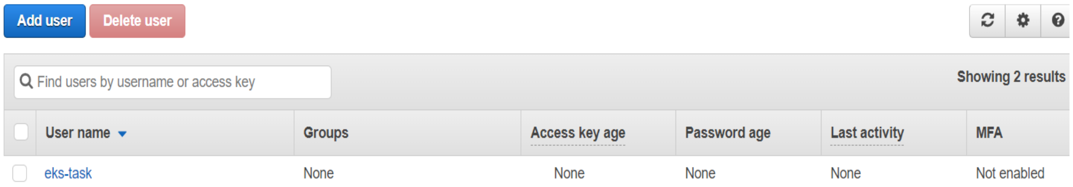
Let,s start the project

Step 1->

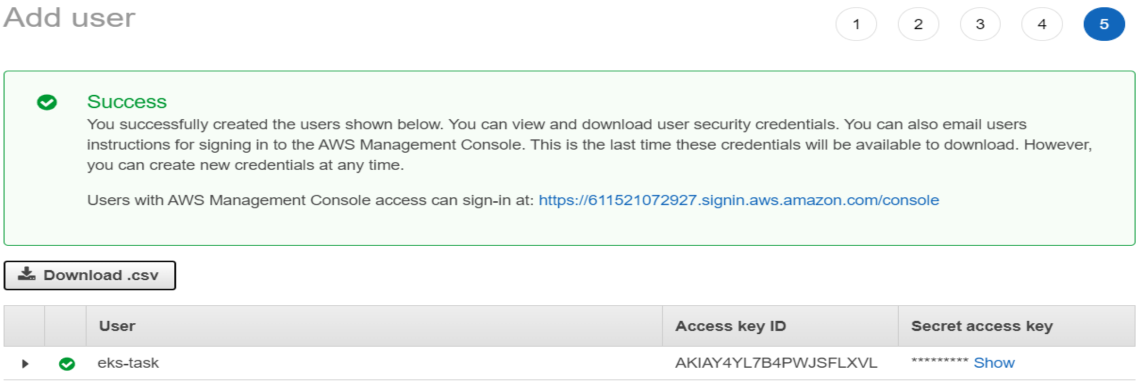
First I have created the user with administration access , this user have all the power same as admin. But , when multiple teams are working together then creating the user with administration access is not a good idea.



	User name	Groups	Access key age	Password age	Last activity	MFA
□	eks-task	None	None	None	None	Not enabled

When we create the user they provide us with the .csv file that contains the Access key ID & Secret access key , this can be downloaded only once.

SO DON,T FORGET TO DOWNLOAD IT !!



Success

You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: https://611521072927.signin.aws.amazon.com/console

User	Access key ID	Secret access key
eks-task	AKIAY4YL7B4PWJSFLXVL	********* Show

Login to AWS using the created user

```
C:\Users\LENOVO PC>aws configure

AWS Access Key ID [****************DVPJ]: AKIAY4YL7B4PWJSFLXVL

AWS Secret Access Key [****************dbUB]: ****************************************

Default region name [ap-south-1]:
```

Default output format [None]

Step 2 ->

Now let's create the cluster

This cluster is launched in Mumbai Region , in this cluster I have created two node groups

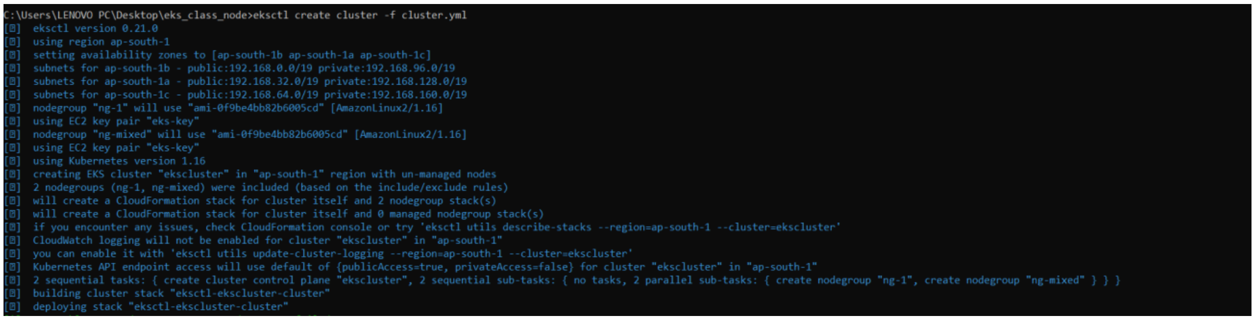
NODE GROUP 1 – This node group has desired capacity of 3 instances with t2.micro instance type

