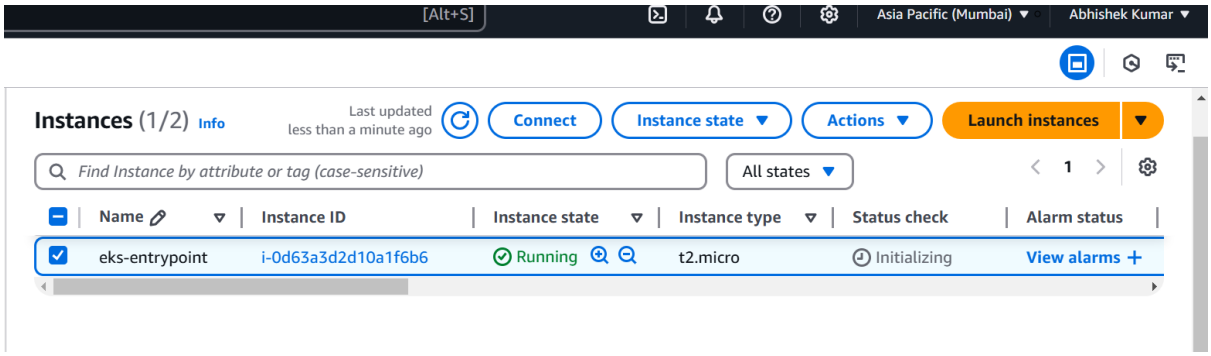
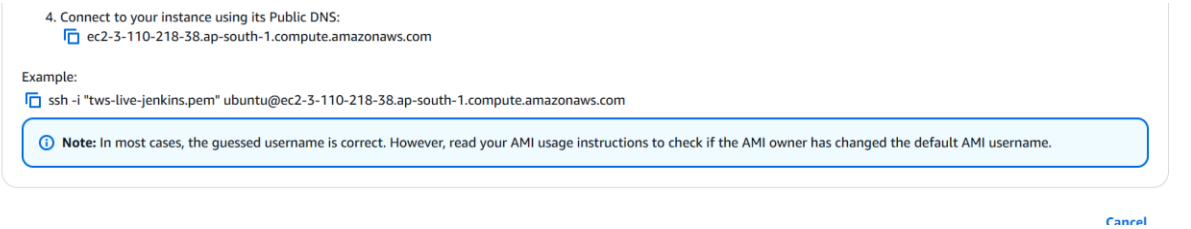
Two-tier Application Deployment on EKS (Elastic Kubernetes Service)

Create an EC2 Instance



Connect to EC2 via SSH




```
HP@DESKTOP-JD85TTO MINGW64 ~/Downloads
$ ssh -i "tws-live-jenkins.pem" ubuntu@ec2-3-110-218-38.ap-south-1.compute.amazonaws.com
The authenticity of host 'ec2-3-110-218-38.ap-south-1.compute.amazonaws.com (3.110.218.38)' can't be established.
ED25519 key fingerprint is SHA256:+snyl9Bl9dx9pPIpbBtHP3+pZ7zd5XDQvqAH+udhgc0.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-110-218-38.ap-south-1.compute.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1021-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Sun Jan 26 10:49:17 UTC 2025

  System load:  0.02              Processes:             105
  Usage of /:   24.9% of 6.71GB   Users logged in:       0
  Memory usage: 20%               IPv4 address for enX0: 172.31.12.203
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-12-203:~$ |
```

Update the Instance

```
ubuntu@ip-172-31-12-203:~$ sudo apt update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:6 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:8 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
```

Install AWS CLI

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
sudo apt install unzip
unzip awscliv2.zip
sudo ./aws/install -i /usr/local/aws-cli -b /usr/local/bin –update

