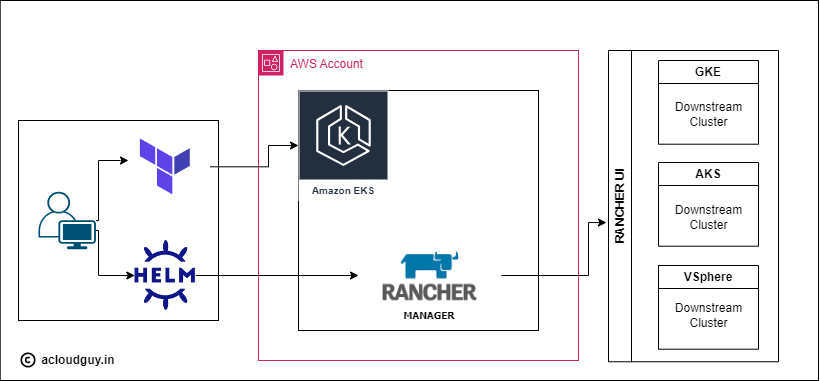
**Deploy Rancher on AWS EKS using Terraform & Helm Charts**



**Project Overview**

This project revolves around AWS EKS where we deploy Rancher (platform for Kubernetes management). Rancher is a Kubernetes management tool to deploy and run clusters anywhere and on any provider. Amazon EKS & Rancher is very vital combination when it comes to managing multi-cluster Kubernetes workloads from Single Dashboard

**SOLUTIONS ARCHITECTURE OVERVIEW**



**First Let’s understand the real world use case :**

* **Hybrid Cloud Management for a Global E-Commerce Platform:**Imagine a large e-commerce platform that serves customers globally. The platform’s infrastructure spans multiple regions to ensure low-latency access and high availability. The architecture includes microservices running on Kubernetes clusters managed by AWS EKS. Additionally, the company has on-premises data centers hosting legacy applications. Rancher provides a unified dashboard for managing EKS clusters spread across various regions.
* **Finance: Multi-Tiered Application Deployment:**A financial institution is migrating its legacy monolithic applications to microservices architecture on AWS EKS. The institution operates in multiple regions and requires secure and efficient deployment of multi-tiered applications. Rancher simplifies the deployment of microservices across EKS clusters in different regions.
* **Retail: Seasonal Application Scaling:**A retail chain experiences significant fluctuations in website traffic during holiday seasons. They need a solution to dynamically scale their applications on EKS to handle increased demand. Rancher enables automated scaling of applications based on predefined policies. Provides visibility into application performance, helping to optimize resource allocation during peak times.
* **Manufacturing: Edge Computing for IoT Devices:**A manufacturing company utilizes IoT devices across its facilities to monitor and optimize production processes. They need a solution to manage Kubernetes clusters at the edge for real-time data processing. Rancher supports the deployment of Kubernetes clusters at the edge, close to IoT devices. It also enables centralized management of edge clusters for easier monitoring and updates.
* **Media and Entertainment: Content Delivery Optimization:**A media streaming service operates globally and needs to optimize content delivery for users. They want to deploy and manage Kubernetes clusters efficiently to ensure low-latency streaming. It Integrates with AWS services like Amazon CloudFront for efficient content caching. Allows for easy scaling and updating of streaming applications across clusters. Facilitates the deployment of edge-native applications to process data locally.

These diverse use cases demonstrate the versatility of Rancher on **AWS EKS** in addressing industry-specific challenges and enhancing the management of Kubernetes clusters in various contexts. **Rancher** simplifies operations, enhances security, and provides a unified platform for managing the hybrid cloud environment.

***Prerequisite —***

1. AWS ACCOUNT with admin privileges.
2. AWS EC2 Instance (Bastion Host)
3. [Terraform](https://developer.hashicorp.com/terraform/tutorials/aws-get-started/install-cli)
4. [HELM](https://helm.sh/docs/intro/install/)
5. [KUBECTL](https://kubernetes.io/docs/tasks/tools/)

***AWS Services Usage —***

* AWS EKS
* AWS IAM
* AWS EC2
* AWS ELB
* AWS VPC

STEP BY STEP GUIDE ***-***

STEP 1 : Clone the repo & check the versions of installed tools

* Login to AWS EC2 instance (Bastion Host).
* Install tools — Terraform, helm, kubectl, aws cli.
* Check versions : aws version, kubectl version, helm version.

