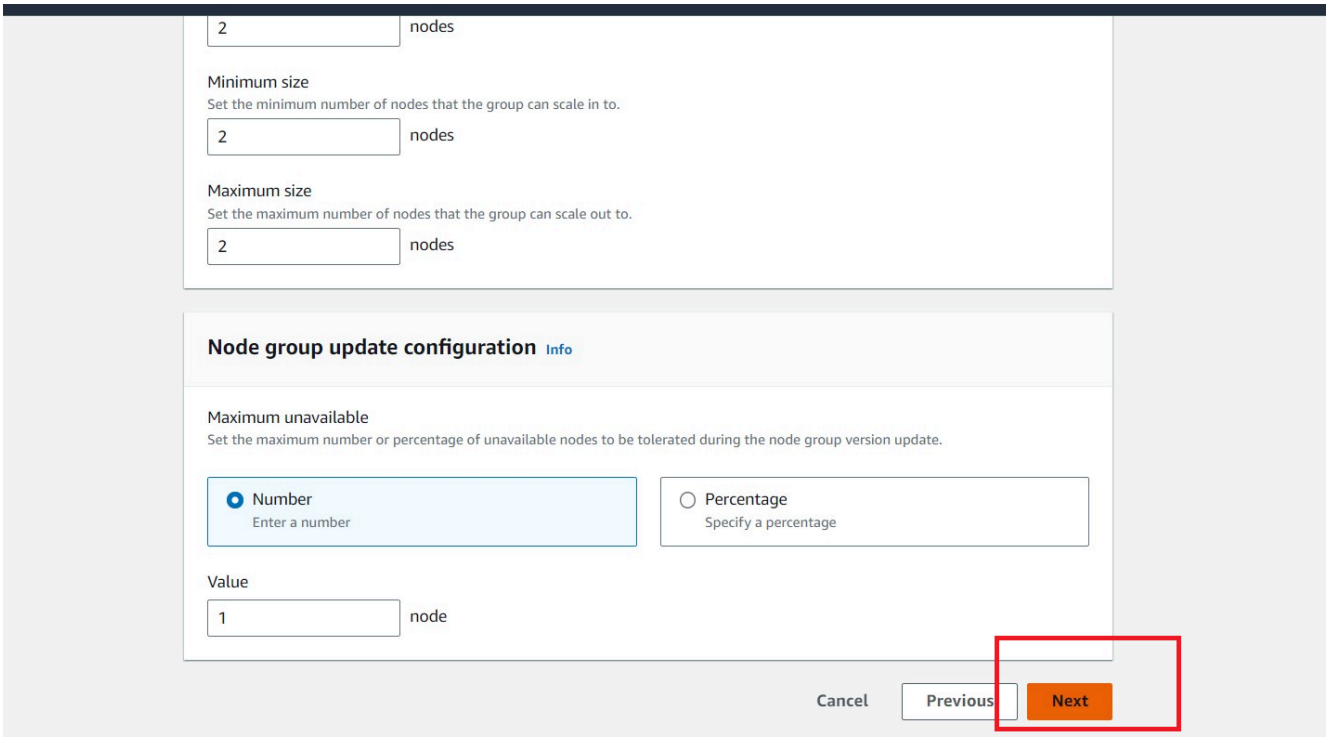
On the next page remove t3.medium and add t2.medium as instance type.



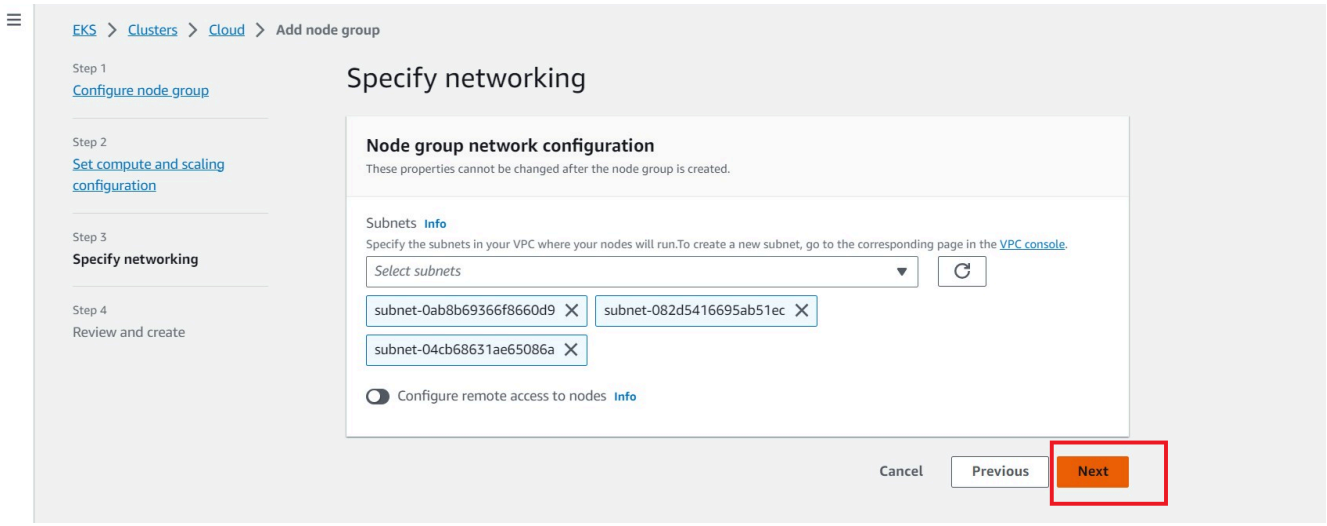
Select t2.medium



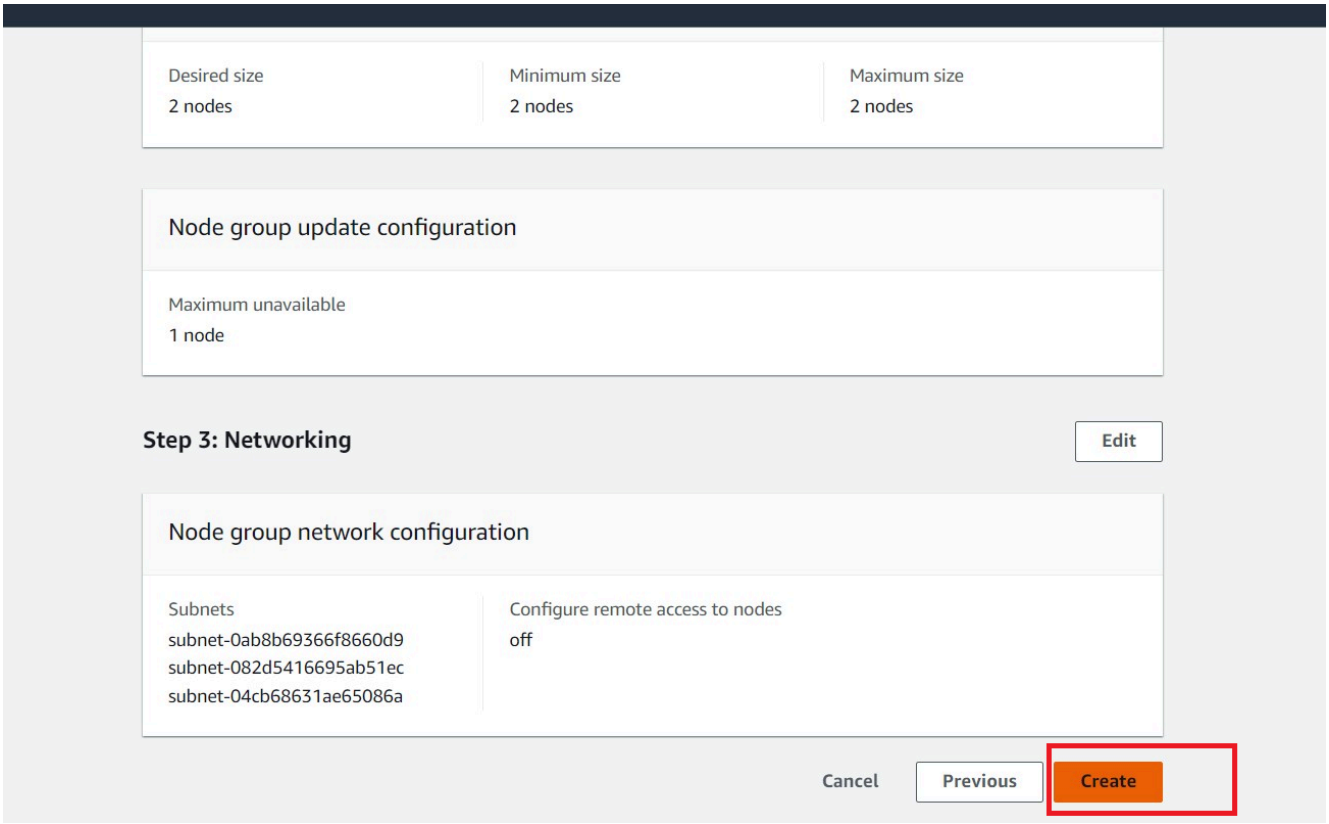
Click “Next”



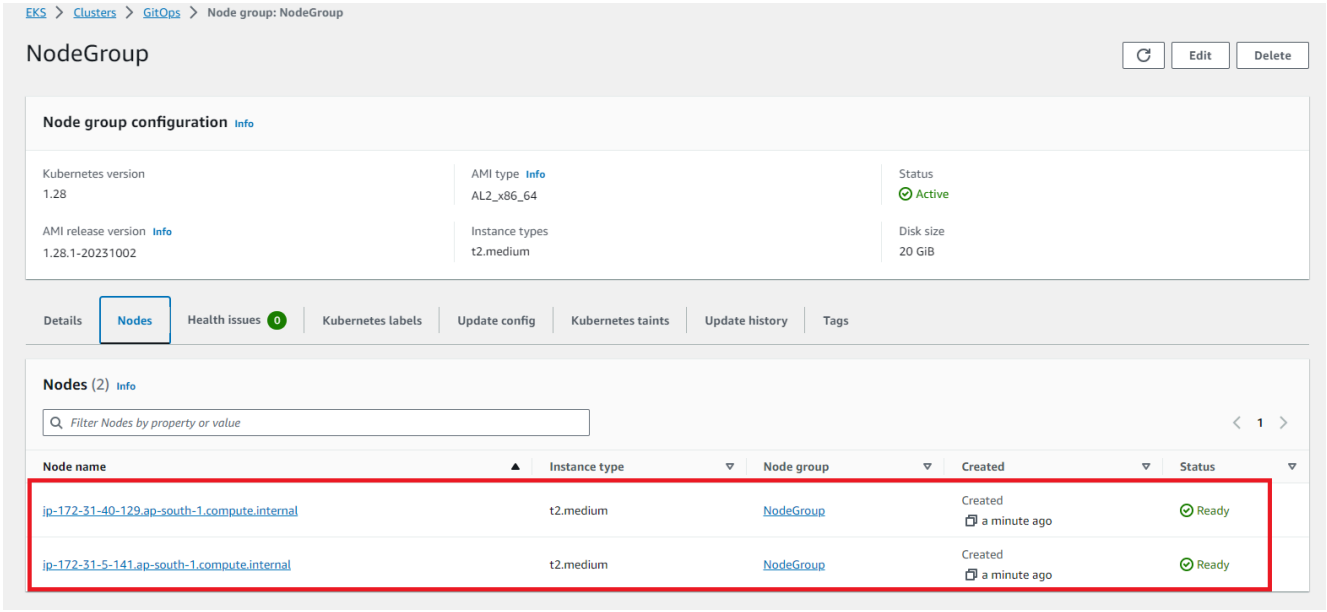
Click “Next”



