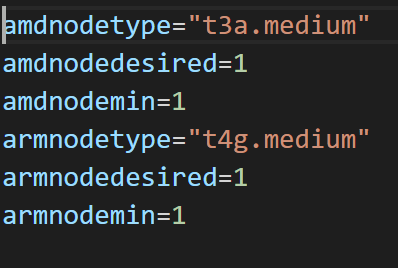
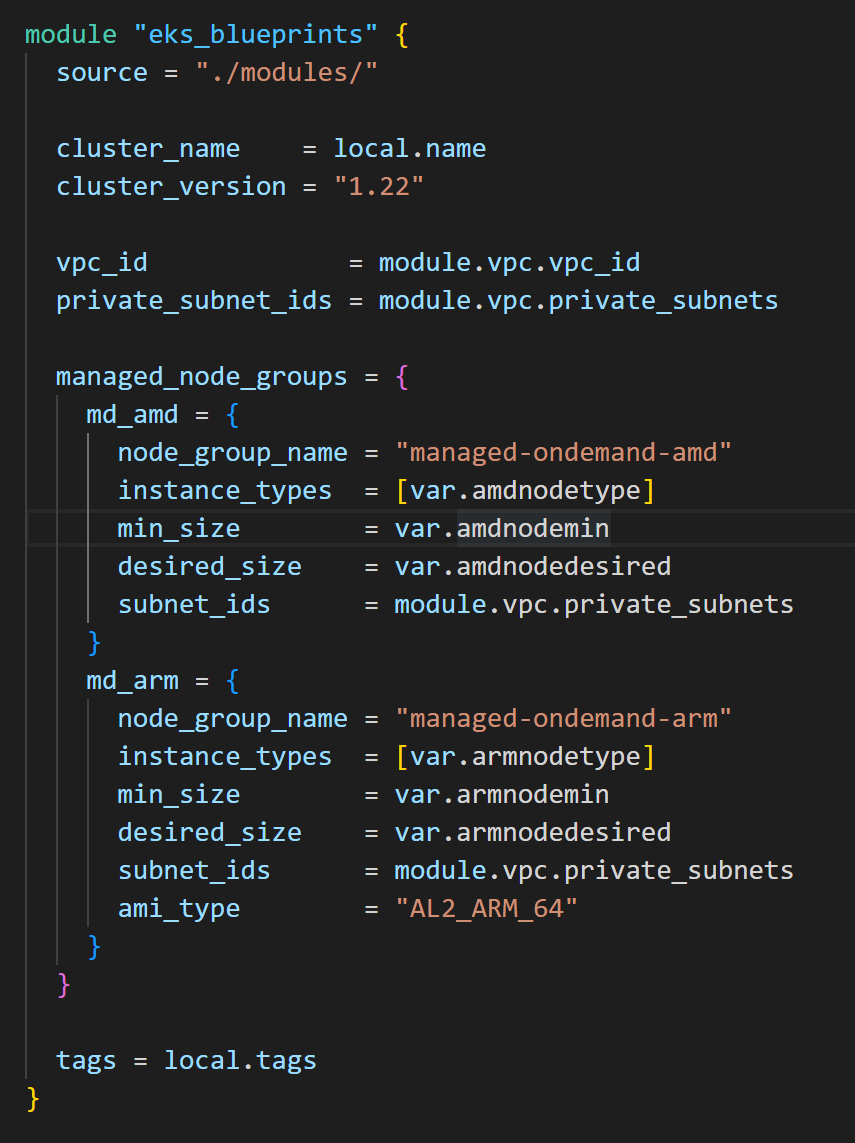
Assessment Task

This document provides the explanation, code snippets and screenshot of UI of all the task provided in the assessment.