NODE GROUP MIXED – In this node I have launched the spot instances which will we created automatically when need arises , the spot instances are cheaper than Ondemand Instance

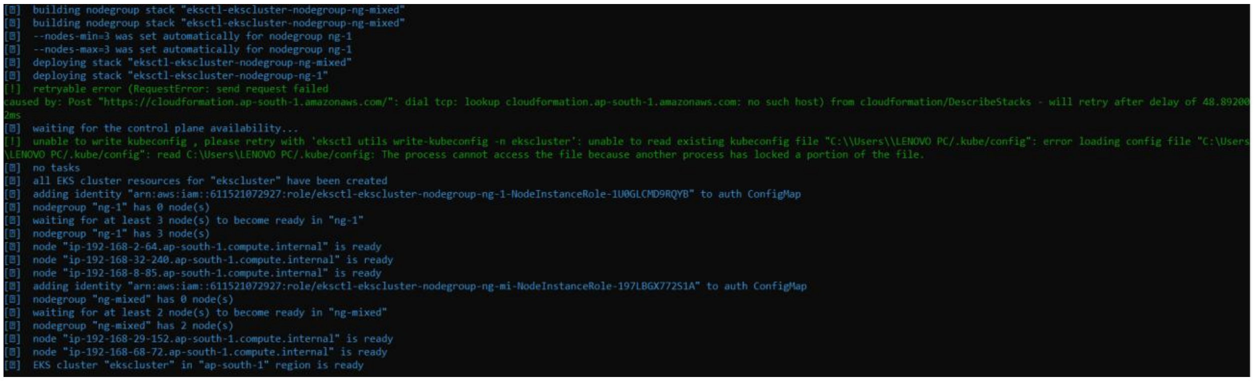
For creating the project we project we have to cluster we have to run the command:

eksctl create cluster –f cluster.yml

For code refer to the cluster.yml file

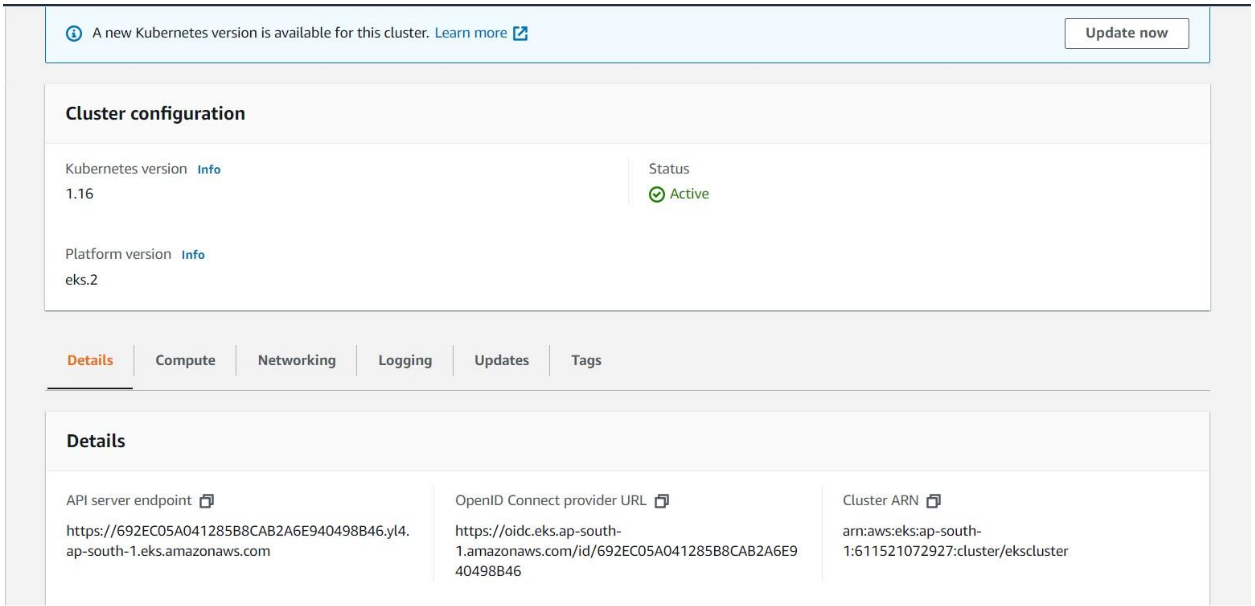


```
C:\Users\LENOVO PC\Desktop\eks_class_node>eksctl create cluster -f cluster.yml
[ℹ]  eksctl version 0.21.0
[ℹ]  using region ap-south-1
[ℹ]  setting availability zones to [ap-south-1b ap-south-1a ap-south-1c]
[ℹ]  subnets for ap-south-1b - public:192.168.0.0/19 private:192.168.96.0/19
[ℹ]  subnets for ap-south-1a - public:192.168.32.0/19 private:192.168.128.0/19
[ℹ]  subnets for ap-south-1c - public:192.168.64.0/19 private:192.168.160.0/19
[ℹ]  nodegroup "ng-1" will use "ami-0f9be4bb82b6005cd" [AmazonLinux2/1.16]
[ℹ]  using EC2 key pair "eks-key"
[ℹ]  nodegroup "ng-mixed" will use "ami-0f9be4bb82b6005cd" [AmazonLinux2/1.16]
[ℹ]  using EC2 key pair "eks-key"
[ℹ]  using Kubernetes version 1.16
[ℹ]  creating EKS cluster "ekscluster" in "ap-south-1" region with un-managed nodes
[ℹ]  2 nodegroups (ng-1, ng-mixed) were included (based on the include/exclude rules)
[ℹ]  will create a CloudFormation stack for cluster itself and 2 nodegroup stack(s)
[ℹ]  will create a CloudFormation stack for cluster itself and 0 managed nodegroup stack(s)
[ℹ]  if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-stacks --region=ap-south-1 --cluster=ekscluster'
[ℹ]  CloudWatch logging will not be enabled for cluster "ekscluster" in "ap-south-1"
[ℹ]  you can enable it with 'eksctl utils update-cluster-logging --region=ap-south-1 --cluster=ekscluster'
[ℹ]  Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess=false} for cluster "ekscluster" in "ap-south-1"
[ℹ]  2 sequential tasks: { create cluster control plane "ekscluster", 2 sequential sub-tasks: { no tasks, 2 parallel sub-tasks: { create nodegroup "ng-1", create nodegroup "ng-mixed" } } }
[ℹ]  building cluster stack "eksctl-ekscluster-cluster"
[ℹ]  deploying stack "eksctl-ekscluster-cluster"
```



```
[ℹ]  building nodegroup stack "eksctl-ekscluster-nodegroup-ng-mixed"
[ℹ]  building nodegroup stack "eksctl-ekscluster-nodegroup-ng-mixed"
[ℹ]  --nodes-min=3 was set automatically for nodegroup ng-1
[ℹ]  --nodes-max=3 was set automatically for nodegroup ng-1
[ℹ]  deploying stack "eksctl-ekscluster-nodegroup-ng-mixed"
[ℹ]  deploying stack "eksctl-ekscluster-nodegroup-ng-1"
[!]  retryable error (RequestError: send request failed
caused by: Post "https://cloudformation.ap-south-1.amazonaws.com/": dial tcp: lookup cloudformation.ap-south-1.amazonaws.com: no such host) from cloudformation/DescribeStacks - will retry after delay of 48.89200
2ms
[ℹ]  waiting for the control plane availability...
[!]  unable to write kubeconfig , please retry with 'eksctl utils write-kubeconfig -n ekscluster': unable to read existing kubeconfig file "C:\\Users\\LENOVO PC/.kube/config": error loading config file "C:\Users
\LENOVO PC/.kube/config": read C:\Users\LENOVO PC/.kube/config: The process cannot access the file because another process has locked a portion of the file.
[ℹ]  no tasks
[ℹ]  all EKS cluster resources for "ekscluster" have been created
[ℹ]  adding identity "arn:aws:iam::611521072927:role/eksctl-ekscluster-nodegroup-ng-1-NodeInstanceRole-1U0GLCMD9RQYB" to auth ConfigMap
[ℹ]  nodegroup "ng-1" has 0 node(s)
[ℹ]  waiting for at least 3 node(s) to become ready in "ng-1"
[ℹ]  nodegroup "ng-1" has 3 node(s)
[ℹ]  node "ip-192-168-2-64.ap-south-1.compute.internal" is ready
[ℹ]  node "ip-192-168-32-240.ap-south-1.compute.internal" is ready
[ℹ]  node "ip-192-168-8-85.ap-south-1.compute.internal" is ready
[ℹ]  adding identity "arn:aws:iam::611521072927:role/eksctl-ekscluster-nodegroup-ng-mi-NodeInstanceRole-197LBGX772S1A" to auth ConfigMap
[ℹ]  nodegroup "ng-mixed" has 0 node(s)
[ℹ]  waiting for at least 2 node(s) to become ready in "ng-mixed"
[ℹ]  nodegroup "ng-mixed" has 2 node(s)
[ℹ]  node "ip-192-168-29-152.ap-south-1.compute.internal" is ready
[ℹ]  node "ip-192-168-68-72.ap-south-1.compute.internal" is ready
[ℹ]  EKS cluster "ekscluster" in "ap-south-1" region is ready
```