Click “Create”



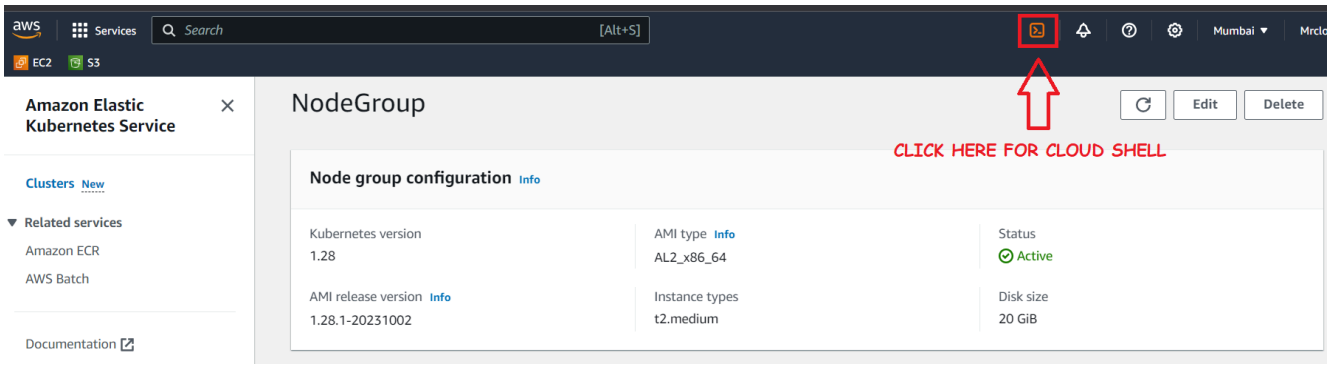
Node Groups will take some time to create.



Worker nodes created.

Step 3: ARGOCD SETUP

Click on the AWS cloud shell icon on the top right

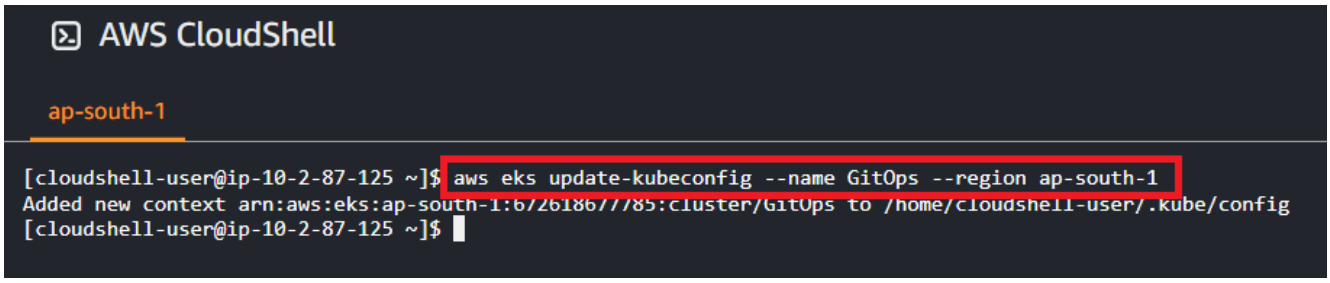


click on connect

Better to open in a new tab

First set context by providing the following command

```
aws eks update-kubeconfig --name EKS_CLUSTER_NAME --region CLUSTER_REGI(
#example
aws eks update-kubeconfig --name GitOps --region ap-south-1
```



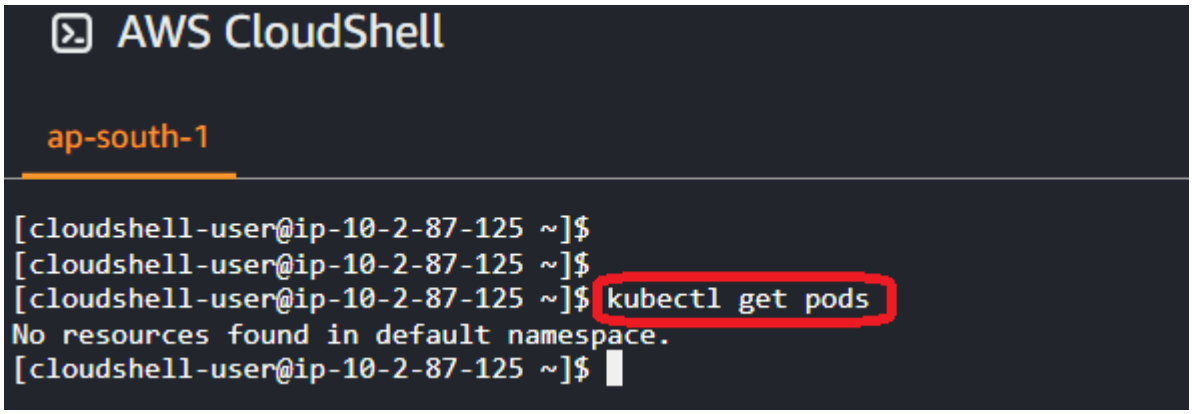
Check for Nodes

```
kubectl get nodes
```

```
[cloudshell-user@ip-10-2-87-125 ~]$
[cloudshell-user@ip-10-2-87-125 ~]$ kubectl get nodes
NAME                                           STATUS   ROLES    AGE     VERSION
ip-172-31-40-129.ap-south-1.compute.internal   Ready    <none>   4m8s    v1.28.1-eks-43840fb
ip-172-31-5-141.ap-south-1.compute.internal    Ready    <none>   4m28s   v1.28.1-eks-43840fb
[cloudshell-user@ip-10-2-87-125 ~]$
```

Check for pods, You will get no resources found.

```
kubectl get pods
```



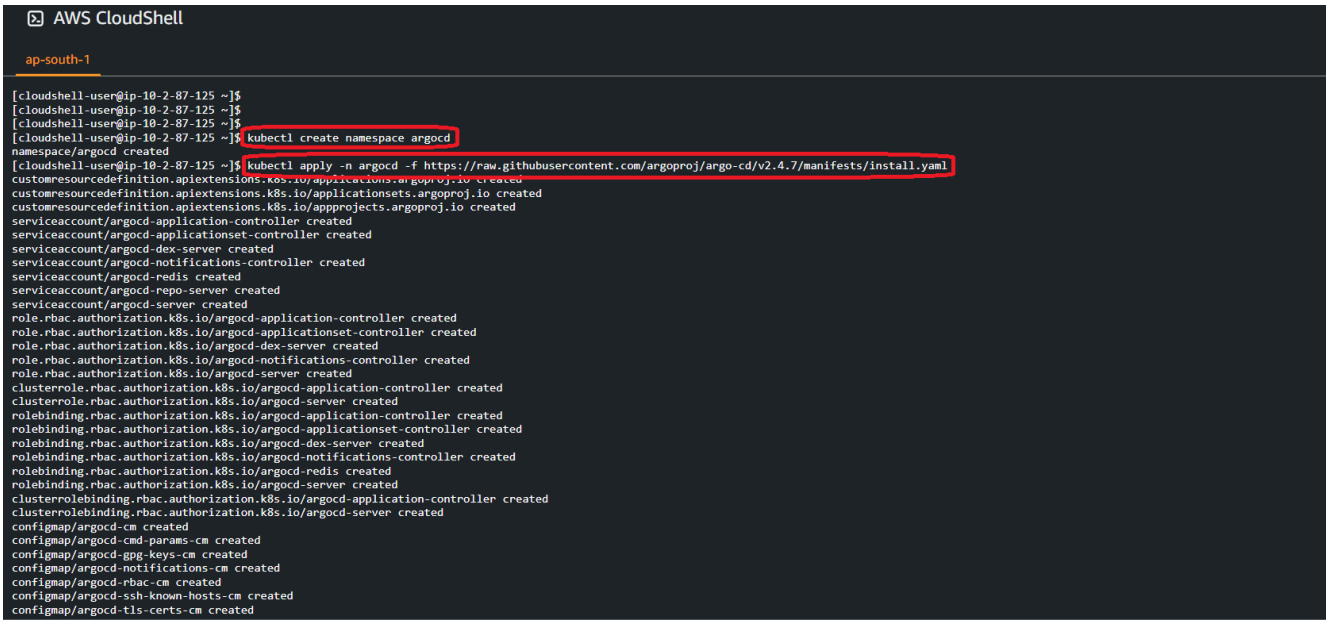
Let's install ArgoCD

ARGOCD INSTALLATION LINK

You will redirected to this page

All those components could be installed using a manifest provided by the Argo Project: use the below commands

```
'aw.githubusercontent.com/argoproj/argo-cd/v2.4.7/manifests/install.yaml
```

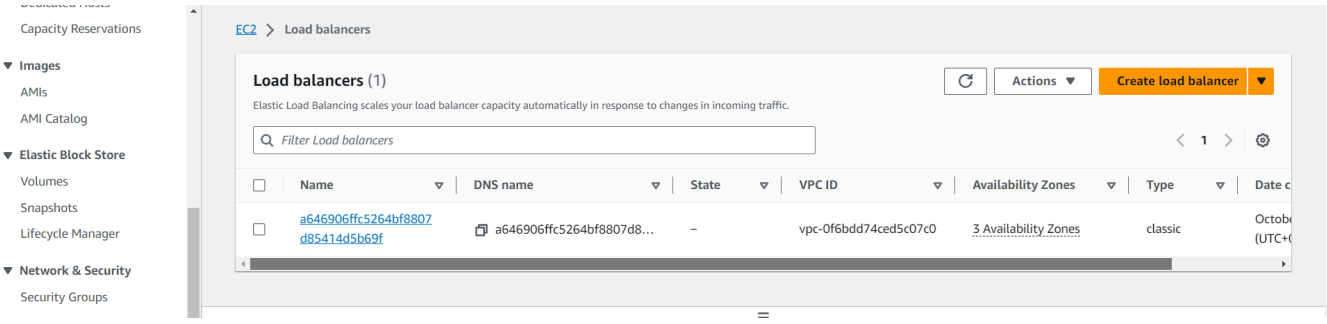


COMMANDS ARGOCD

By default, argocd-server is not publicly exposed. For this project, we will use a Load Balancer to make it usable:

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

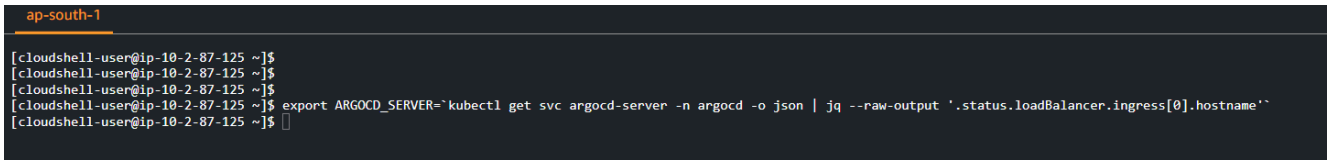
One load balancer will created in the AWS



Wait about 2 minutes for the LoadBalancer creation

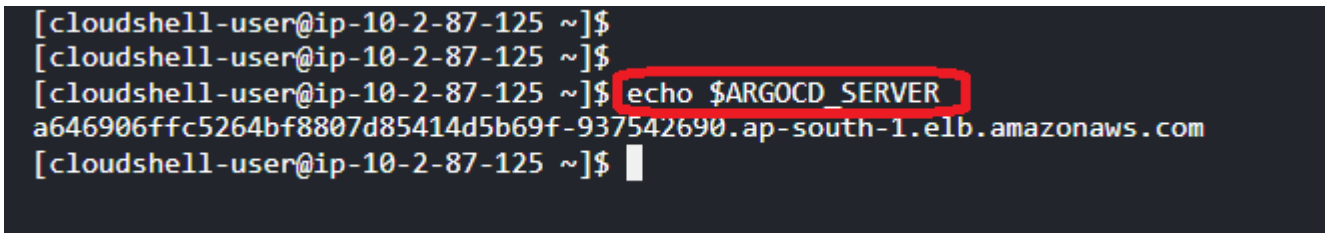
```
:d -o json | jq --raw-output '.status.loadBalancer.ingress[0].hostname'`
```

when you run this command, it will export the hostname of the ArgoCD server's load balancer and store it in the `ARGOCD_SERVER` environment variable, which you can then use in other commands or scripts to interact with the ArgoCD server. This can be useful when you need to access the ArgoCD web UI or interact with the server programmatically.



If run this command you will get the load balancer external IP

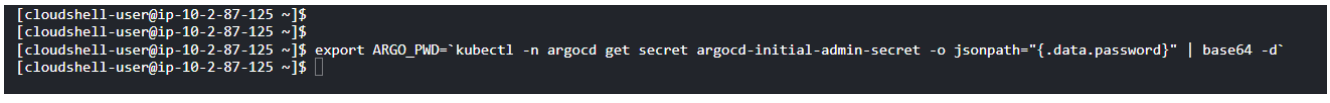
```
echo $ARGOCD_SERVER
```



Login

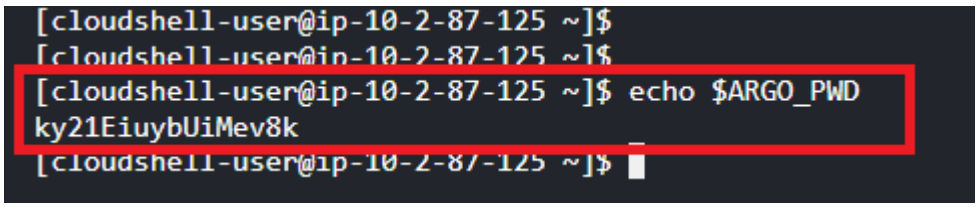
The command you provided is used to extract the password for the initial admin user of ArgoCD, decode it from base64 encoding, and store it in an environment variable named `ARGO_PWD` .

```
export ARGO_PWD=`kubectl -n argocd get secret argocd-initial-admin-secre
```



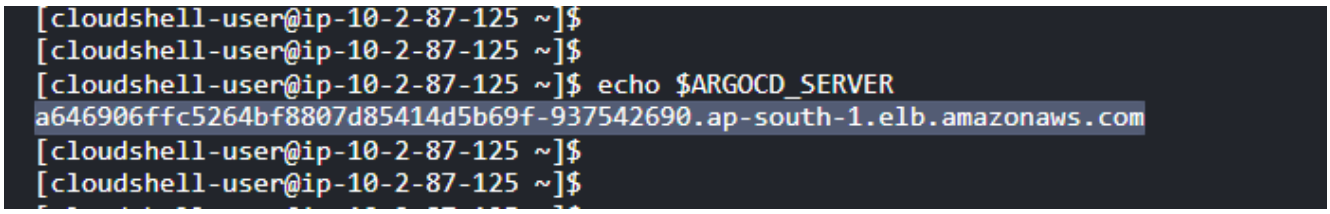
If you want to see your password provide the below command

```
echo $ARGO_PWD
```

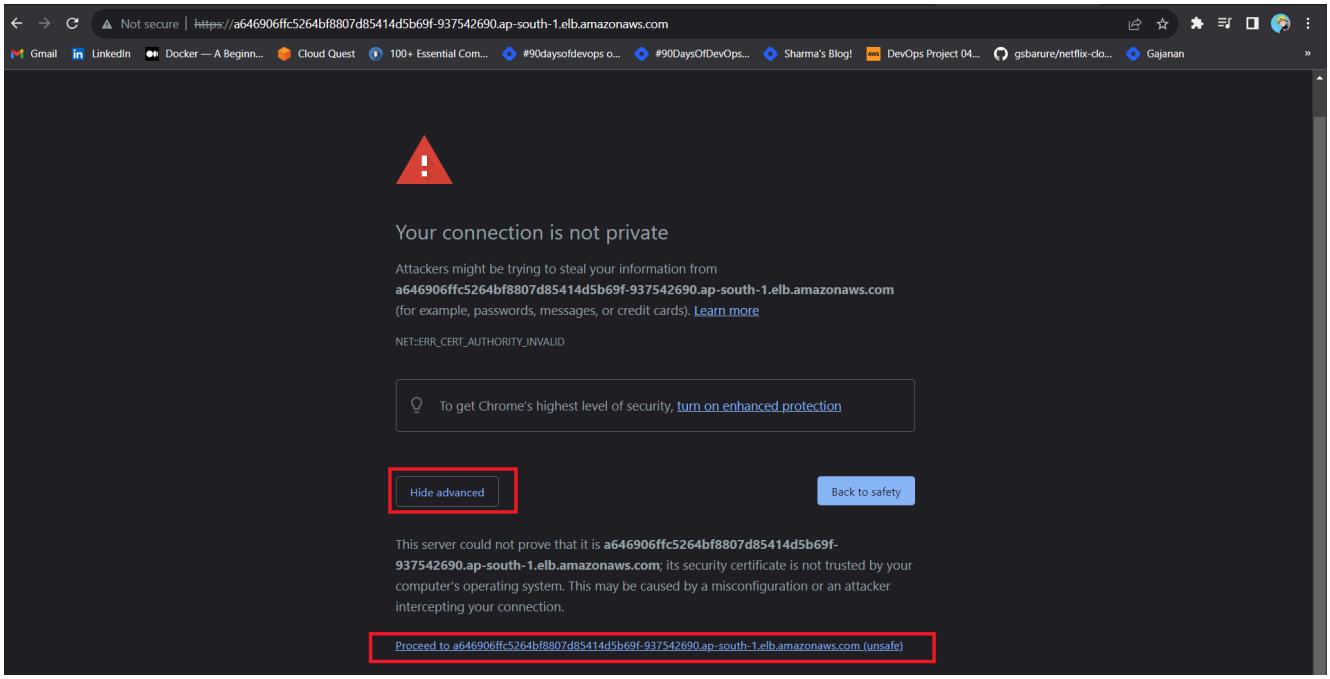


Now copy the load balancer IP and paste it into the browser

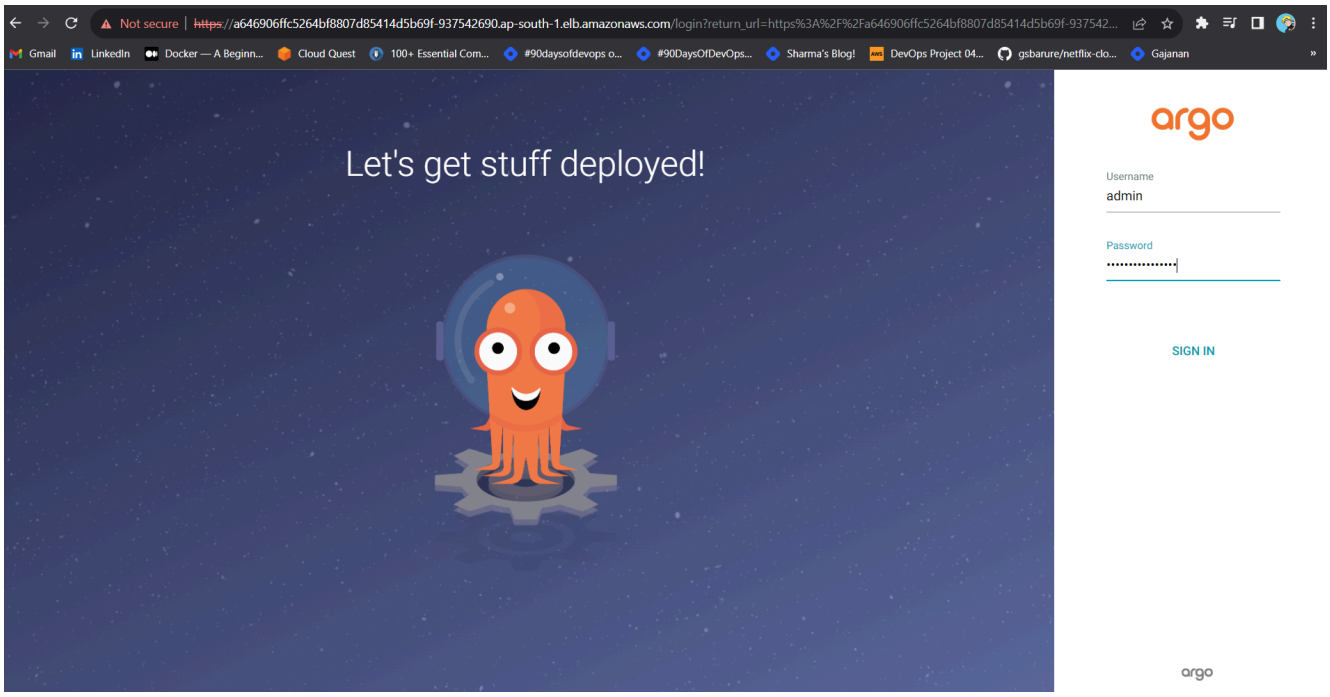
```
echo $ARGOCD_SERVER
```



Now you will see this page. if you get an error click on advanced and click on proceed.