Now clone the Git Repo :  Prerequisite

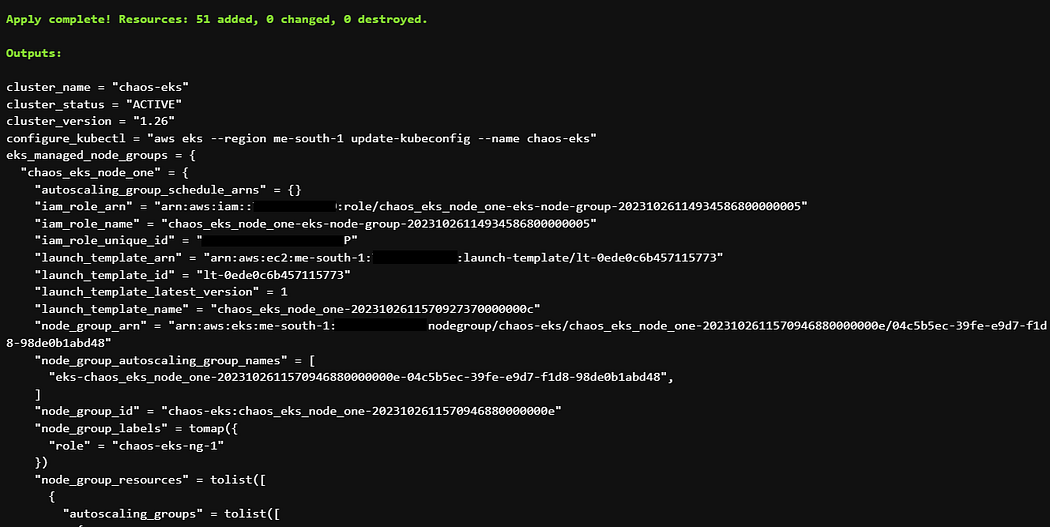
* AWS ACCOUNT with admin privileges.
* AWS EC2 Instance (Bastion Host)
* Terraform
* HELM
* KUBECTL
* AWS Services Usage —
* AWS EKS
* AWS IAM
* AWS EC2
* AWS ELB
* AWS VPC
* STEP BY STEP GUIDE -
* STEP 1 : Clone the repo & check the versions of installed tools
* Login to AWS EC2 instance (Bastion Host).
* Install tools — Terraform, helm, kubectl, aws cli.
* Check versions : aws version, kubectl version, helm version.
* Now clone the Git Repo :

https://github.com/Venn1991/Rancher-EKS.git

* AWS ACCOUNT with admin privileges.
* AWS EC2 Instance (Bastion Host)
* Terraform
* HELM
* KUBECTL
* AWS Services Usage —
* AWS EKS
* AWS IAM
* AWS EC2
* AWS ELB
* AWS VPC
* STEP BY STEP GUIDE -
* STEP 1 : Clone the repo & check the versions of installed tools
* Login to AWS EC2 instance (Bastion Host).
* Install tools — Terraform, helm, kubectl, aws cli.
* Check versions : aws version, kubectl version, helm version.
* Now clone the Git Repo : Terraform Repo Link
* Give Star & Follow me on GitHub.
* Repo has 3 files — main.tf, variables.tf & terraform.tf
* STEP 2 : Deploying AWS resources through Terraform.
* Now go to the folder location & run below commands.
* Terraform init
* Terraform validate
* Terraform plan
* Terraform apply
* This will deploy infrastructure resources in AWS. It will take around 15–20 mins to get it deployed.
* It will have 1 EKS master cluster & 2 worker nodes (ec2 spot instances) attached to the aws EKS master cluster.
* STEP 3 : Add Helm Repositories
* helm repo add devpro https://devpro.github.io/helm-charts
* This repository contains Helm charts to build clusters with all components running in containers.
* helm repo add jetstack https://charts.jetstack.io
* helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
* helm repo add rancher-latest https://releases.rancher.com/server-charts/latest
* helm repo update
* helm repo list
* Terraform Repo Link
* Give Star & Follow me on GitHub.
* Repo has 3 files — main.tf, variables.tf & terraform.tf
* STEP 2 : Deploying AWS resources through Terraform.
* Now go to the folder location & run below commands.
* Terraform init
* Terraform validate
* Terraform plan
* Terraform apply
* This will deploy infrastructure resources in AWS. It will take around 15–20 mins to get it deployed.
* It will have 1 EKS master cluster & 2 worker nodes (ec2 spot instances) attached to the aws EKS master cluster.
* STEP 3 : Add Helm Repositories
* helm repo add devpro https://devpro.github.io/helm-charts
* This repository contains Helm charts to build clusters with all components running in containers.
* helm repo add jetstack https://charts.jetstack.io
* helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
* helm repo add rancher-latest https://releases.rancher.com/server-charts/latest
* helm repo update
* helm repo list
* Give Star & Follow me on [GitHub.](https://github.com/Kunal-Shah107)
* Repo has 3 files — main.tf, variables.tf & terraform.tf