See here our cluster created with all the required instances ,it will take around 15-20min to create and activate our cluster



ⓘ A new Kubernetes version is available for this cluster. Learn more

Cluster configuration

Kubernetes version Info
1.16

Status
✓ Active

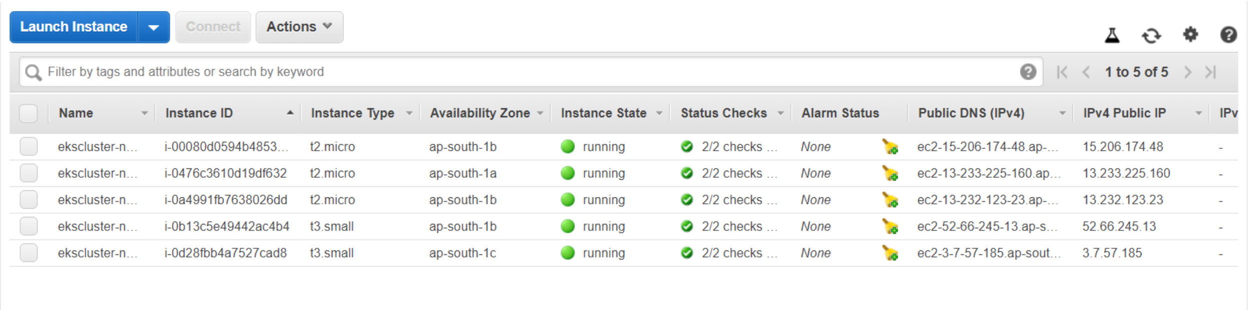
Platform version Info
eks.2

Details

API server endpoint
https://692EC05A041285B8CAB2A6E940498B46.yl4.ap-south-1.eks.amazonaws.com

OpenID Connect provider URL
https://oidc.eks.ap-south-1.amazonaws.com/id/692EC05A041285B8CAB2A6E940498B46