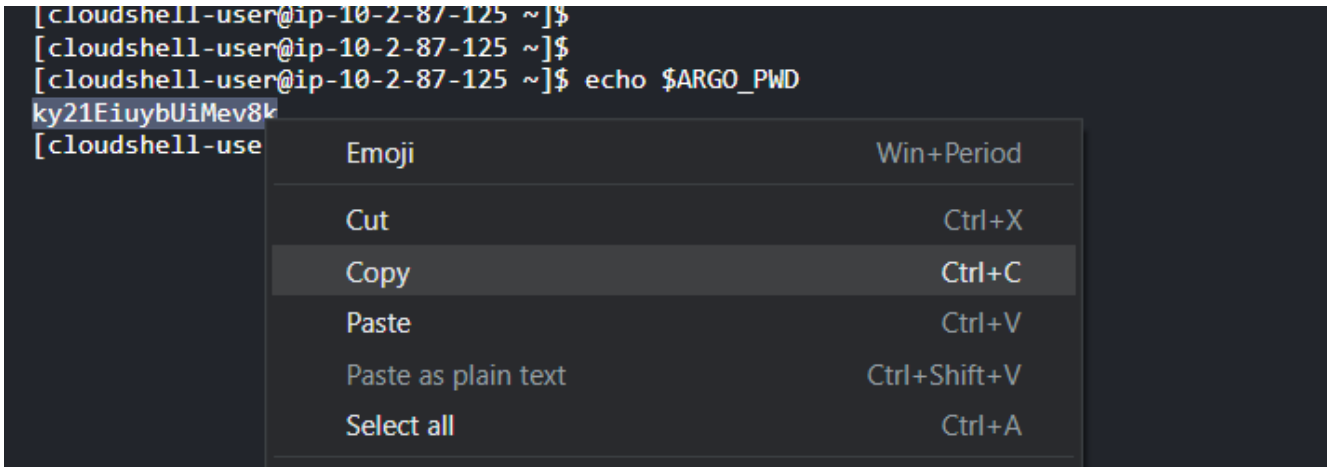
Now you will see this page and log in to ArgoCD



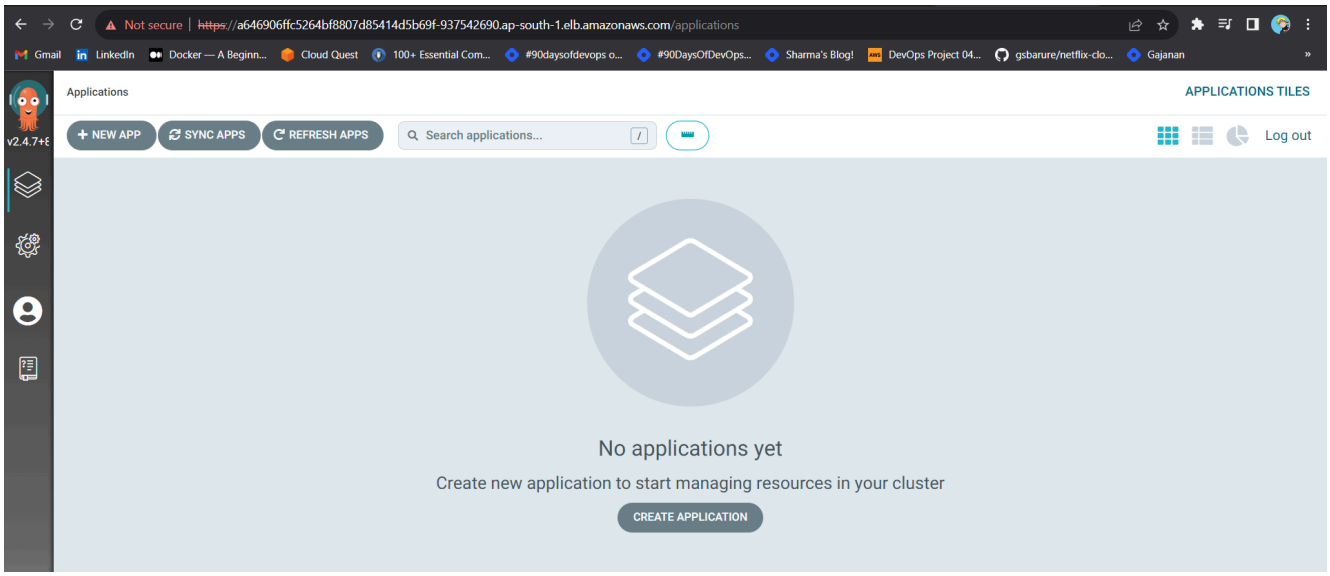
Username is admin

For the password, you have to provide the below command and copy it

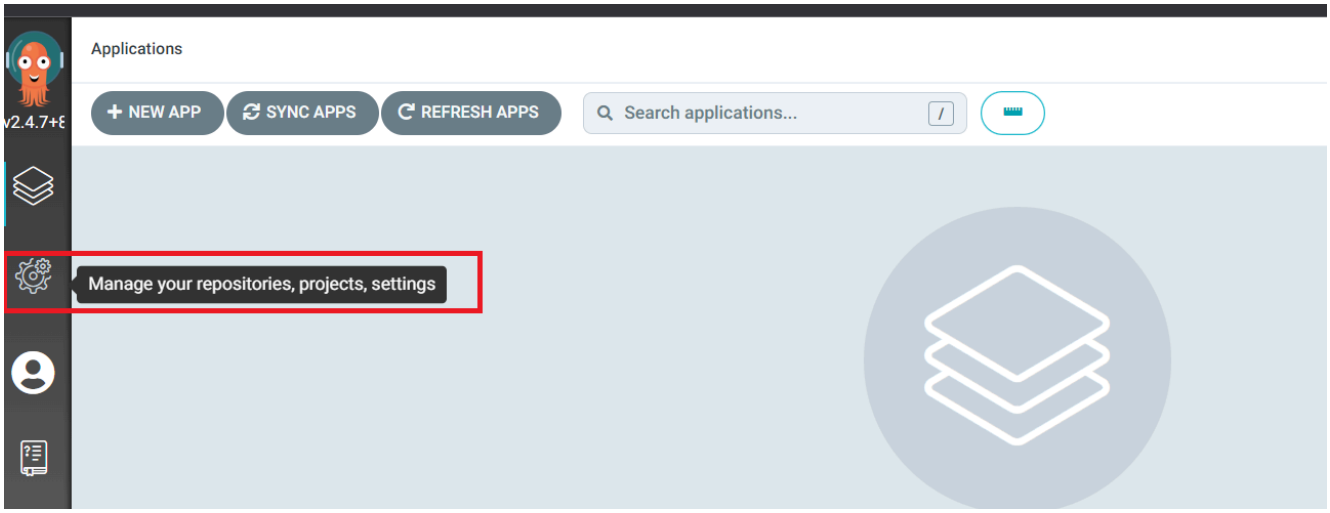
```
echo $ARGO_PWD
```



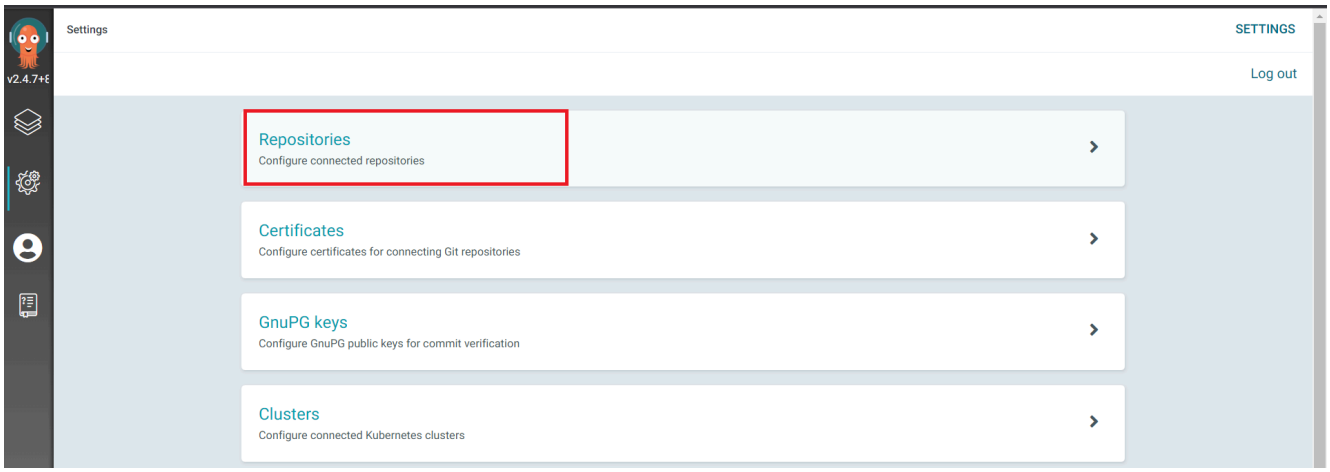
Click on Signin and you will see this page.



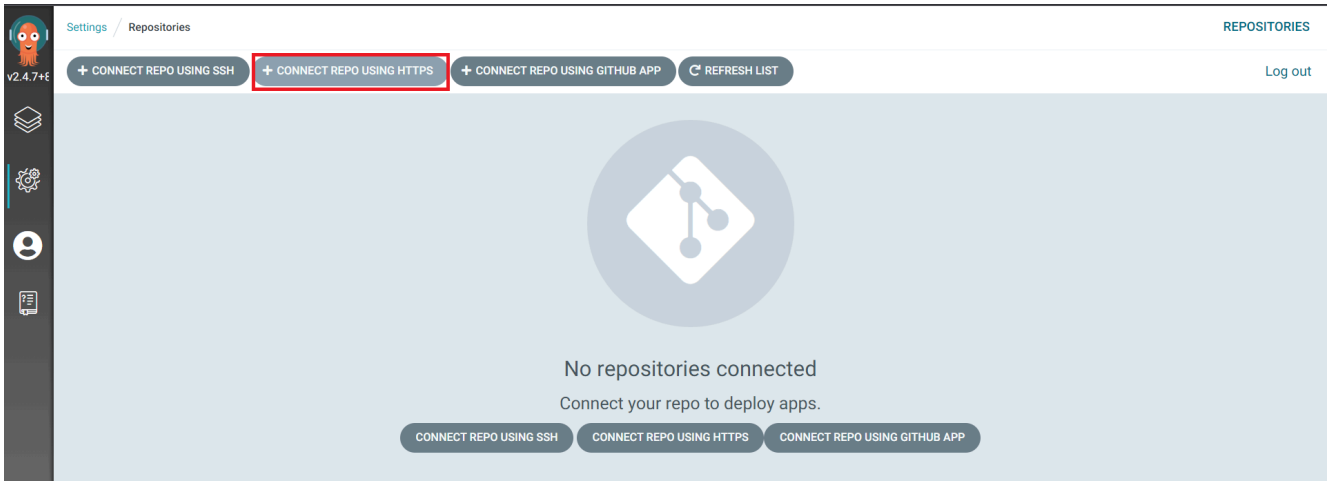
Now click on the Setting gear icon in the left side panel