**STEP 2 : Deploying AWS resources through Terraform.**

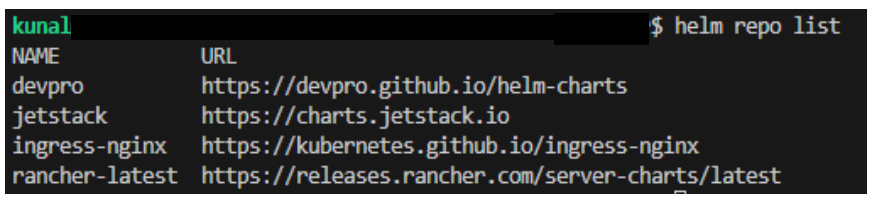
* Now go to the folder location & run below commands.
* **Terraform init**
* **Terraform validate**
* **Terraform plan**
* **Terraform apply**



* This will deploy infrastructure resources in AWS. It will take around 15–20 mins to get it deployed.
* It will have 1 EKS master cluster & 2 worker nodes (ec2 spot instances) attached to the aws EKS master cluster.

**STEP 3 : Add Helm Repositories**

* **helm repo add devpro**[**https://devpro.github.io/helm-charts**](https://devpro.github.io/helm-charts)
* This repository contains Helm charts to build clusters with all components running in containers.
* helm repo add jetstack [https://charts.jetstack.io](https://charts.jetstack.io/)
* helm repo add ingress-nginx <https://kubernetes.github.io/ingress-nginx>
* helm repo add rancher-latest <https://releases.rancher.com/server-charts/latest>
* helm repo update
* helm repo list



**STEP 4 : Access the EKS Cluster through kubectl**

* Run below mentioned commands from EC2 Bastion:
* **aws eks — region me-south-1 update-kubeconfig — name <eks\_cluster\_name>**
* **kubectl get nodes**
* **kubectl get all -A**
* **TIP**— If you face any permission issue then change permission for ~/.kube/config
* This confirms that **Cluster on EKS is ready with requested worker nodes.**

**STEP 5 : Ngnix Ingress Installation (exposing to Internet)**

* Run below mentioned command :

*helm upgrade — install \*

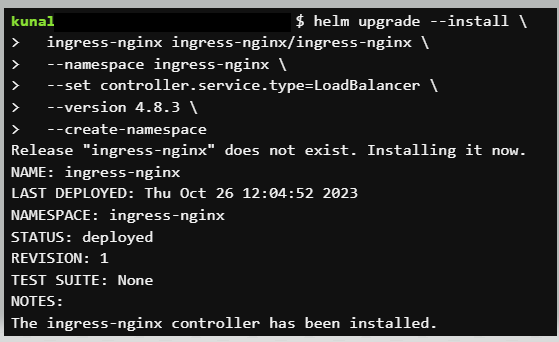
*ingress-nginx ingress-nginx/ingress-nginx \*

*— namespace ingress-nginx \*

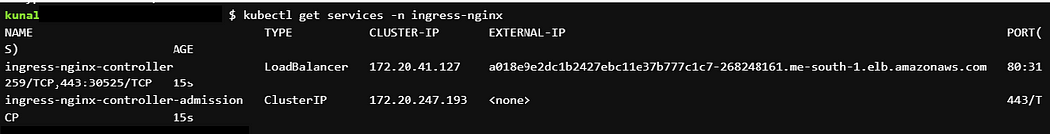
*— set controller.service.type=LoadBalancer \*