Cluster ARN
arn:aws:eks:ap-south-1:611521072927:cluster/ekscluster

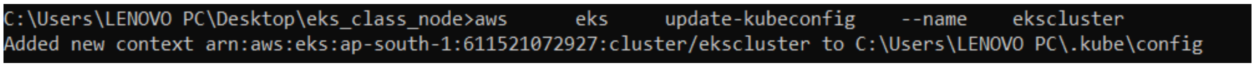


Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv
ekscluster-n...	i-00080d0594b4853...	t2.micro	ap-south-1b	● running	✓ 2/2 checks ...	None	ec2-15-206-174-48.ap-...	15.206.174.48	-
ekscluster-n...	i-0476c3610d19df632	t2.micro	ap-south-1a	● running	✓ 2/2 checks ...	None	ec2-13-233-225-160.ap...	13.233.225.160	-
ekscluster-n...	i-0a4991fb7638026dd	t2.micro	ap-south-1b	● running	✓ 2/2 checks ...	None	ec2-13-232-123-23.ap-...	13.232.123.23	-
ekscluster-n...	i-0b13c5e49442ac4b4	t3.small	ap-south-1b	● running	✓ 2/2 checks ...	None	ec2-52-66-245-13.ap-s...	52.66.245.13	-
ekscluster-n...	i-0d28fbb4a7527cad8	t3.small	ap-south-1c	● running	✓ 2/2 checks ...	None	ec2-3-7-57-185.ap-sout...	3.7.57.185	-

We also have to update the config file of kubernetes in .kube folder , this file contains where our Kubernetes is running

Rather than doing it manually we can do it automatically by running this command

aws eks update-kubeconfig --name ekscluster



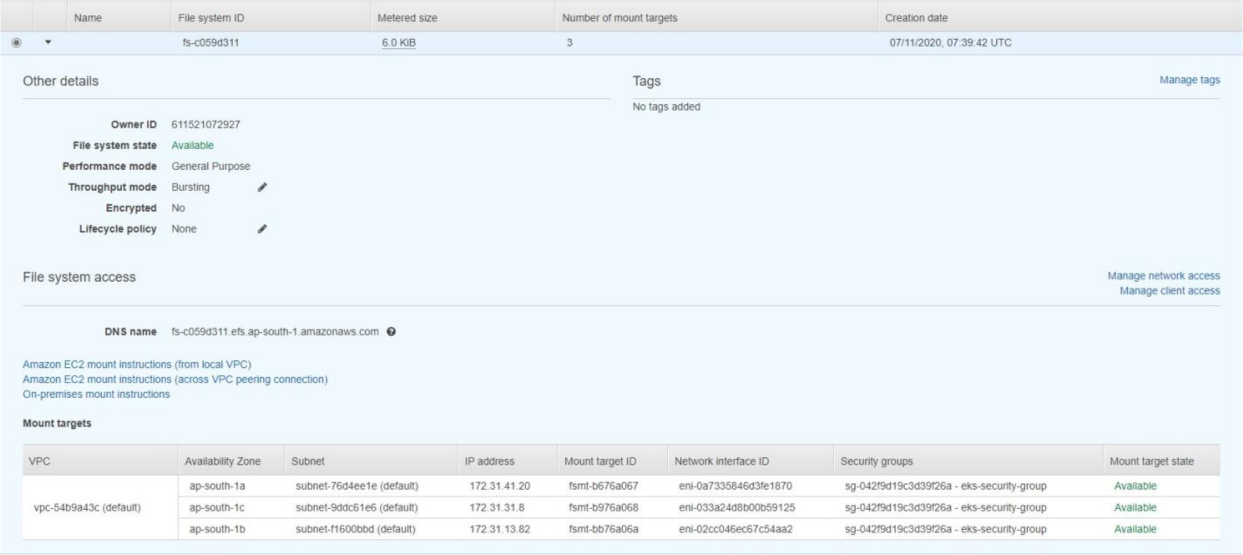
```
C:\Users\LENOVO PC\Desktop\eks_class_node>aws      eks      update-kubeconfig     --name     ekscluster
Added new context arn:aws:eks:ap-south-1:611521072927:cluster/ekscluster to C:\Users\LENOVO PC\.kube\config
```