# Creation of EKS Cluster with 2 node groups (amd64 and arm64)

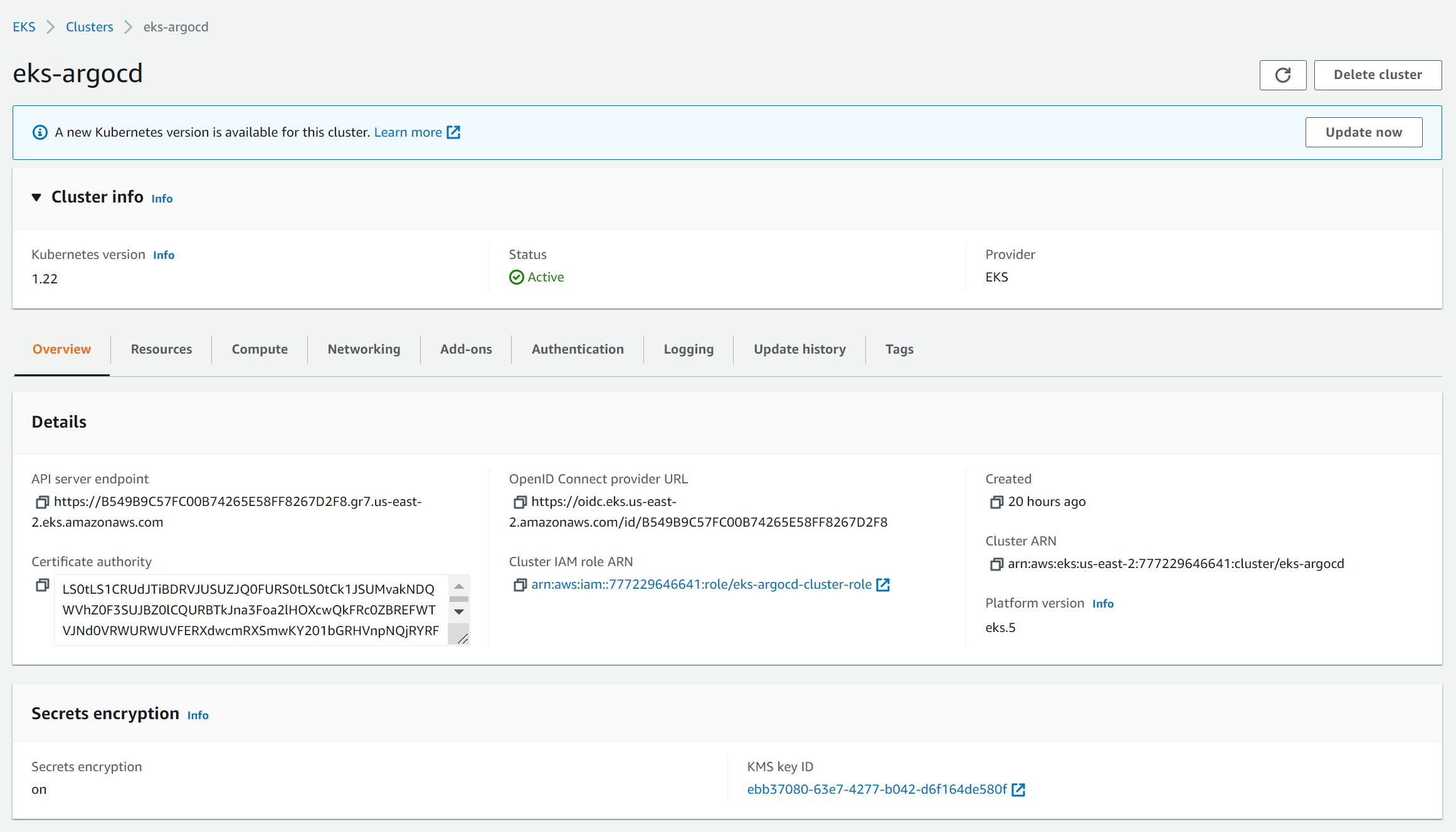
EKS cluster is created using terraform with the help of module eks\_blueprint. Two managed nodegroups has been created for the cluster with the help of this repo as seen in the screenshot. Instance type, minimum instance count and desired instance count can be selected in terraform.tfvars file.