*— version 4.8.3 \*

*— create-namespace*



* **Check the services:**kubectl get services -n ingress-nginx
* Copy & save Loadbalancer DNS ( It will used in step 7)



**STEP 6 : Install Certificates manager.**

* Set env variables
* **CERT\_MANAGER\_VERSION=v1.12.4**
* **RANCHER\_VERSION=2.7.6**
* kubectl apply -f [https://github.com/cert-manager/cert-manager/releases/download/${CERT\_MANAGER\_VERSION}/cert-manager.crds.yaml](https://github.com/cert-manager/cert-manager/releases/download/$%7BCERT_MANAGER_VERSION%7D/cert-manager.crds.yaml)

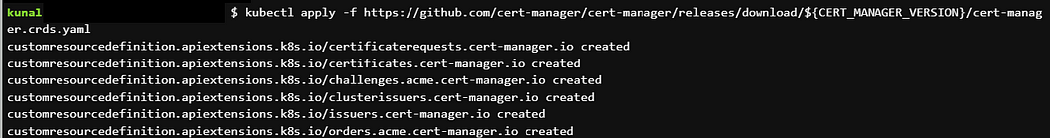
*helm upgrade — install cert-manager jetstack/cert-manager \*

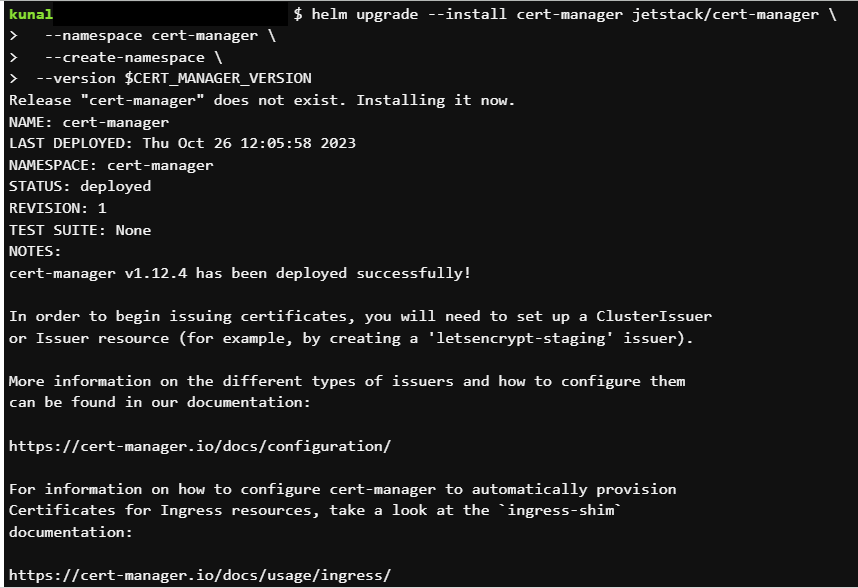
*— namespace cert-manager \*

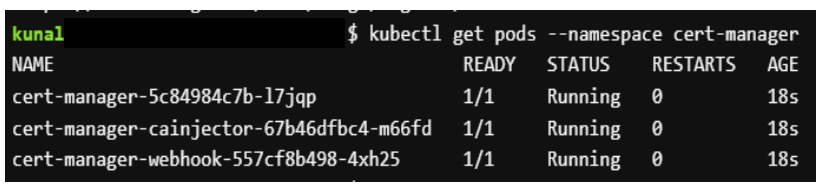
*— create-namespace \*

*— version $CERT\_MANAGER\_VERSION*

* **Check the pods :**kubectl get pods -namespace cert-manager ( 3 pods should be in Running status)







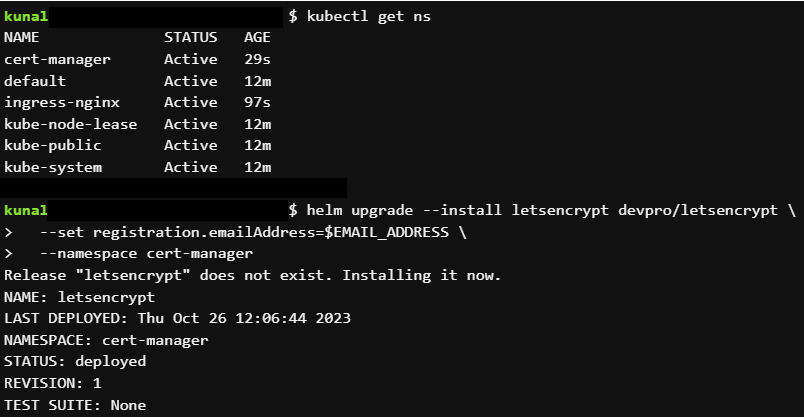
* Lets have new certificate from **letsencrypt**