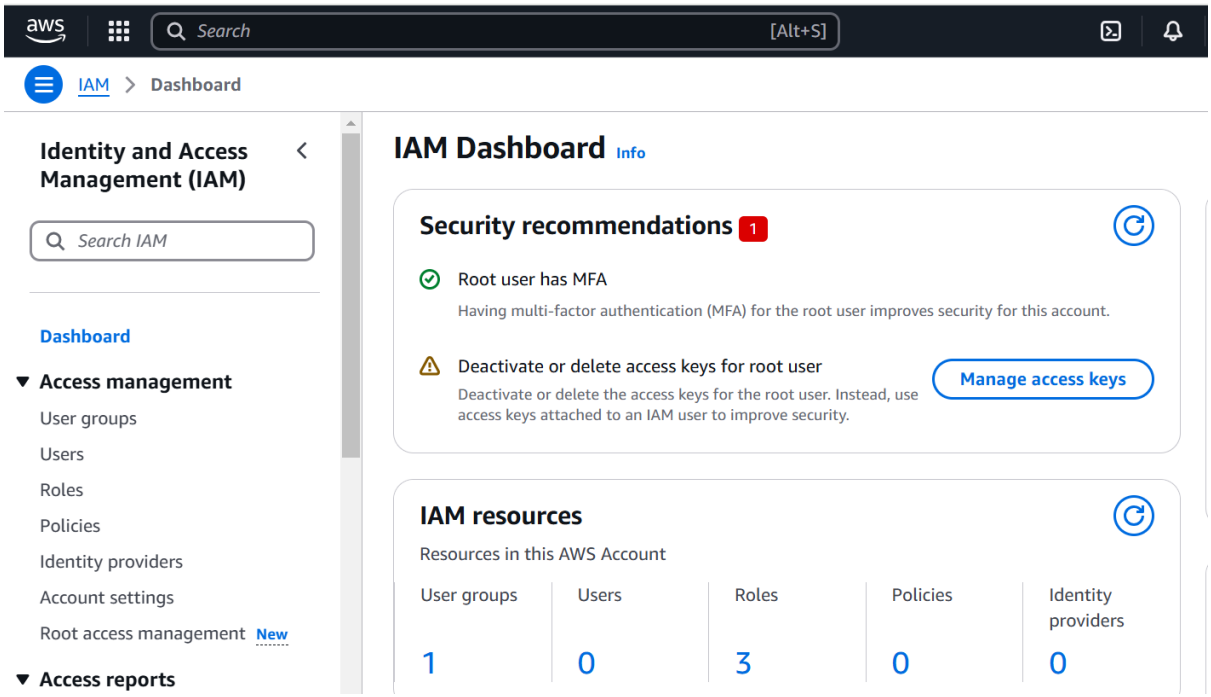
```
ubuntu@ip-172-31-12-203:~$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
sudo apt install unzip
unzip awscliv2.zip
sudo ./aws/install -i /usr/local/aws-cli -b /usr/local/bin --update
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 64.5M  100 64.5M    0     0   100M      0 --:--:-- --:--:-- --:--:--  100M
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  zip
The following NEW packages will be installed:
  unzip
0 upgraded, 1 newly installed, 0 to remove and 27 not upgraded.
Need to get 174 kB of archives.
After this operation, 384 kB of additional disk space will be used.
Get:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 unzip amd64 6.0-28ubuntu4.1 [174 kB]