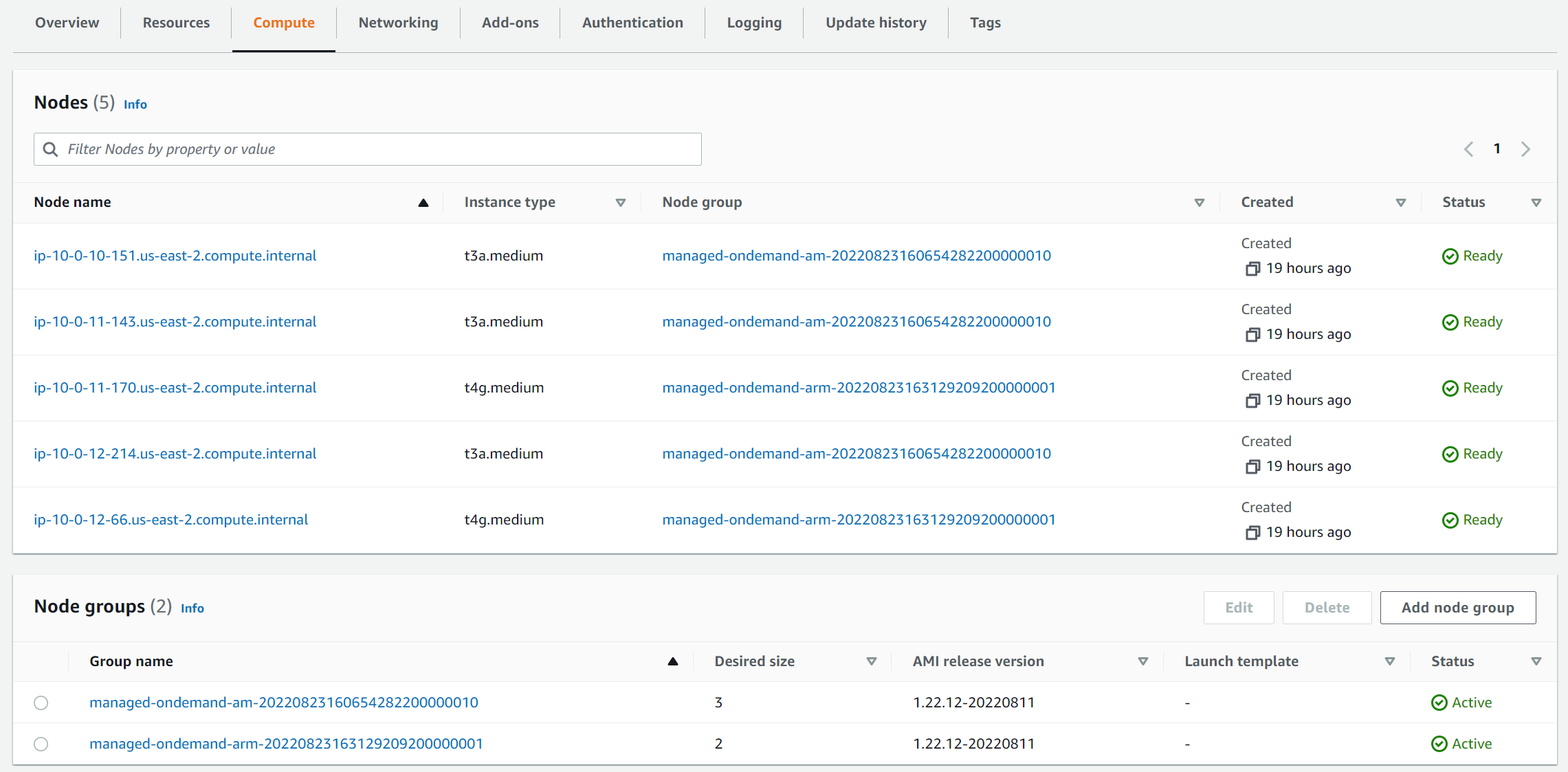


To implement full automation in this script, it creates a new VPC for the EKS cluster in which all the resources are deployed by the script. It creates 3 private subnets, 3 public subnets and a single NAT Gateway to save cost.



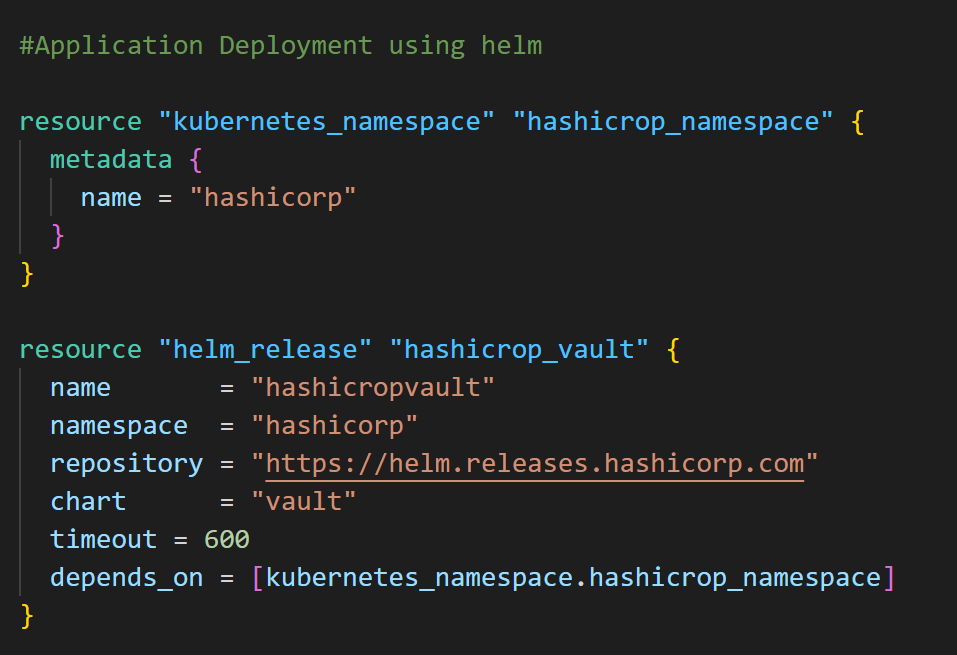
**Screenshots of EKS Cluster and Node Groups**