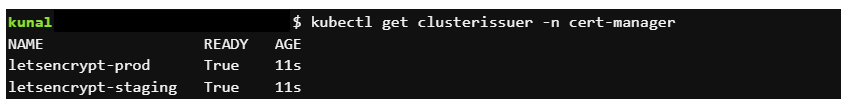
*helm upgrade — install letsencrypt devpro/letsencrypt \*

*— set registration.emailAddress=$EMAIL\_ADDRESS \*

*— namespace cert-manager*

* **Now run this command :**kubectl get clusterissuer -n cert-manager ( 2 Cluster issuers should be True)

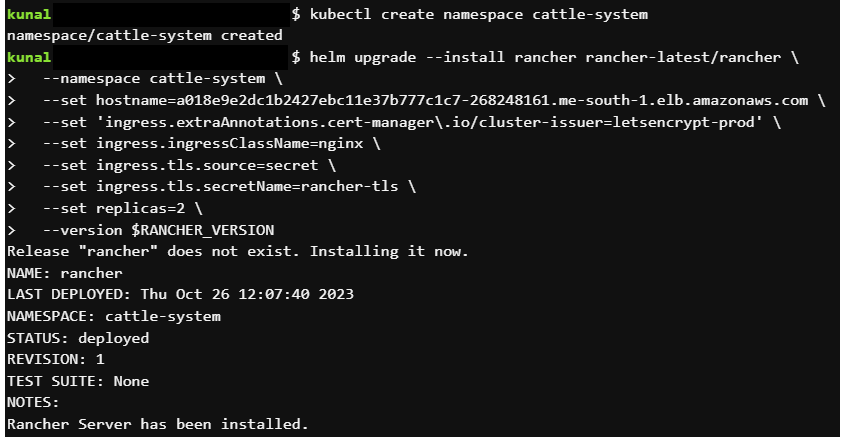




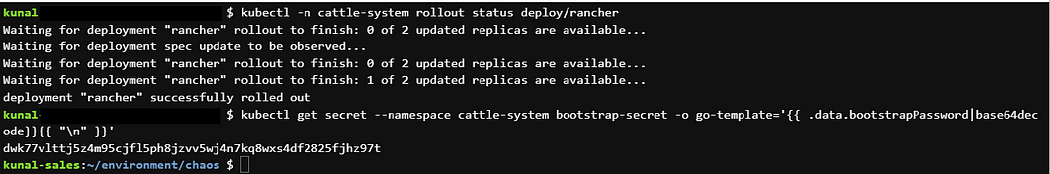
**STEP 7 : Install Rancher**

* kubectl create namespace cattle-system

*helm upgrade — install rancher rancher-latest/rancher \  
— namespace cattle-system \  
— set hostname=<LOAD\_BALANCER\_DNS> \  
— set ‘ingress.extraAnnotations.cert-manager\.io/cluster-issuer=letsencrypt-prod’ \  
— set ingress.ingressClassName=nginx \  
— set ingress.tls.source=secret \  
— set ingress.tls.secretName=rancher-tls \  
— set replicas=2 \  
— version $RANCHER\_VERSION*