Fetched 174 kB in 0s (9940 kB/s)
Selecting previously unselected package unzip.
(Reading database ... 70610 files and directories currently installed.)
Preparing to unpack .../unzip_6.0-28ubuntu4.1_amd64.deb ...
Unpacking unzip (6.0-28ubuntu4.1) ...
Setting up unzip (6.0-28ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

It will Install AWS Command line Interface.

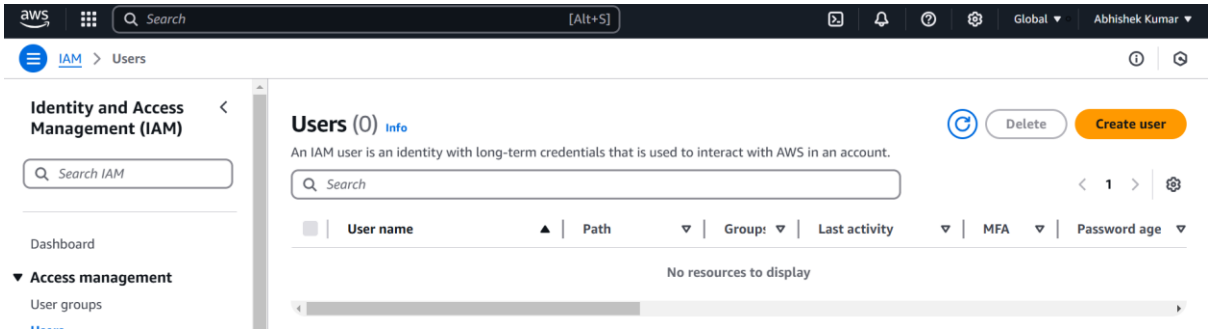
Check the installed version
aws --version

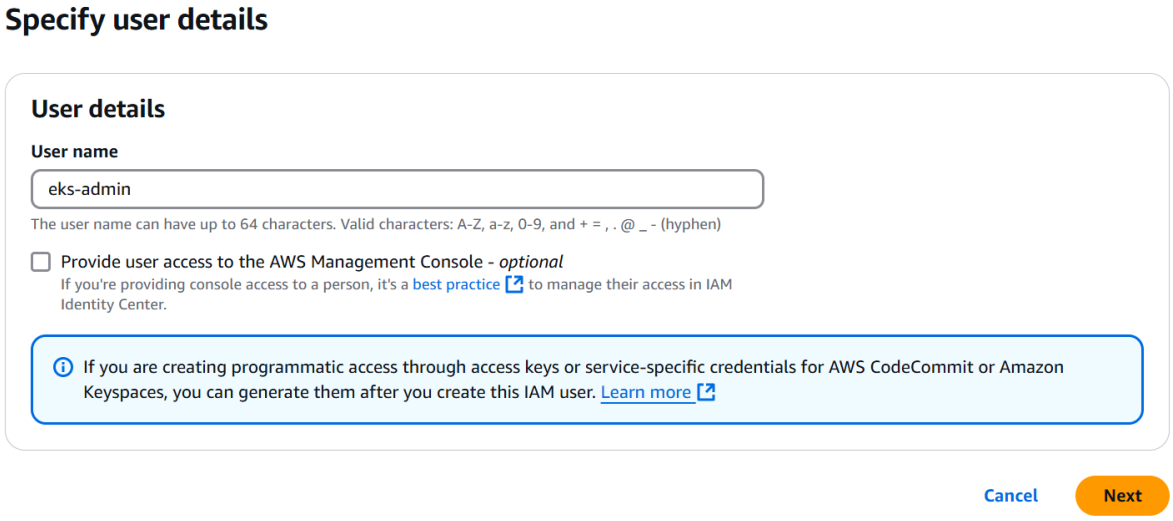
```
ubuntu@ip-172-31-12-203:~$ aws --version
aws-cli/2.23.6 Python/3.12.6 Linux/6.8.0-1021-aws exe/x86_64.ubuntu.24
ubuntu@ip-172-31-12-203:~$
```

Create IAM Role



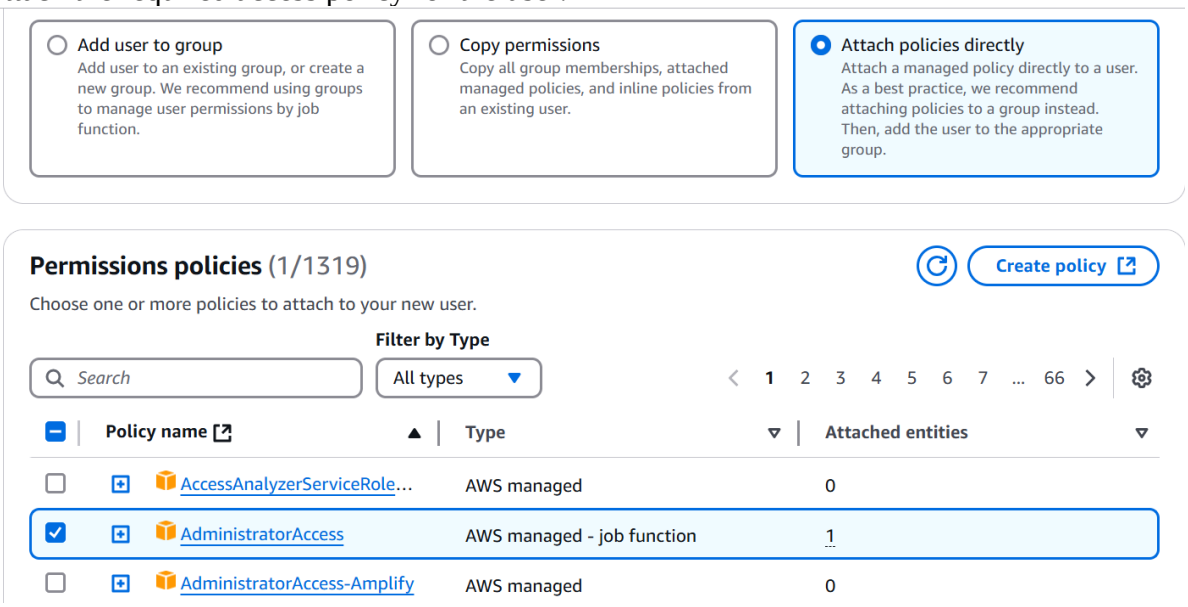
Click on Create User