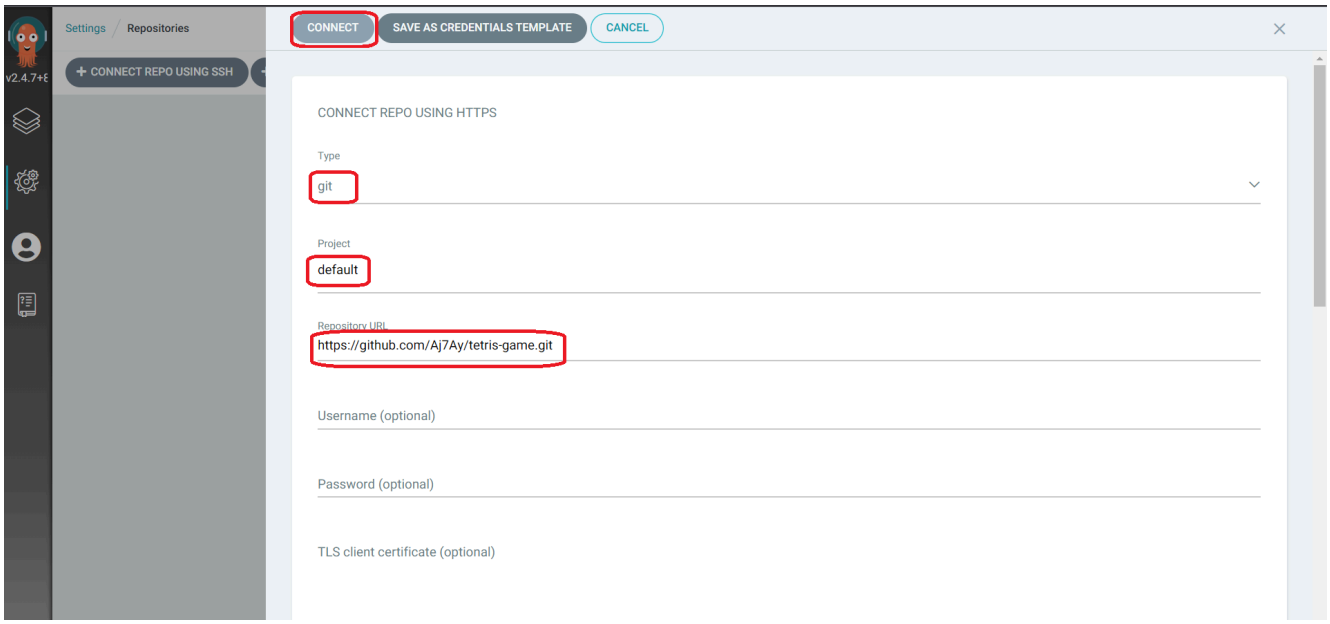
Click on Repositories



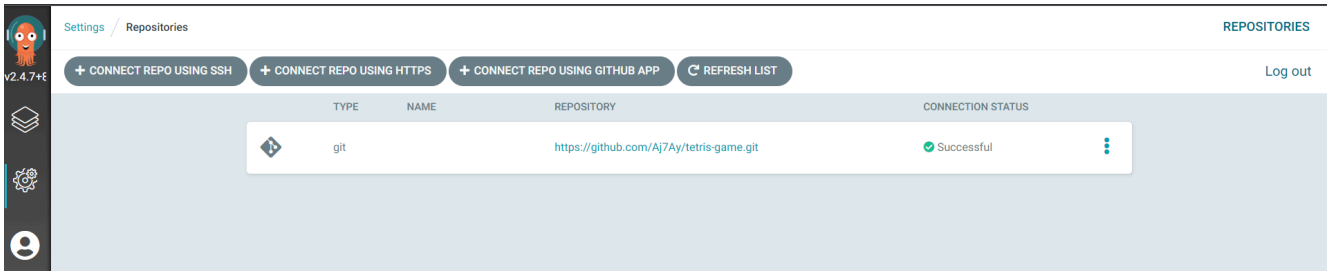
Now click on Connect Repo Using HTTPS



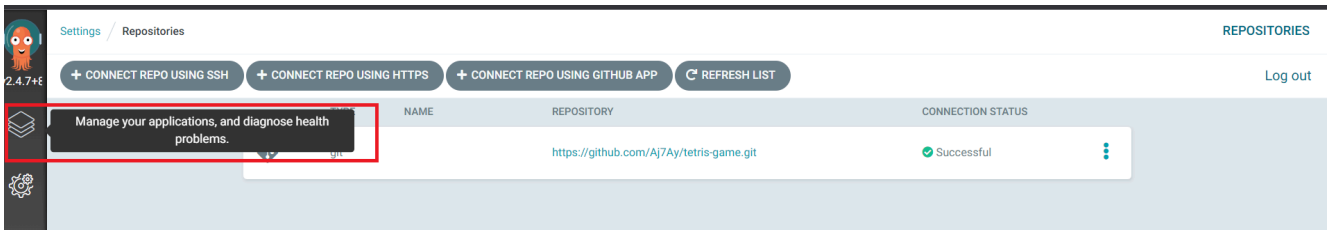
Add Github details, Type as git, Project as default and provide the GitHub URL of this project and click on connect



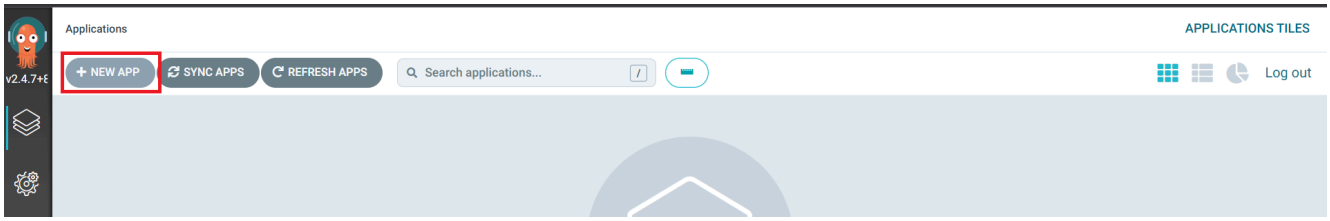
You will get Connection Status as Successful



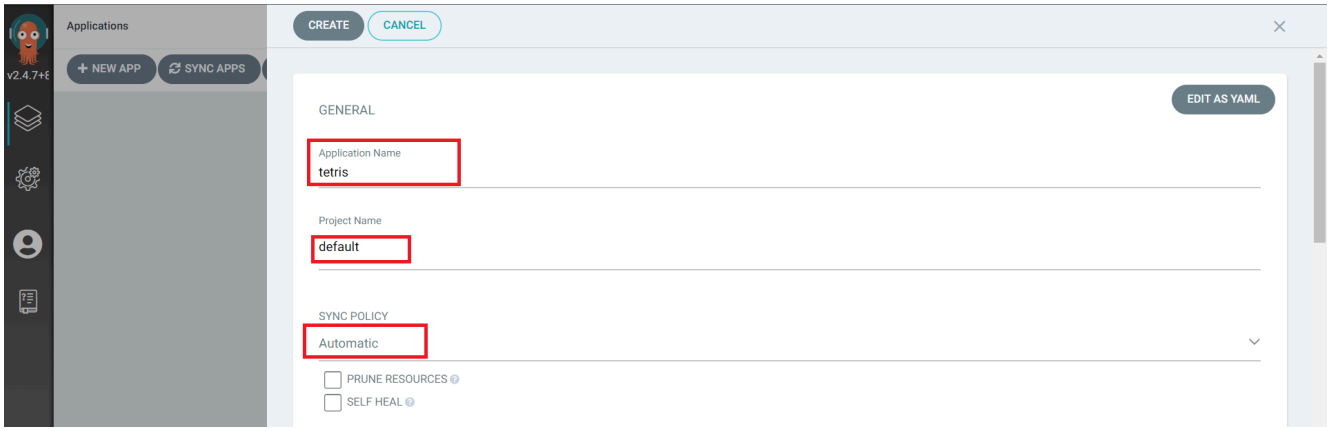
Click on Manage Your application

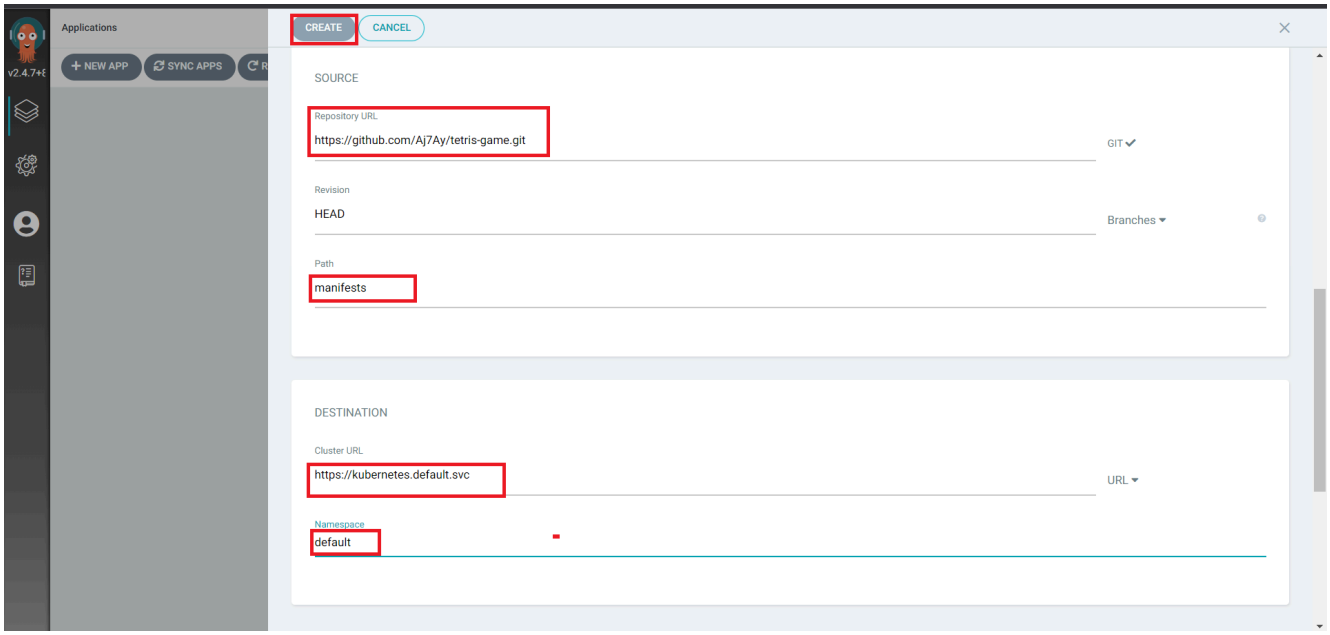


You will see this page and click on New App



Now provide the following details as in the image





Finally, click on Create

In the deployment file, we used the image as

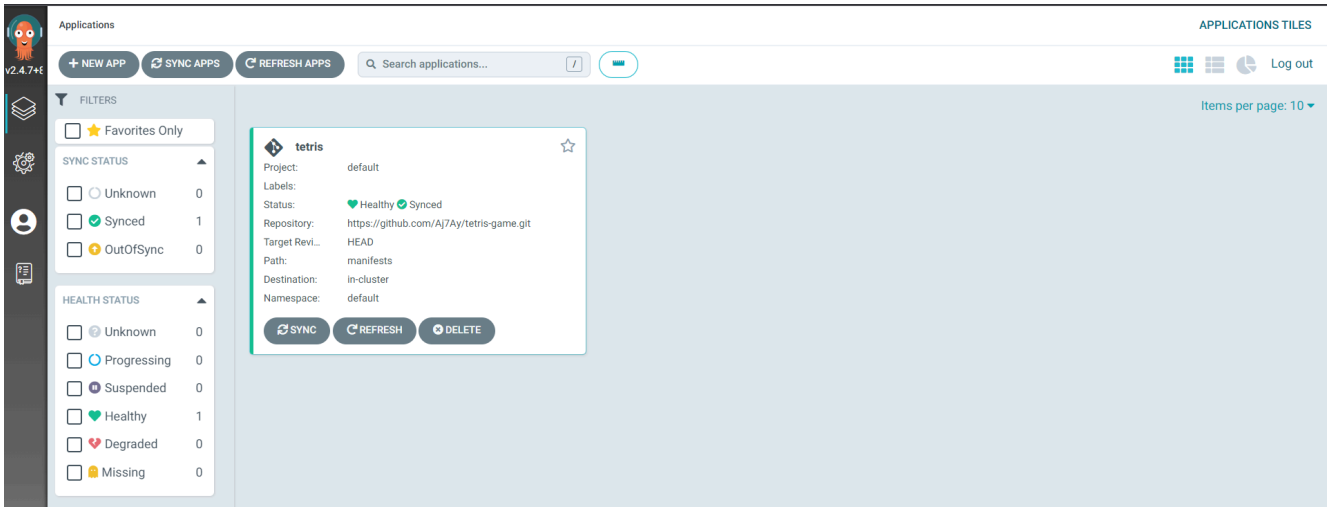
tetris-game / manifests / tetris-deploy.yaml