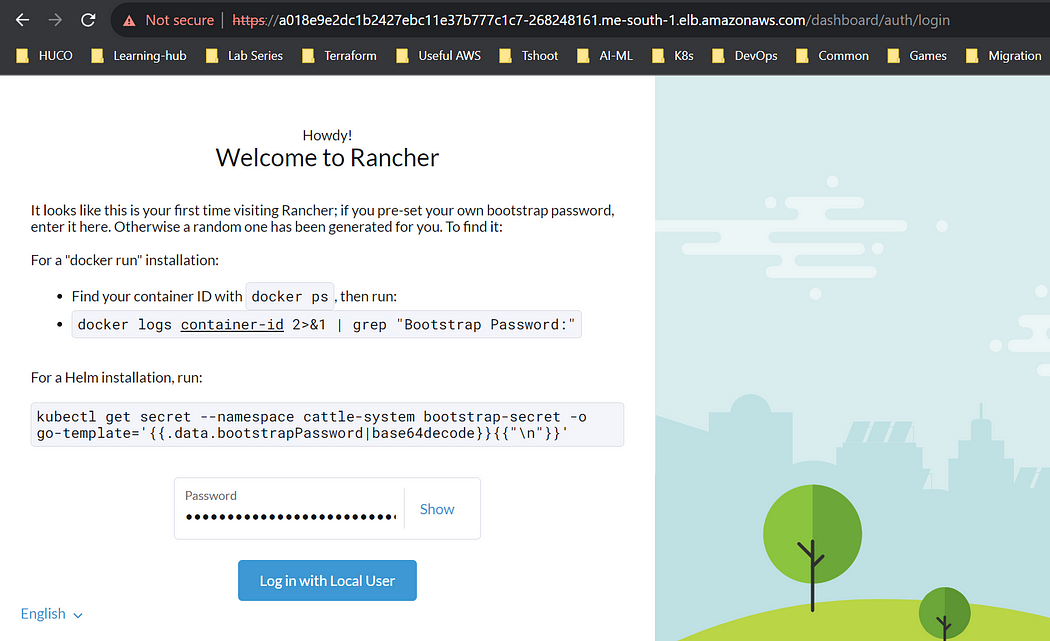


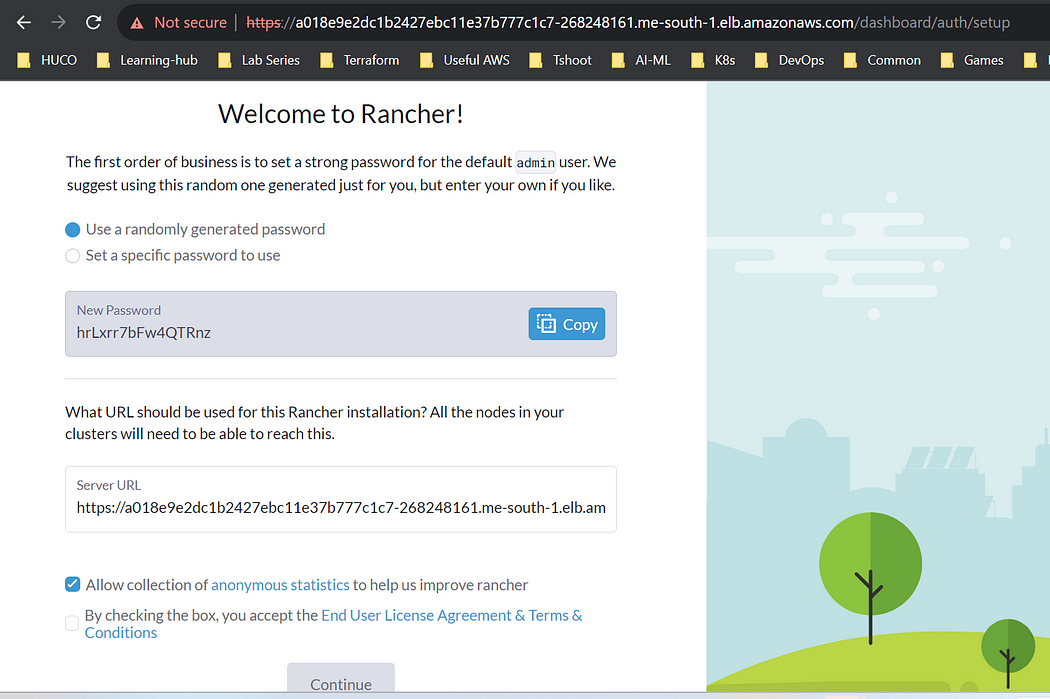
* Check the status of installation & wait for it to complete.
* kubectl -n cattle-system rollout status deploy/rancher
* kubectl get secret — namespace cattle-system bootstrap-secret -o go-template=’{{ .data.bootstrapPassword|base64decode}}{{ “\n” }}’