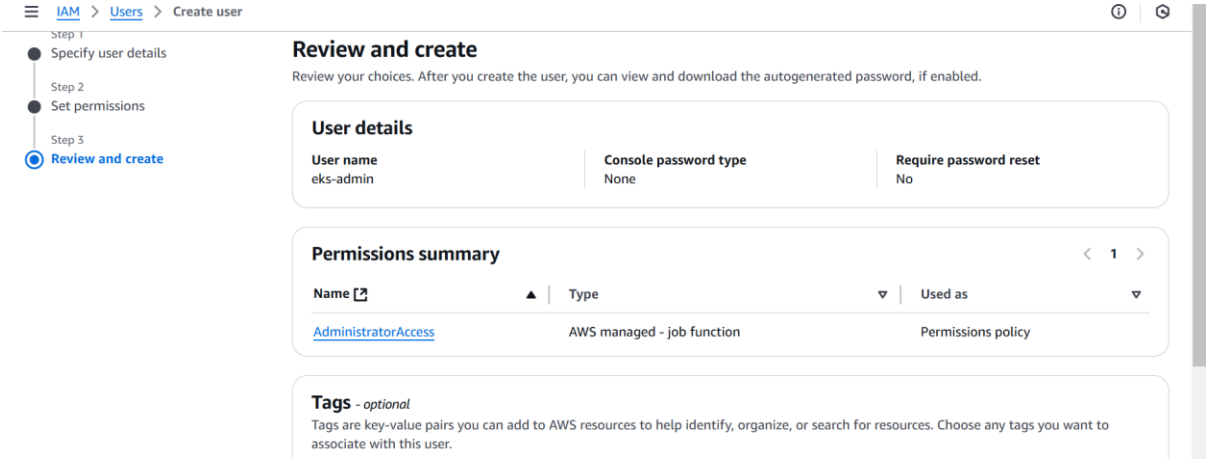


Click on next

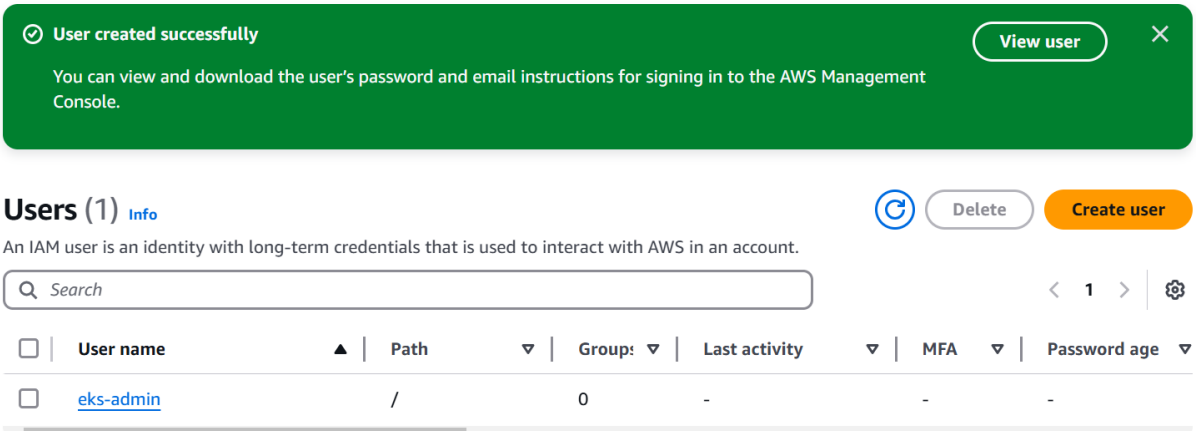
Attach the required access policy for the user.



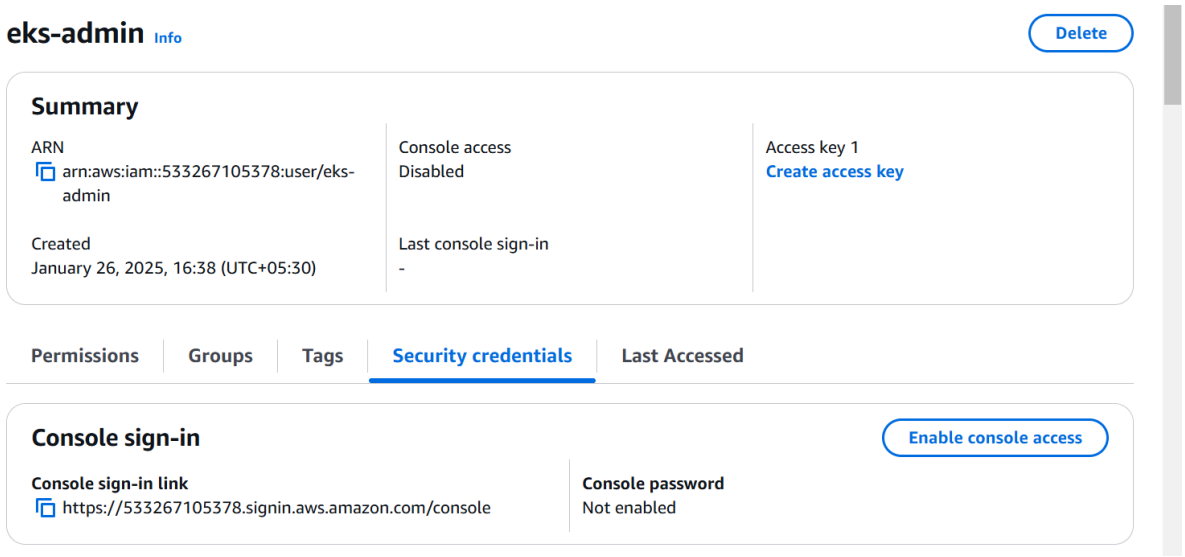
Click on Next



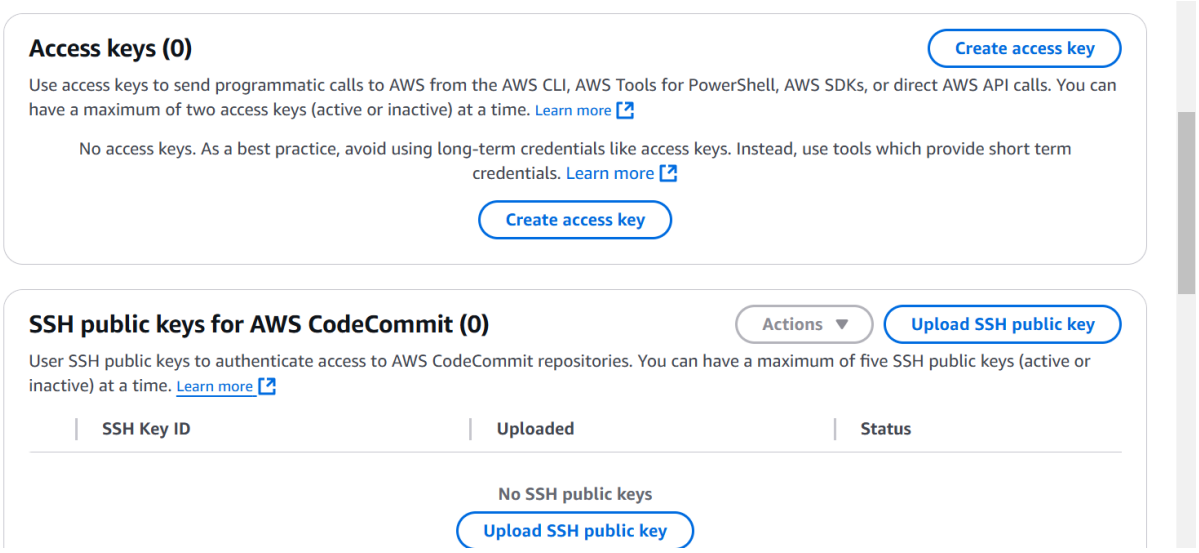
Click on “Create User”



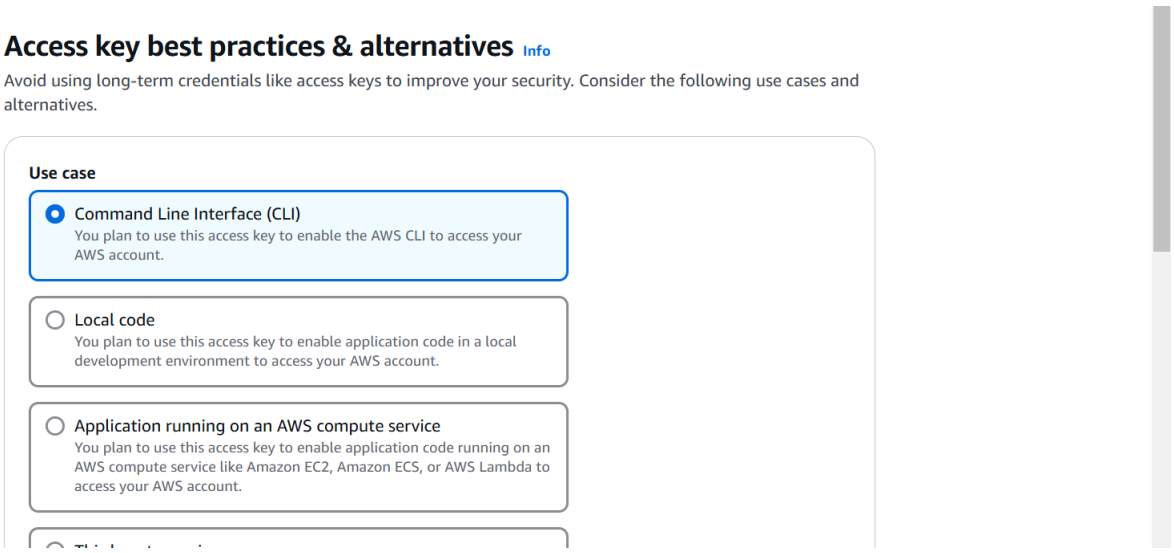
Click on the Created User

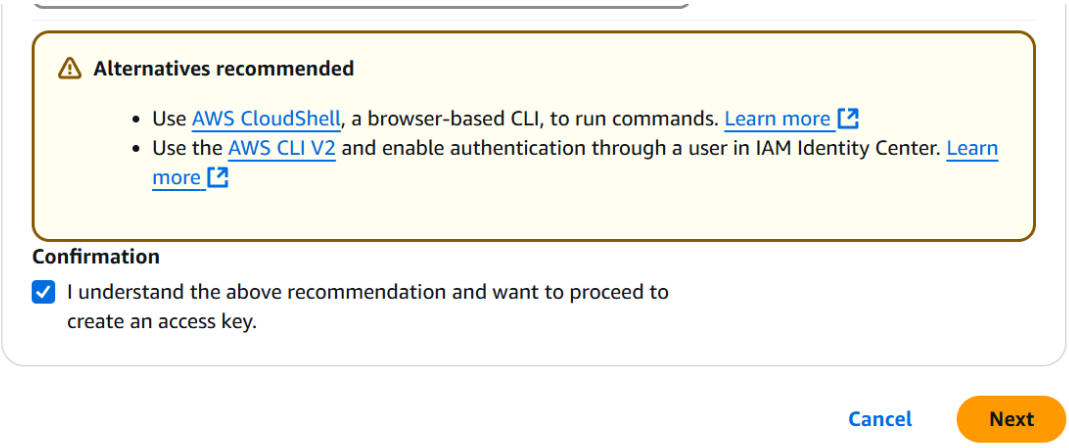


Click on “Security Credentials”



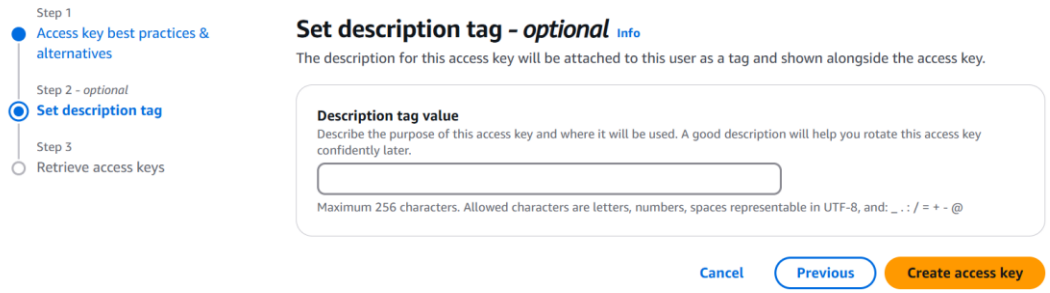
Click on “Create Access Key”

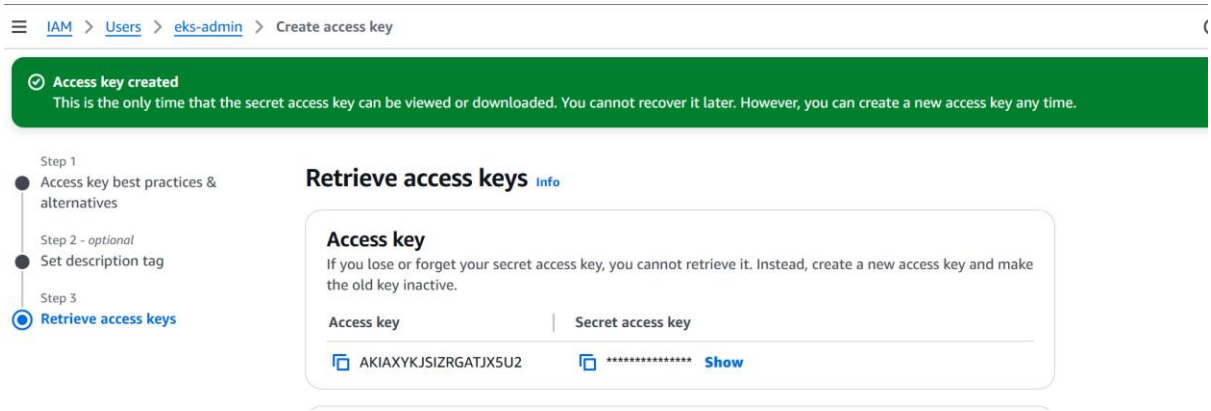




click on Next

Click on “Create Access Key”





Access Key has been Successfully Created.

Copy the Access Key and Secret Key

Go to terminal

Paste the AWS Access Key and Secret Key