Step 3 ->

Now let’s create EFS storage so that further it can be used as PVC. We haven’t used the EBS storage due to some reasons:

1. EBS can’t be attached to multiple Instances
2. EBS is can’t be attached to instance running in other subnet

So I have created the EFS in cluster VPC, this VPC is created for our cluster and with a Security Group that has all incoming and outgoing traffic , it is also created when we created the cluster



		Name	File system ID	Metered size	Number of mount targets	Creation date
●	▾		fs-c059d311	6.0 KiB	3	07/11/2020, 07:39:42 UTC

Other details

Owner ID	611521072927
File system state	Available
Performance mode	General Purpose
Throughput mode	Bursting
Encrypted	No
Lifecycle policy	None

Tags

No tags added

File system access

DNS name fs-c059d311.efs.ap-south-1.amazonaws.com

Mount targets

VPC	Availability Zone	Subnet	IP address	Mount target ID	Network interface ID	Security groups	Mount target state
vpc-54b9a43c (default)	ap-south-1a	subnet-76d4ee1e (default)	172.31.41.20	fsmt-b676a067	eni-0a7335846d3fe1870	sg-042f9d19c3d39f26a - eks-security-group	Available
	ap-south-1c	subnet-9ddc61e6 (default)	172.31.31.8	fsmt-b976a068	eni-033a24d8b00b59125	sg-042f9d19c3d39f26a - eks-security-group	Available
	ap-south-1b	subnet-f1600bbd (default)	172.31.13.82	fsmt-bb76a06a	eni-02cc046ec67c54aa2	sg-042f9d19c3d39f26a - eks-security-group	Available

Manage tags
Manage network access
Manage client access
Amazon EC2 mount instructions (from local VPC)
Amazon EC2 mount instructions (across VPC peering connection)
On-premises mount instructions

Step 4->

 Now here we create EFS provisioner , this will allows us to mount the EFS storage as PersistentVolumes(PV).I have also created a separate namespace as eks-task to perform this task

For creating the namespace and EFS provisioner run the following command:

kubectl create ns eks-task

kubectl create –f create-efs-provisioner.yaml –n eks-task

For code refer to the create-efs-provisioner.yaml file

NOTE: By default the instance doesn't support EFS , so go inside the instance and run the yum install amazon-efs-utils , it will download the required softwares

Step 5->

In this step we modifiy some permission using the ROLE BASED ACCESS CONTROL by few lines of code.

For creating the ROLE run the command:

kubectl create –f create-rbac.yaml -n eks-task

For code refer to the create-rbac.yaml file

Step 6->

Here we create the storage class that will create pvc and provide pv dynamically.I have also created two pvc each of 10GiB. These pvc will be attached to mysql and ghost

For creating the storage and pvc run the following command:

kubectl create –f create-storage.yaml –n eks-task

For code refer to the create-storage.yaml file

Step 7->

Since now we will be launching our MYSQL and Ghost ,so for their login we require username & password but specifying the actual will make our login unsafe ,so for this I have created a secret.yml file that contains login and password in encoded form. For encoding I have used base64 encoder

For running the secret.yml run the command:

kubectl create –f mysecret.yml –n eks-task

For code refer to the mysecret.yml file

Step 8->

Now we will launch our MYSQL Database that will connect to our Ghost architecture. For this I have created a deploy-mysql.yaml file

For launching the MYSQL run the following command:

Kubectl create –f deploy-mysql.yaml –n eks-task

For code refer to the deploy-mysql.yaml file

Step 9->

Now at last I have launched the Ghost architecture , this will connect to our MYSQL database. For this I have created a deploy-ghost.yaml file