Aj7Ay Update tetris-deploy.yaml

Code Blame 21 lines (20 loc) · 496 Bytes

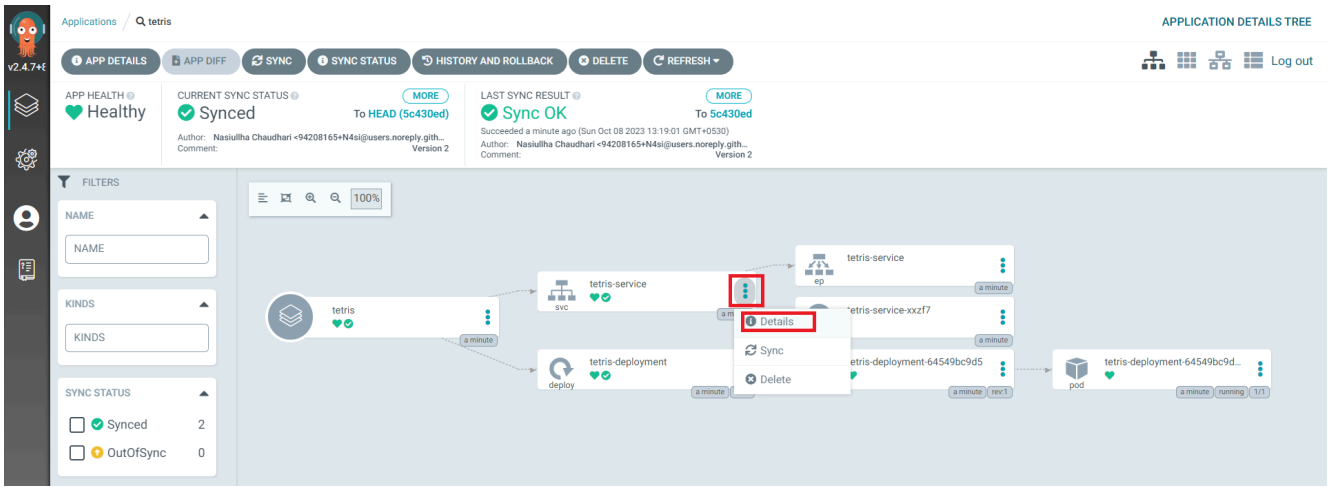
```yaml
# tetris-deployment.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  name: tetris-deployment
spec:
  replicas: 1 # You can adjust the number of replicas as needed
  selector:
    matchLabels:
      app: tetris
  template:
    metadata:
      labels:
        app: tetris
    spec:
      containers:
        - name: tetris
          image: nasi101/tetris # Replace with the actual image tag
          ports:
            - containerPort: 80 # Replace with the port your Tetris game listens on
```

Now you will see a new App has been created named tetris

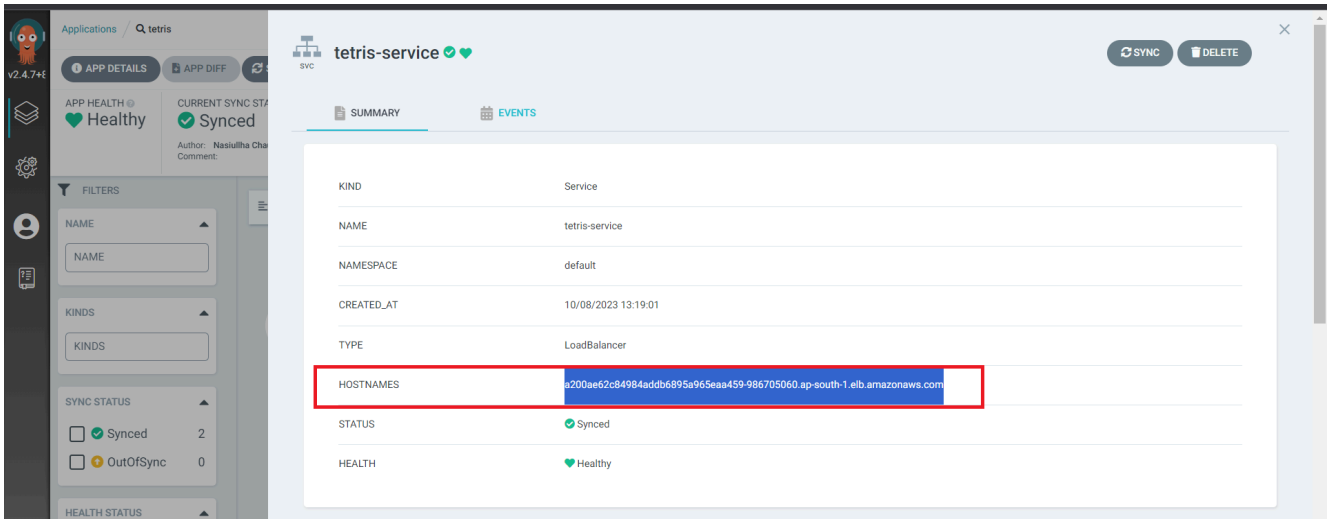


Click on tetris

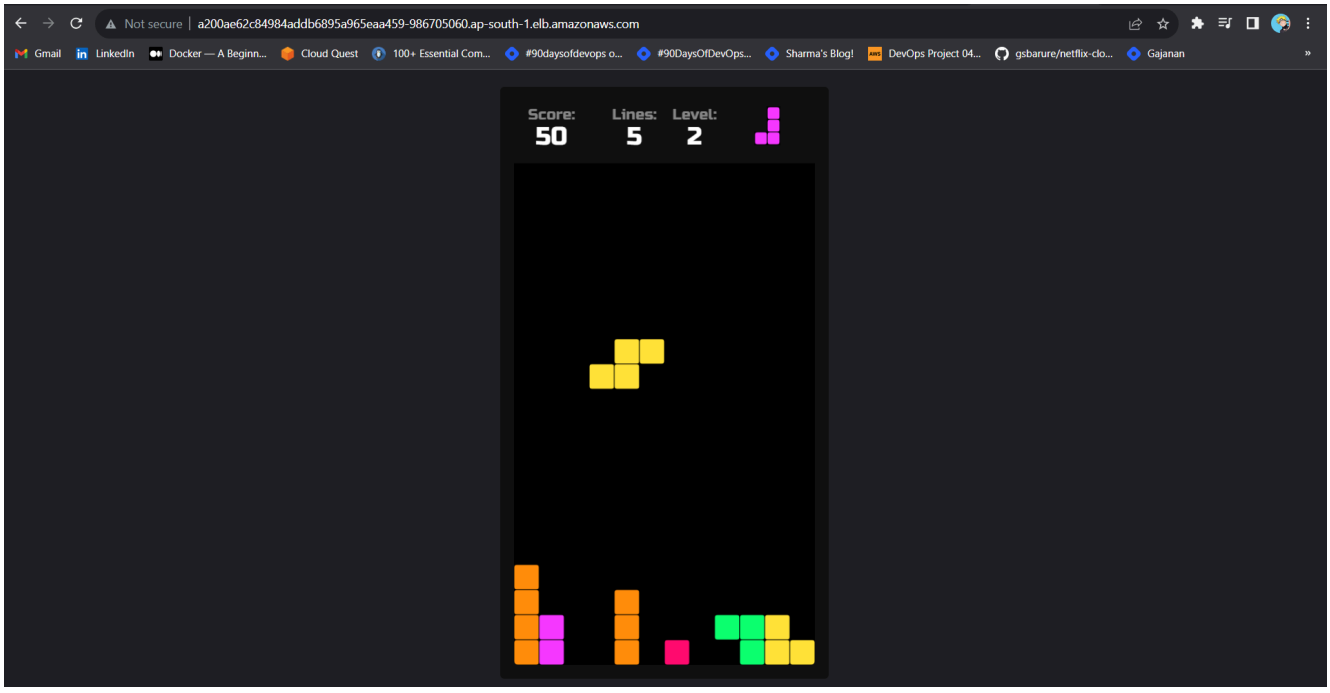
Now click on three dots beside tetris-service and click on the details



Now copy the hostname address



Paste it in a browser you will see this page



Now play the game of version 1.