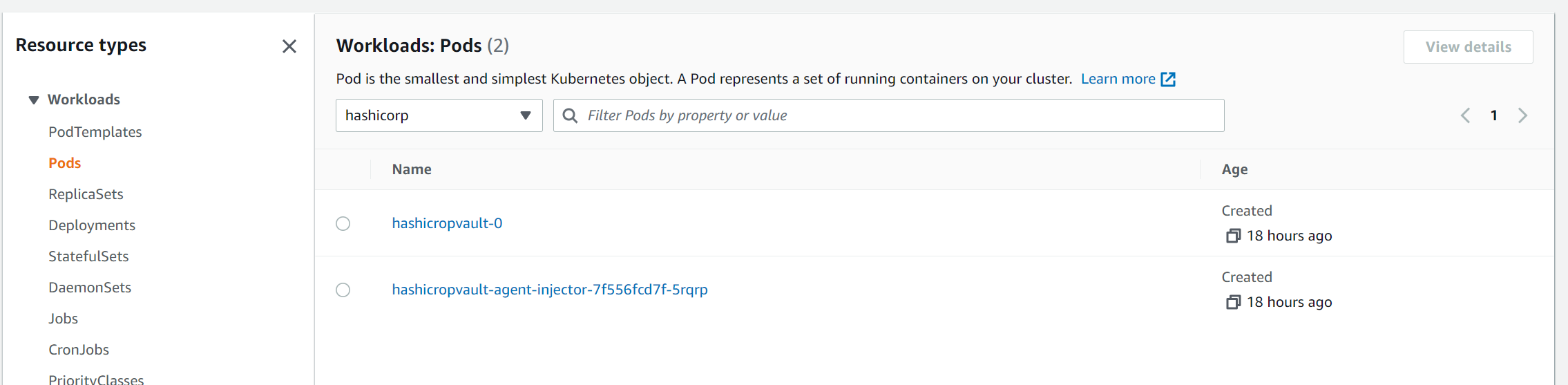




# Deploy a helm chart HashiCorp Vault

HashiCorp Vault can be deployed by just enabling vault addon in the eks blueprints kubernetes addons. The script deploys it using helm release as shown below. It also creates ane new namespace and deploy vault in that namespace. We can deploy any application with the officially provided helm chart. For the sake of simplicity script is deploying default values of the chart. To change the value, create a yaml file and porive it in the helm\_release block