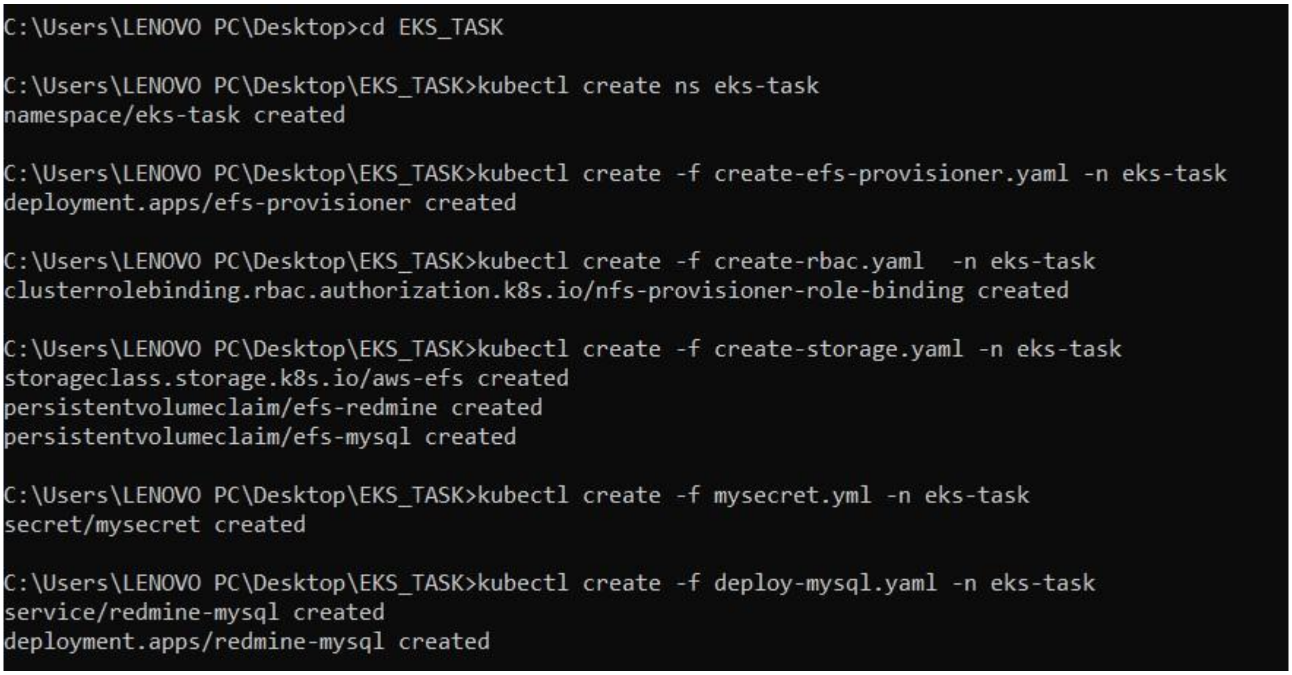
For launching the ghost architecture run the following command:

Kubectl create –f deploy-ghost.yaml –n eks-task

For code refer to the deploy-ghost.yaml file

NOTE: Launch the MYSQL before launching the Ghost architecture , otherwise the architecture would fail

Screenshot for all the above commands :



```
C:\Users\LENOVO PC\Desktop>cd EKS_TASK

C:\Users\LENOVO PC\Desktop\EKS_TASK>kubectl create ns eks-task
namespace/eks-task created

C:\Users\LENOVO PC\Desktop\EKS_TASK>kubectl create -f create-efs-provisioner.yaml -n eks-task
deployment.apps/efs-provisioner created

C:\Users\LENOVO PC\Desktop\EKS_TASK>kubectl create -f create-rbac.yaml  -n eks-task
clusterrolebinding.rbac.authorization.k8s.io/nfs-provisioner-role-binding created

C:\Users\LENOVO PC\Desktop\EKS_TASK>kubectl create -f create-storage.yaml -n eks-task
storageclass.storage.k8s.io/aws-efs created
persistentvolumeclaim/efs-redmine created
persistentvolumeclaim/efs-mysql created

C:\Users\LENOVO PC\Desktop\EKS_TASK>kubectl create -f mysecret.yml -n eks-task
secret/mysecret created

C:\Users\LENOVO PC\Desktop\EKS_TASK>kubectl create -f deploy-mysql.yaml -n eks-task
service/redmine-mysql created
deployment.apps/redmine-mysql created
```

Thank you!!! ;-)