```
ubuntu@ip-172-31-12-203: ~
ubuntu@ip-172-31-12-203:~$ aws configure
AWS Access Key ID [None]: AKIAXYKJSIZRGATJX5U2
AWS Secret Access Key [None]: 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
Default region name [None]: ap-south-1
Default output format [None]:
ubuntu@ip-172-31-12-203:~$
```

Install kubectl

curl -o kubectl https://amazon-eks.s3.us-west-2.amazonaws.com/1.19.6/2021-01-05/bin/linux/amd64/kubectl
chmod +x ./kubectl
sudo mv ./kubectl /usr/local/bin
kubectl version --short –client

```
ubuntu@ip-172-31-12-203: ~
ubuntu@ip-172-31-12-203:~$ curl -o kubectl https://amazon-eks.s3.us-west-2.amazonaws.com/1.19.6/2021-01-05/bin/linux/amd64/kubectl
chmod +x ./kubectl
sudo mv ./kubectl /usr/local/bin
kubectl version --short --client
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 57.4M  100 57.4M    0     0  4177k      0  0:00:14  0:00:14 --:--:-- 5054k
Client Version: v1.19.6-eks-49a6c0
ubuntu@ip-172-31-12-203:~$ kubectl version
Client Version: version.Info{Major:"1", Minor:"19+", GitVersion:"v1.19.6-eks-49a6c0", GitCommit:"49a6c0bf091506e7bafcdb1b142351b69363355a", GitTreeState:"clean", BuildDate:"2020-12-23T22:13:28Z", GoVersion:"go1.15.5", Compiler:"gc", Platform:"linux/amd64"}
The connection to the server localhost:8080 was refused - did you specify the right host or port?
ubuntu@ip-172-31-12-203:~$
```

Check version :

- kubectl version

Install eksctl

curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C /tmp
sudo mv /tmp/eksctl /usr/local/bin
eksctl version