Step 4: Change the version of the Game

Now the change version of the game in the deployment file

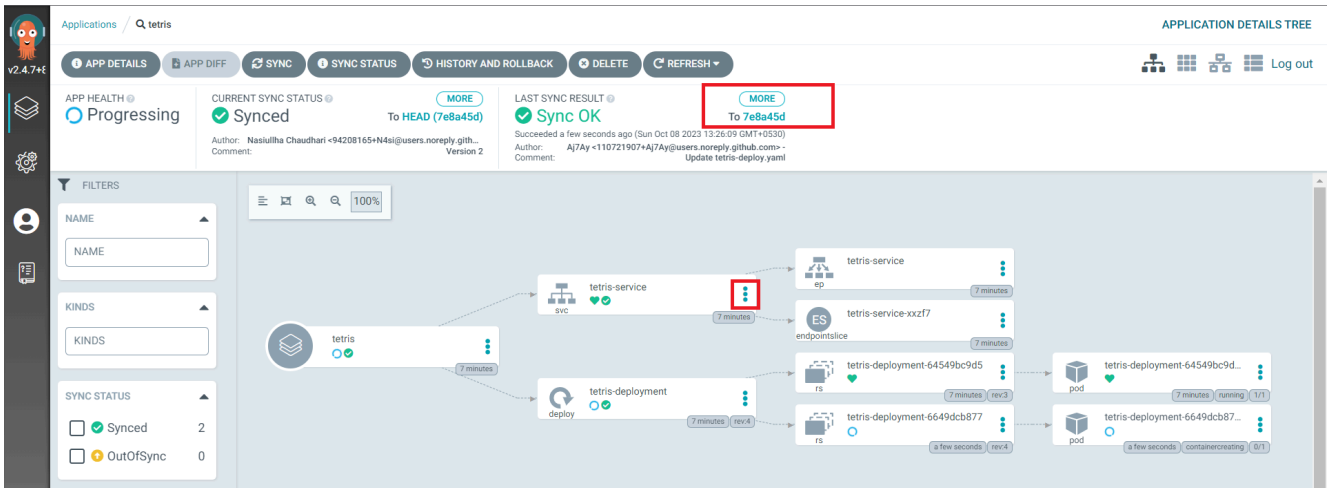
tetris-game / manifests / tetris-deploy.yaml

N4si Version 2

Code Blame 21 lines (20 loc) · 498 Bytes

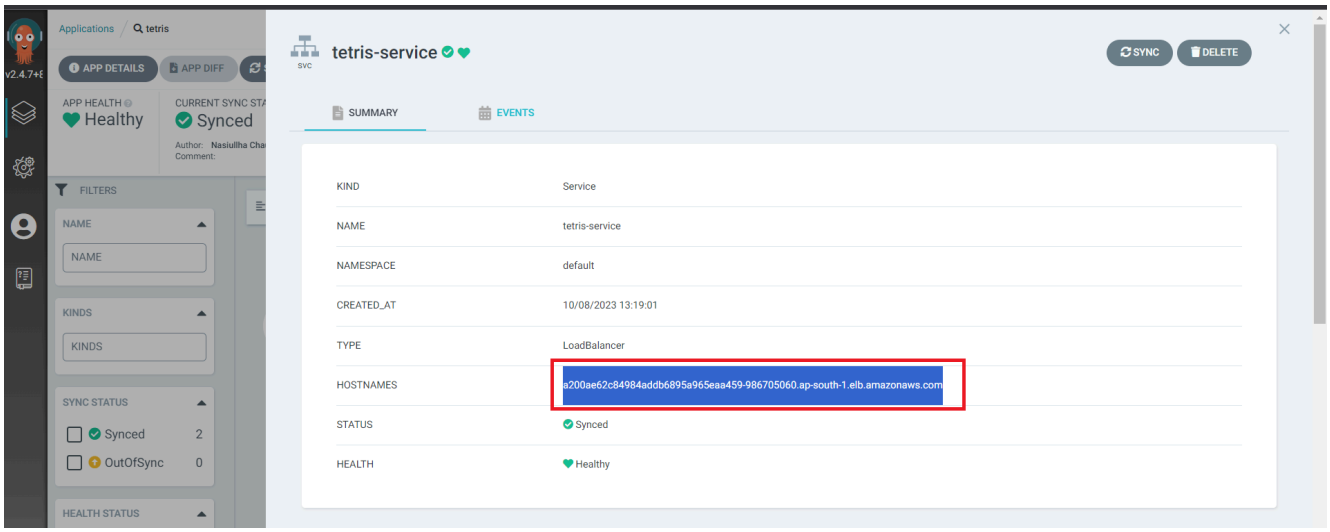
```yaml
# tetris-deployment.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  name: tetris-deployment
spec:
  replicas: 1 # You can adjust the number of replicas as needed
  selector:
    matchLabels:
      app: tetris
  template:
    metadata:
      labels:
        app: tetris
    spec:
      containers:
        - name: tetris
          image: nasi101/tetrisv2 # Replace with the actual image tag
          ports:
            - containerPort: 80 # Replace with the port your Tetris game listens on
```

ArgoCD now automatically starts the build when you change anything in your Repository or the files

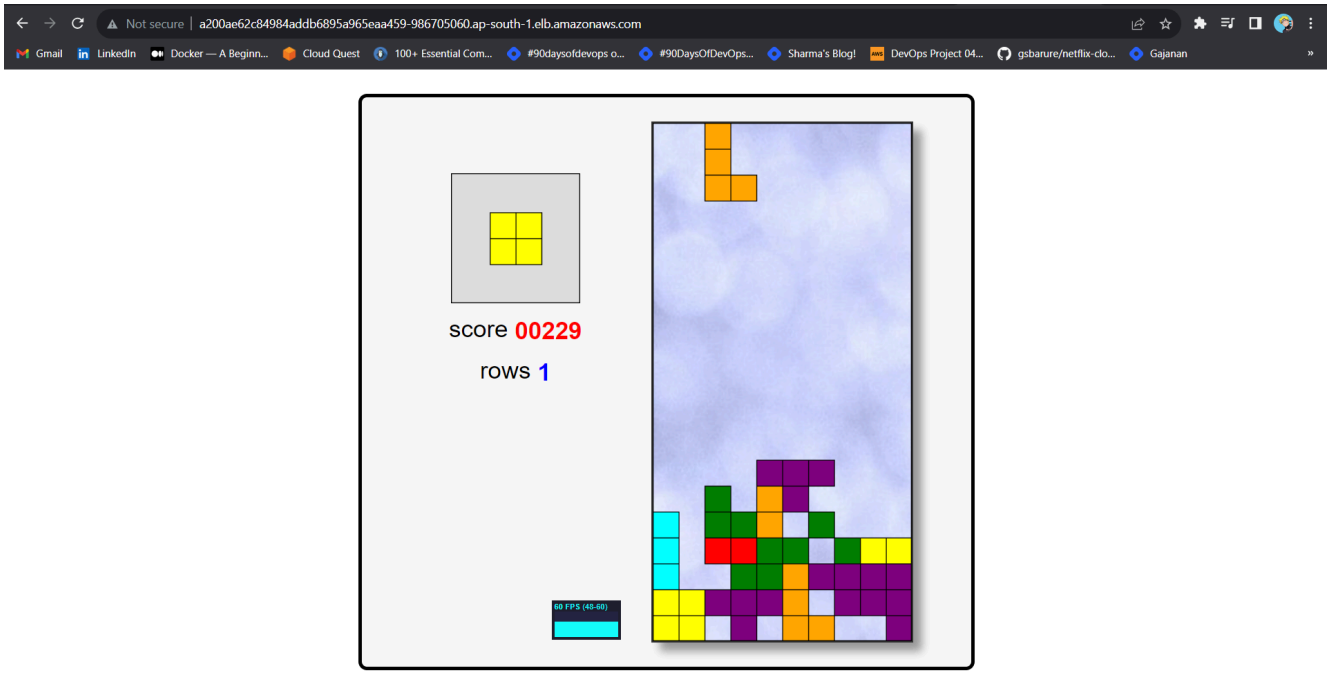


Now Last Sync id changed

Click on three dots at service and copy the hostname address



Now you will see the second version of the Game



Let's Break the Highest score in Tetris.

```
kubectl get all
```

```
[cloudshell-user@ip-10-2-87-125 ~]$
[cloudshell-user@ip-10-2-87-125 ~]$
[cloudshell-user@ip-10-2-87-125 ~]$
[cloudshell-user@ip-10-2-87-125 ~]$ kubectl get all
NAME                                 READY   STATUS    RESTARTS   AGE
pod/tetris-deployment-6649dcb877-rkxk2   1/1     Running   0          4m29s

NAME                     TYPE           CLUSTER-IP      EXTERNAL-IP                                                                PORT(S)        AGE
service/kubernetes       ClusterIP      10.100.0.1      <none>                                                                     443/TCP        35m
service/tetris-service   LoadBalancer   10.100.69.208   a200ae62c84984addb6895a965eaa459-986705060.ap-south-1.elb.amazonaws.com   80:30627/TCP   11m

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/tetris-deployment   1/1     1            1           11m

NAME                                           DESIRED   CURRENT   READY   AGE
replicaset.apps/tetris-deployment-64549bc9d5   0         0         0       11m
replicaset.apps/tetris-deployment-6649dcb877   1         1         1       5m4s
[cloudshell-user@ip-10-2-87-125 ~]$
```

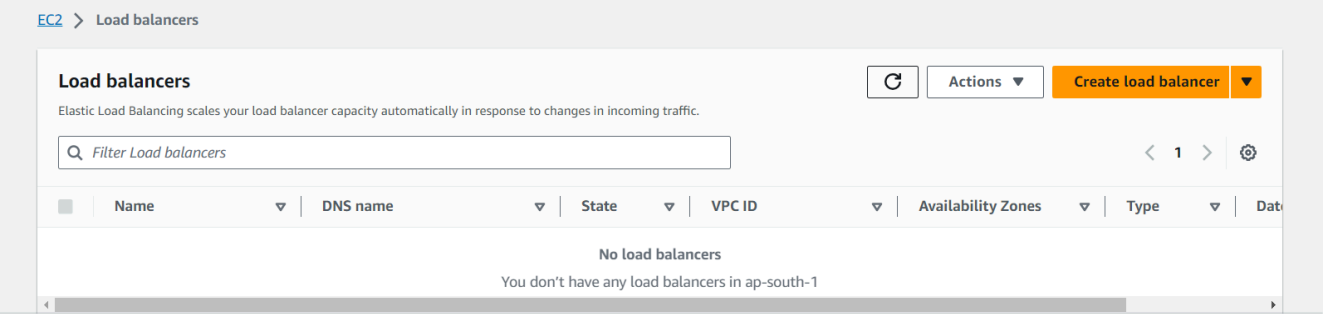
Step 5: Termination

Go to Cloud shell and delete service for ArgoCD load balancer

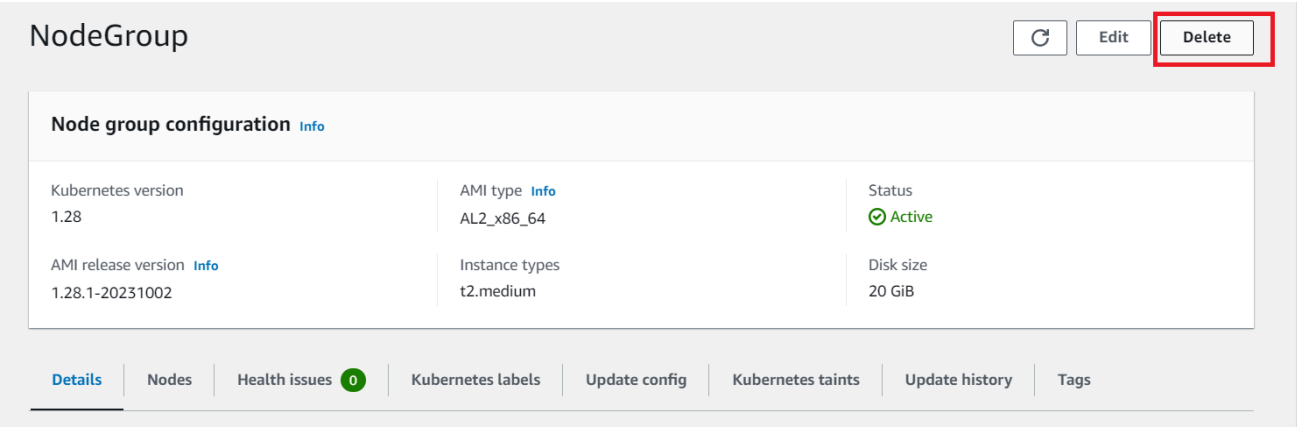
```
kubectl delete svc argocd-server -n argocd
```

```
[cloudshell-user@ip-10-2-87-125 ~]$
[cloudshell-user@ip-10-2-87-125 ~]$
[cloudshell-user@ip-10-2-87-125 ~]$
[cloudshell-user@ip-10-2-87-125 ~]$ kubectl delete svc argocd-server -n argocd
service "argocd-server" deleted
[cloudshell-user@ip-10-2-87-125 ~]$
```