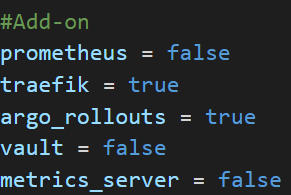
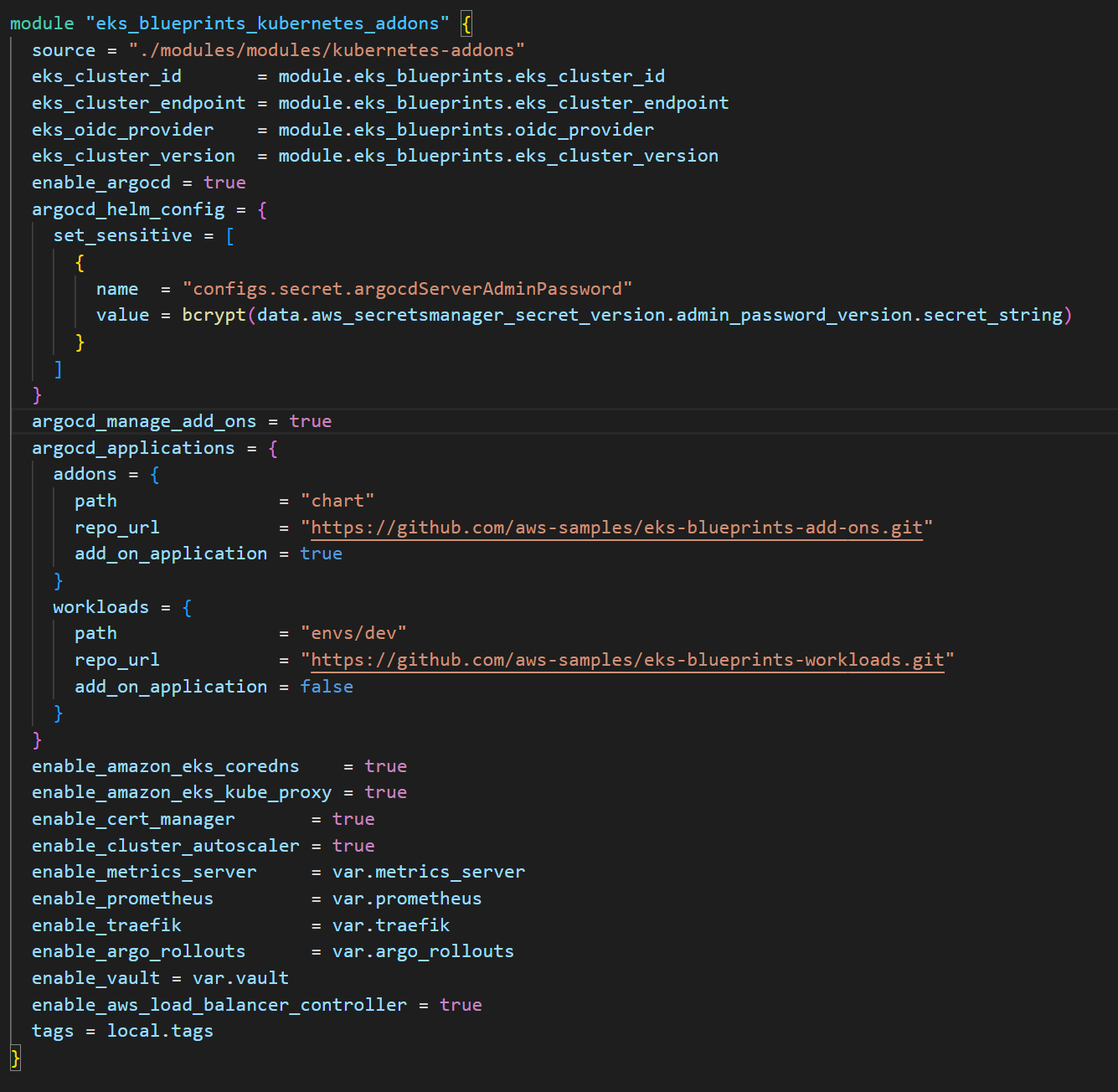


For the sake of simplicity script is deploying default values of the chart. To change the value, create a yaml file and porvide it in the helm\_release block as given below.