```
ubuntu@ip-172-31-12-203: ~
ubuntu@ip-172-31-12-203:~$ curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C /tmp
sudo mv /tmp/eksctl /usr/local/bin
eksctl version
0.202.0
```

To Create Cluster

```
create cluster --name tws-cluster --region ap-south-1 --node-type t3.small --nodes-min 2 --nodes-max 3
```

```
ubuntu@ip-172-31-12-203: ~
ubuntu@ip-172-31-12-203:~$ eksctl create cluster --name tws-cluster --region ap-south-1 --node-type t3.small --nodes-min 2 --nodes-max
3
2025-01-26 11:35:19 [ℹ]  eksctl version 0.202.0
2025-01-26 11:35:19 [ℹ]  using region ap-south-1
2025-01-26 11:35:19 [ℹ]  setting availability zones to [ap-south-1a ap-south-1b ap-south-1c]
2025-01-26 11:35:19 [ℹ]  subnets for ap-south-1a - public:192.168.0.0/19 private:192.168.96.0/19
2025-01-26 11:35:19 [ℹ]  subnets for ap-south-1b - public:192.168.32.0/19 private:192.168.128.0/19
2025-01-26 11:35:19 [ℹ]  subnets for ap-south-1c - public:192.168.64.0/19 private:192.168.160.0/19
2025-01-26 11:35:19 [ℹ]  nodegroup "ng-2b125820" will use "" [AmazonLinux2/1.30]
2025-01-26 11:35:19 [ℹ]  using Kubernetes version 1.30
2025-01-26 11:35:19 [ℹ]  creating EKS cluster "tws-cluster" in "ap-south-1" region with managed nodes
2025-01-26 11:35:19 [ℹ]  will create 2 separate CloudFormation stacks for cluster itself and the initial managed nodegroup
2025-01-26 11:35:19 [ℹ]  if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-stacks --region=ap-sou
th-1 --cluster=tws-cluster'
2025-01-26 11:35:19 [ℹ]  Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess=false} for cluster "tws-c
luster" in "ap-south-1"
2025-01-26 11:35:19 [ℹ]  Cloudwatch logging will not be enabled for cluster "tws-cluster" in "ap-south-1"
2025-01-26 11:35:19 [ℹ]  you can enable it with 'eksctl utils update-cluster-logging --enable-types={SPECIFY-YOUR-LOG-TYPES-HERE (e.g.
all)} --region=ap-south-1 --cluster=tws-cluster'
2025-01-26 11:35:19 [ℹ]  default addons metrics-server, vpc-cni, kube-proxy, coredns were not specified, will install them as EKS addon
s
2025-01-26 11:35:19 [ℹ]
2 sequential tasks: { create cluster control plane "tws-cluster",
    2 sequential sub-tasks: {
        2 sequential sub-tasks: {
            1 task: { create addons },
            wait for control plane to become ready,
        },
        create managed nodegroup "ng-2b125820",
    }
}
2025-01-26 11:35:19 [ℹ]  building cluster stack "eksctl-tws-cluster-cluster"
2025-01-26 11:35:19 [ℹ]  deploying stack "eksctl-tws-cluster-cluster"
```

It will create 2 two nodes but if loads increases, it will scale up to 3 nodes.

Run Manifests

kubectl create namespace two-tier-ns
kubectl apply -f .
Kubectl delete -f .