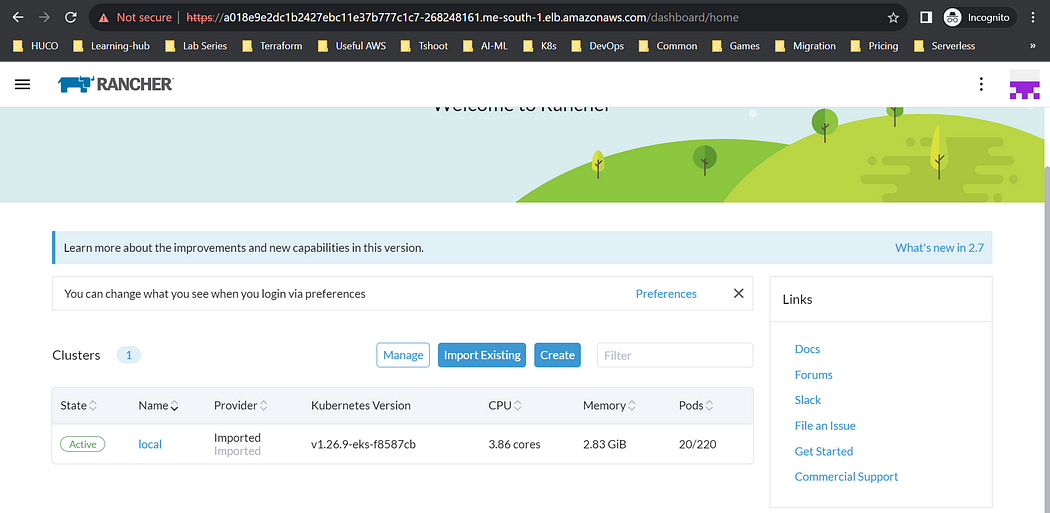


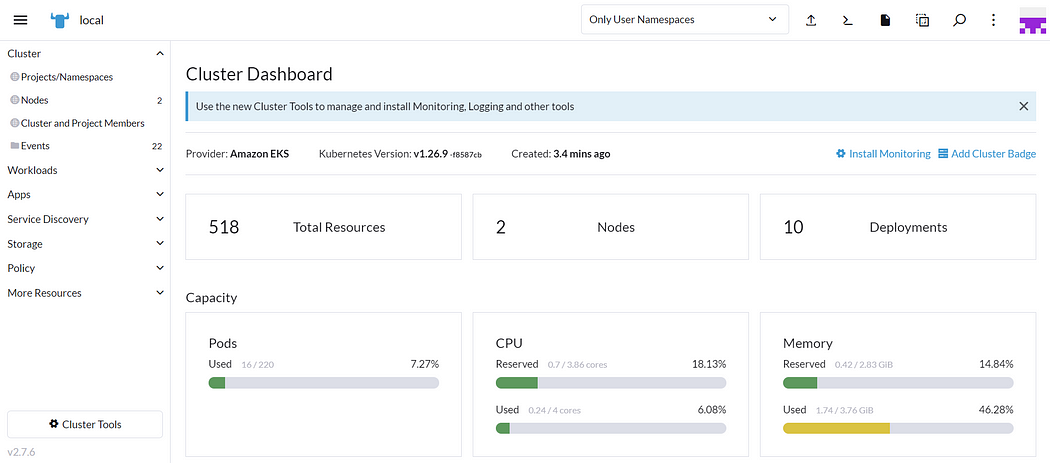
**STEP 8 : Accessing the Rancher UI**

* Copy Load balancer DNS URL & Paste on browser for Initial setup of Rancher.
* You will be asked for password ( copied earlier )
* Hit Log in with Local User.
* Define the admin password, check the box and click on “Continue”. (store it)
* Finally, Rancher is running and you can explore “local” cluster.