# Deploy an application with ArgoCD

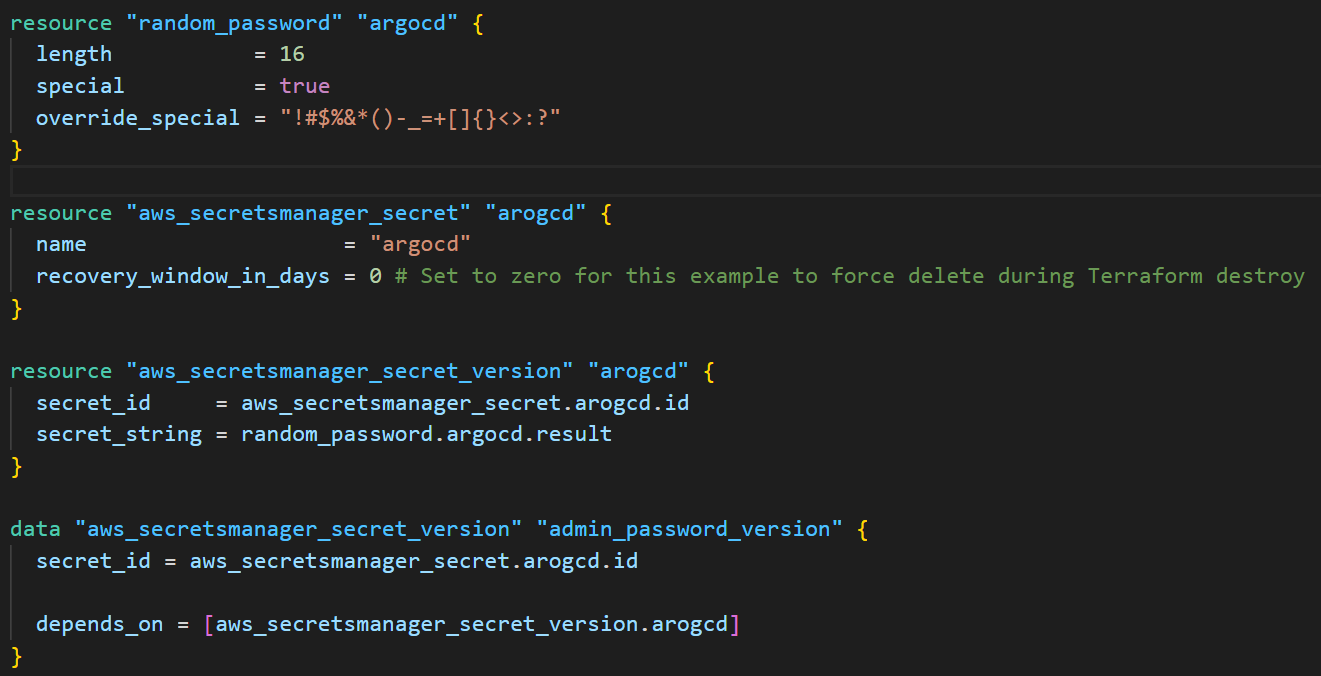
Argocd needs to be added using the eks blueprint addon. The script deployed the sample applications provided argocd in the github repo [*https://github.com/aws-samples/eks-blueprints-workloads.git*](https://github.com/aws-samples/eks-blueprints-workloads.git). Any application can be deployed the same way with github. Code snippet for deploying ArgoCD and applications are below. Script can also enable other essential addons which can be selected in terrafomr.tfvars.



For admin password for ArgoCD we need to deploy a secret in AWS secret manager. To get this password please run the following command after deployment of infrastructure.