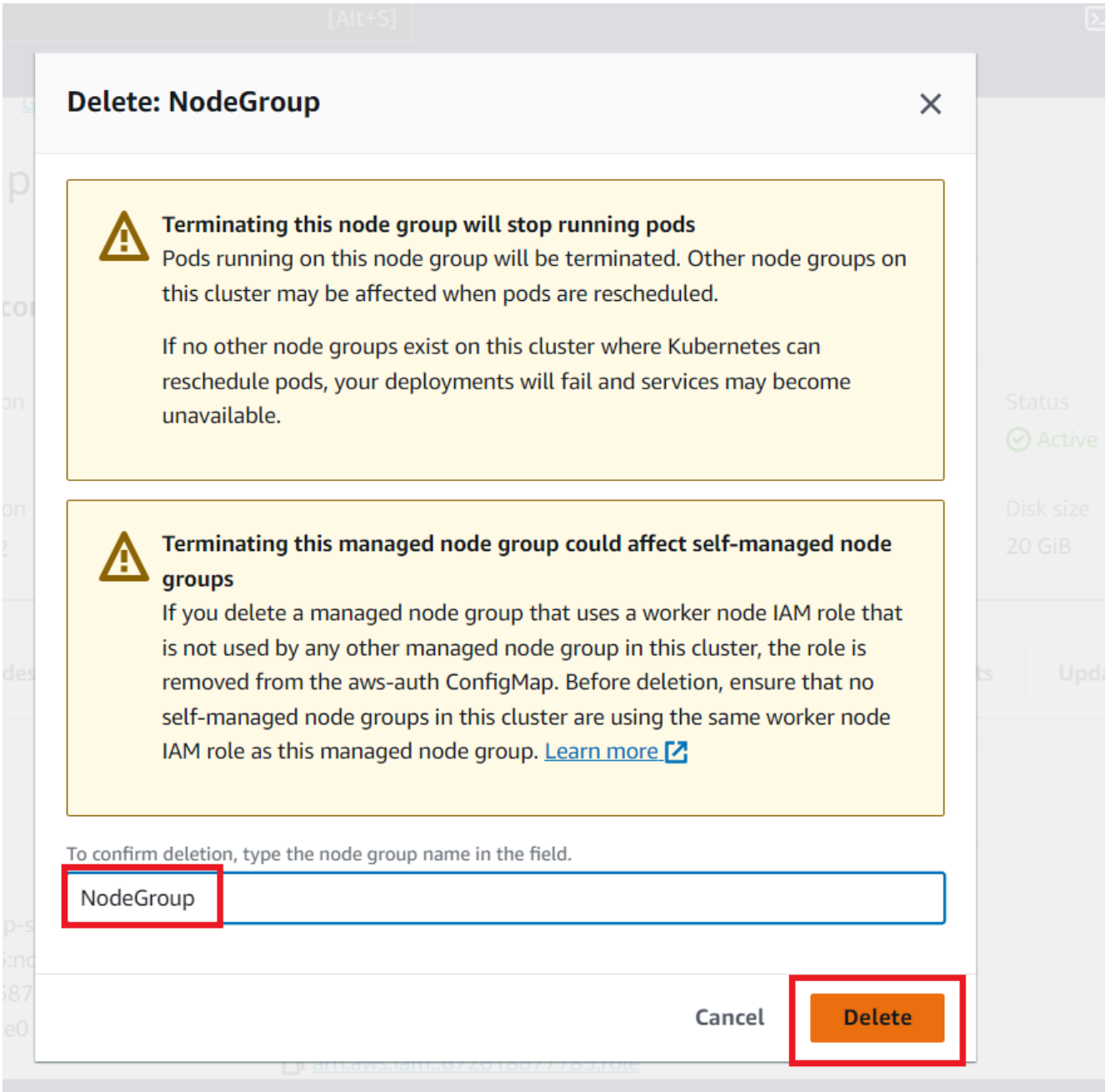
Go to AWS and manually delete the load balancer if it is not deleted.



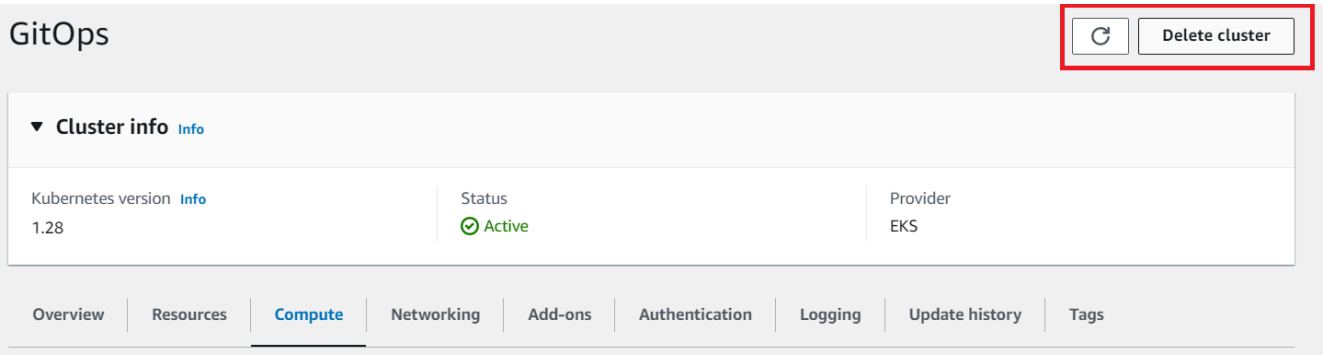
Delete Nodegroup First



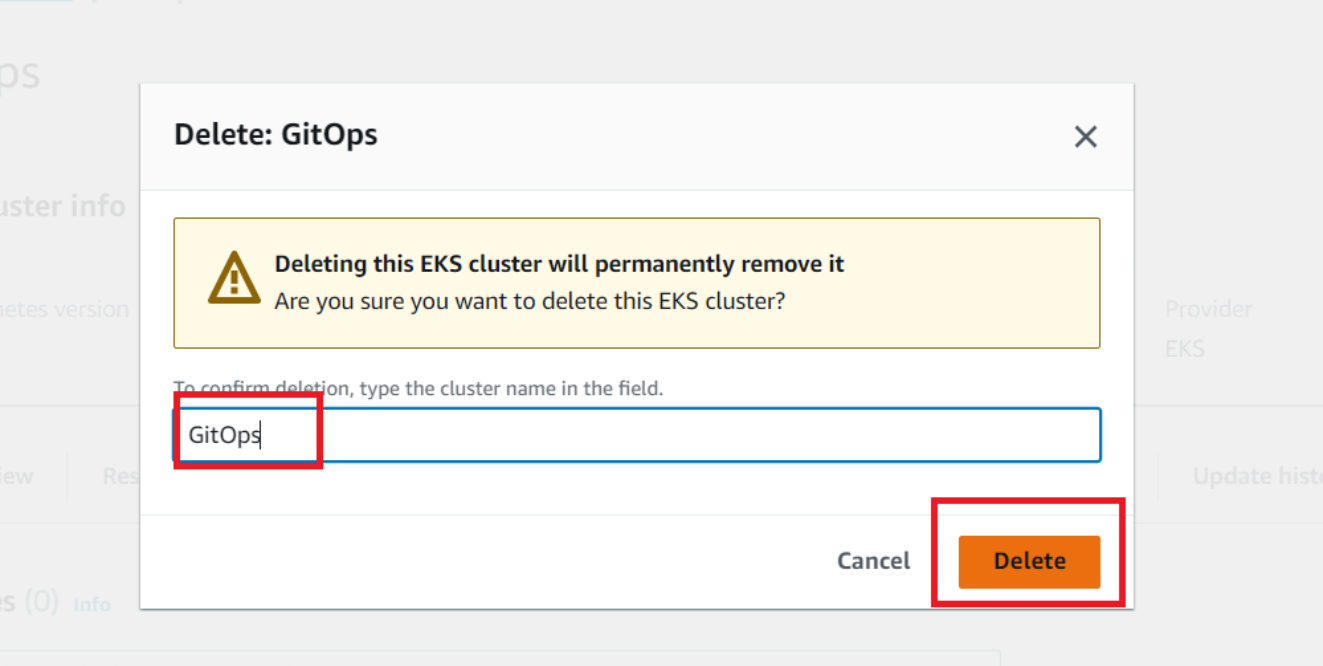
Provide the name that you used for your node group and click delete



Delete the Cluster



Provide the name of the Cluster and delete.



As we conclude our journey into the fascinating realm of GitOps and the thrilling world of Tetris on Kubernetes, let's leave you with a fun fact:

Did you know that the creator of Tetris, Alexey Pajitnov, designed the game in 1984 while working as a computer scientist at the Soviet Academy of Sciences? What started as a simple experiment turned into a worldwide gaming sensation that has captured the hearts and minds of players for decades.

Just like Tetris, GitOps continues to evolve and shape the landscape of modern DevOps, offering innovative solutions for managing complex infrastructures and applications. We hope this adventure has piqued your curiosity and inspired you to explore the limitless possibilities of GitOps further.

So, whether you're deploying code, stacking blocks in Tetris, or exploring new horizons in the tech world, remember that the sky's the limit, and the journey is always filled with surprises. Keep coding, keep gaming, and keep exploring!

Ajay Kumar Yegireddi is a DevSecOps Engineer and System Administrator, with a passion for sharing real-world DevSecOps projects and tasks. **Mr. Cloud Book**, provides hands-on tutorials and practical insights to help others master DevSecOps tools and workflows. Content is designed to bridge the gap between development, security, and operations, making complex concepts easy to understand for both beginners and professionals.

Comments

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

Name *

Email *

Website

Save my name, email, and website in this browser for the next time I comment.

Post Comment

Uncategorized

How to Automate Incident Response : How Q Developer Helped Me Automate a Daily Pain Point

22 July 2025

AI

How to Run Docker Model Runner on Ubuntu 24.04

11 July 2025

AI, DevOps

How to Install docker-ai on Ubuntu 24.04

15 June 2025

Upskill with Ajay: DevSecOps Mastery

Join Mr Cloud book to master DevSecOps through real-world projects. Learn CI/CD, security integration, automation, and more, gaining hands-on skills for industry-level challenges.

Important Links

Privacy Policy

Terms & Conditions

Contact

Resources

Blog

YouTube Channel

Follow Us on YouTube