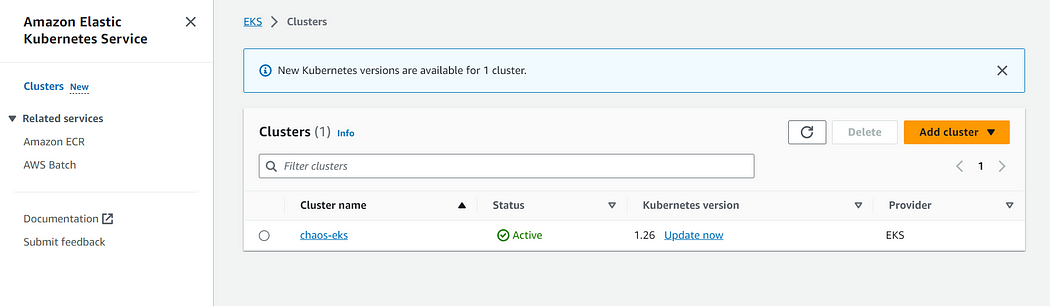


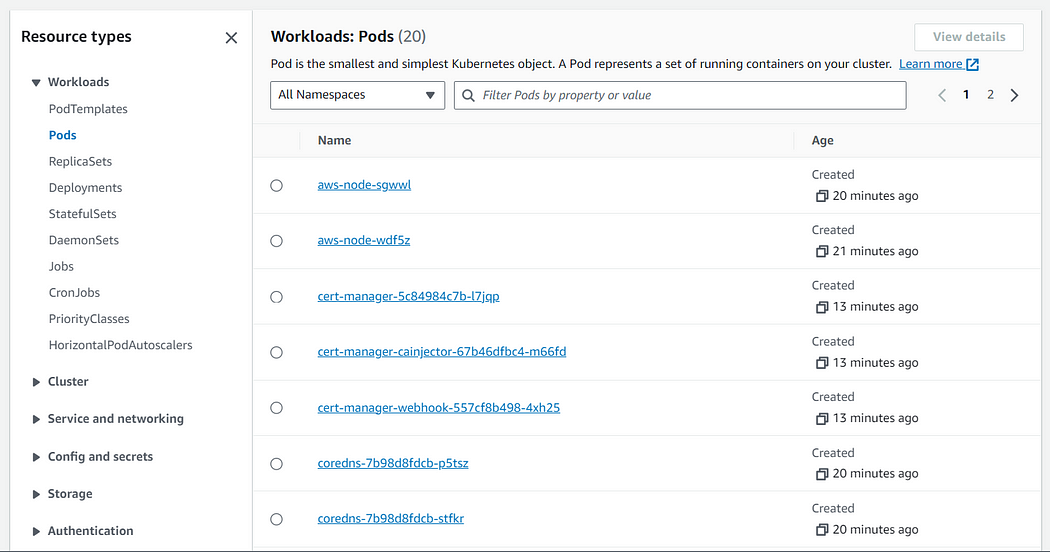


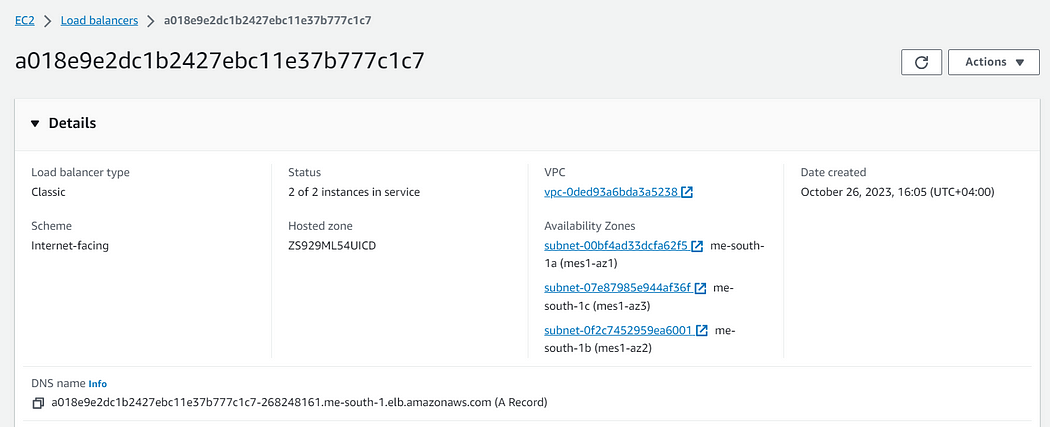




**STEP 9 : Check AWS EKS Console**







**STEP 10 : Decommission**

* Run below command to destroy all resources.
* **kubectl delete ns cert-manager**
* **kubectl delete ns ingress-nginx**
* **terraform destroy — auto-approve**