```
ubuntu@ip-172-31-12-203: ~
ubuntu@ip-172-31-12-203:~$ kubectl get nodes
NAME                                          STATUS   ROLES    AGE   VERSION
ip-192-168-13-129.ap-south-1.compute.internal   Ready    <none>   28m   v1.30.8-eks-aeac579
ip-192-168-91-122.ap-south-1.compute.internal   Ready    <none>   28m   v1.30.8-eks-aeac579
ubuntu@ip-172-31-12-203:~$
```

```
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ cat mysql-secrets.yml
apiVersion: v1
kind: Secret
metadata:
  name: mysql-secret
type: Opaque
data:
  MYSQL_ROOT_PASSWORD: YWRtaW4=
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ kubectl apply -f mysql-secrets.yml
secret/mysql-secret created
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ cat mysql-configmap.yml
apiVersion: v1
kind: ConfigMap
metadata:
  name: mysql-initdb-config
data:
  init.sql: |
    CREATE DATABASE IF NOT EXISTS mydb;
    USE mydb;
    CREATE TABLE messages (id INT AUTO_INCREMENT PRIMARY KEY, message TEXT);
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ kubectl apply -f mysql-configmap.yml
configmap/mysql-initdb-config created
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ cat mysql-deployment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql
  labels:
    app: mysql
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mysql
```

```
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:latest
          env:
            - name: MYSQL_ROOT_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: mysql-secret
                  key: MYSQL_ROOT_PASSWORD
            - name: MYSQL_DATABASE
              value: "mydb"
            - name: MYSQL_USER
              value: "admin"
            - name: MYSQL_PASSWORD
              value: "admin"
          ports:
            - containerPort: 3306
          volumeMounts:
            - name: mysql-initdb
              mountPath: docker-entrypoint-initdb.d
      volumes:
        - name: mysql-initdb
          configMap:
            name: mysql-initdb-config    # Config name
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$
```

ubuntu@ip-172-31-12-203: ~/two-tier-flask-app/eks-manifests

```
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ cat mysql-secrets.yml
apiVersion: v1
kind: Secret
metadata:
  name: mysql-secret
type: Opaque
data:
  MYSQL_ROOT_PASSWORD: YWRtaW4=
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$
```

```
ubuntu@ip-172-31-12-203: ~/two-tier-flask-app/eks-manifests
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ cat mysql-svc.yml
apiVersion: v1
kind: Service
metadata:
  name: mysql
spec:
  selector:
    app: mysql
  ports:
    - port: 3306
      targetPort: 3306
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ kubectl apply -f mysql-svc.yml
service/mysql created
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ s
```

```
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ cat two-tier-app-deployment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: two-tier-app
  labels:
    app: two-tier-app
spec:
  replicas: 1
  selector:
    matchLabels:
      app: two-tier-app
  template:
    metadata:
      labels:
        app: two-tier-app
    spec:
      containers:
        - name: two-tier-app
          image: trainwithshubham/flaskapp:latest
          env:
            - name: MYSQL_HOST
              value: mysql          # this is your mysql's service clusture IP, Make sure to change it with yours
            - name: MYSQL_PASSWORD
              value: "admin"
            - name: MYSQL_USER
              value: "root"
            - name: MYSQL_DB
              value: "mydb"
          ports:
            - containerPort: 5000
          imagePullPolicy: Always
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$
```

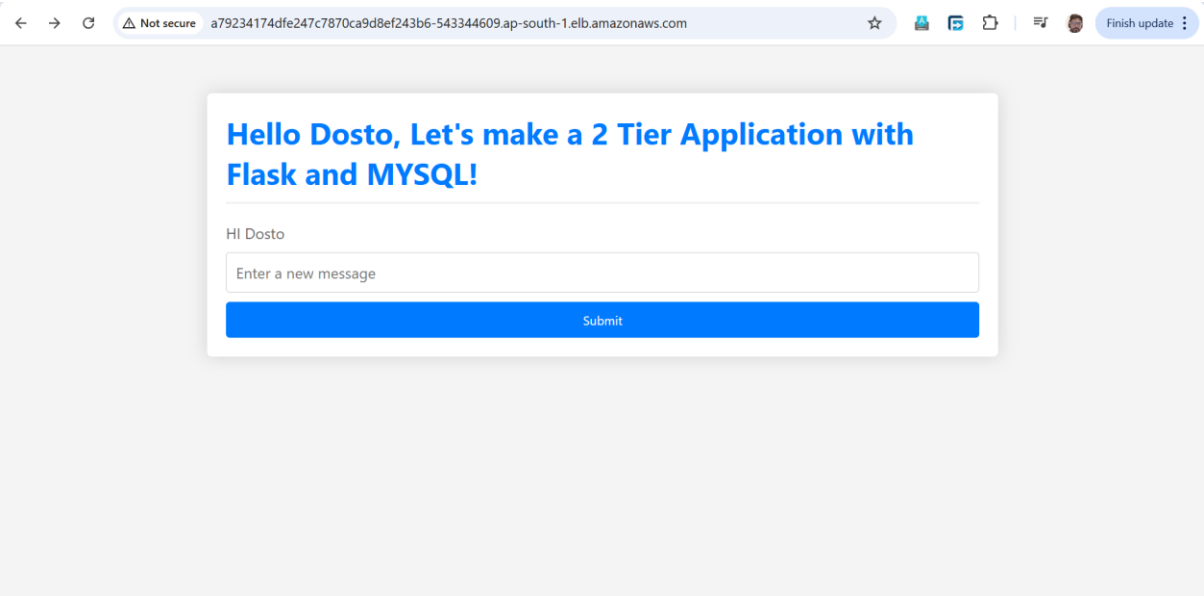
```
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ kubectl apply -f two-tier-app-deployment.yml
deployment.apps/two-tier-app created
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ cat two-tier-app-svc.yml
apiVersion: v1
kind: Service
metadata:
  name: two-tier-app-service
spec:
  selector:
    app: two-tier-app
  type: LoadBalancer
  ports:
    - protocol: TCP
      port: 80
      targetPort: 5000

ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ kubectl apply -f two-tier-app-svc.yml
service/two-tier-app-service created
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$
```

Kubectl get all

```
ubuntu@ip-172-31-12-203: ~/two-tier-flask-app/eks-manifests
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ kubectl get all
NAME                                  READY   STATUS    RESTARTS   AGE
pod/mysql-c869ddbbc-hfc7s             1/1     Running   0          91s
pod/two-tier-app-784fcf9f5d-2kl7w     1/1     Running   0          22m

NAME                            TYPE           CLUSTER-IP     EXTERNAL-IP
                 PORT(S)        AGE
service/kubernetes              ClusterIP      10.100.0.1     <none>
                 443/TCP        112m
service/mysql                   ClusterIP      10.100.43.78   <none>
                 3306/TCP       25m
service/two-tier-app-service    LoadBalancer   10.100.93.56   a79234174dfe247c7870ca9d8ef243b6-543344609.ap-south-1.elb
.amazonaws.com   80:30409/TCP   21m

NAME                            READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/mysql           1/1     1            1           91s
deployment.apps/two-tier-app    1/1     1            1           22m

NAME                                       DESIRED   CURRENT   READY   AGE
replicaset.apps/mysql-c869ddbbc            1         1         1       91s
replicaset.apps/two-tier-app-784fcf9f5d    1         1         1       22m
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$
```

To check whether the secrets reached to configmap

Describe the Pod: `kubectl describe pod mysql-c869ddbbc-hfc7s`



To delete a Cluster

`eksctl delete cluster --name tws-cluster --region ap-south-1`

```
ubuntu@ip-172-31-12-203: ~/two-tier-flask-app/eks-manifests
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$ eksctl delete cluster --name tws-cluster --region ap-south
-1
2025-01-26 13:46:20 [ℹ]  deleting EKS cluster "tws-cluster"
2025-01-26 13:46:20 [ℹ]  will drain 0 unmanaged nodegroup(s) in cluster "tws-cluster"
2025-01-26 13:46:20 [ℹ]  starting parallel draining, max in-flight of 1
2025-01-26 13:46:20 [ℹ]  deleted 0 Fargate profile(s)
2025-01-26 13:46:20 [✓]  kubeconfig has been updated
2025-01-26 13:46:20 [ℹ]  cleaning up AWS load balancers created by Kubernetes objects of Kind Service or Ingress
2025-01-26 13:46:57 [ℹ]
2 sequential tasks: { delete nodegroup "ng-2b125820", delete cluster control plane "tws-cluster" [async]
}
2025-01-26 13:46:57 [ℹ]  will delete stack "eksctl-tws-cluster-nodegroup-ng-2b125820"
2025-01-26 13:46:57 [ℹ]  waiting for stack "eksctl-tws-cluster-nodegroup-ng-2b125820" to get deleted
2025-01-26 13:46:57 [ℹ]  waiting for CloudFormation stack "eksctl-tws-cluster-nodegroup-ng-2b125820"
2025-01-26 13:47:27 [ℹ]  waiting for CloudFormation stack "eksctl-tws-cluster-nodegroup-ng-2b125820"
2025-01-26 13:48:05 [ℹ]  waiting for CloudFormation stack "eksctl-tws-cluster-nodegroup-ng-2b125820"
2025-01-26 13:48:45 [ℹ]  waiting for CloudFormation stack "eksctl-tws-cluster-nodegroup-ng-2b125820"
2025-01-26 13:49:38 [ℹ]  waiting for CloudFormation stack "eksctl-tws-cluster-nodegroup-ng-2b125820"
2025-01-26 13:50:32 [ℹ]  waiting for CloudFormation stack "eksctl-tws-cluster-nodegroup-ng-2b125820"
2025-01-26 13:51:53 [ℹ]  waiting for CloudFormation stack "eksctl-tws-cluster-nodegroup-ng-2b125820"
2025-01-26 13:53:41 [ℹ]  waiting for CloudFormation stack "eksctl-tws-cluster-nodegroup-ng-2b125820"
2025-01-26 13:55:39 [ℹ]  waiting for CloudFormation stack "eksctl-tws-cluster-nodegroup-ng-2b125820"
2025-01-26 13:55:39 [ℹ]  will delete stack "eksctl-tws-cluster-cluster"
2025-01-26 13:55:39 [✓]  all cluster resources were deleted
ubuntu@ip-172-31-12-203:~/two-tier-flask-app/eks-manifests$
```