***aws secretsmanager get-secret-value --secret-id argocd --region*** <REGION>

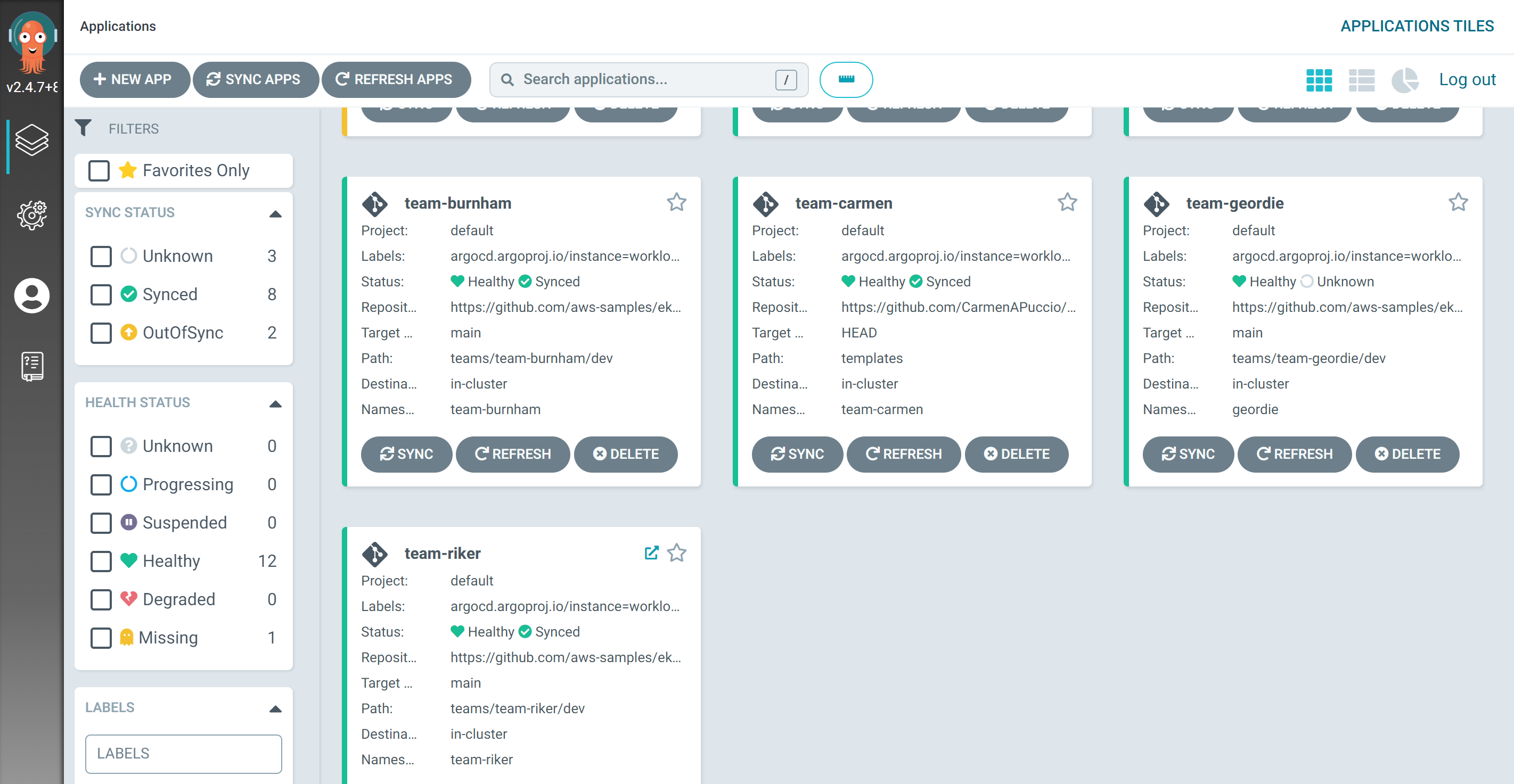
Please find the snippet below for the secret creation



Current this script doesnot have ingress for ArgoCD application to access the UI, kubetctl port-forwarding is used. Use the following command for port forwarding

***kubectl port-forward svc/argo-cd-argocd-server 8080:443 -n argocd***

Below screenshot is showing the 4 sample application deployed by ArgoCD using the repo provided in the script.

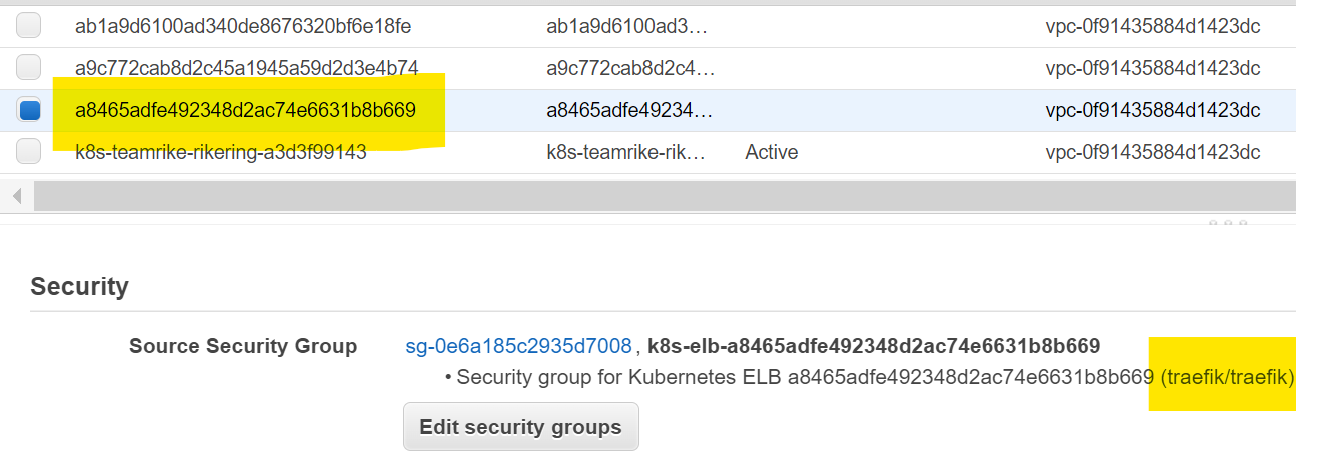


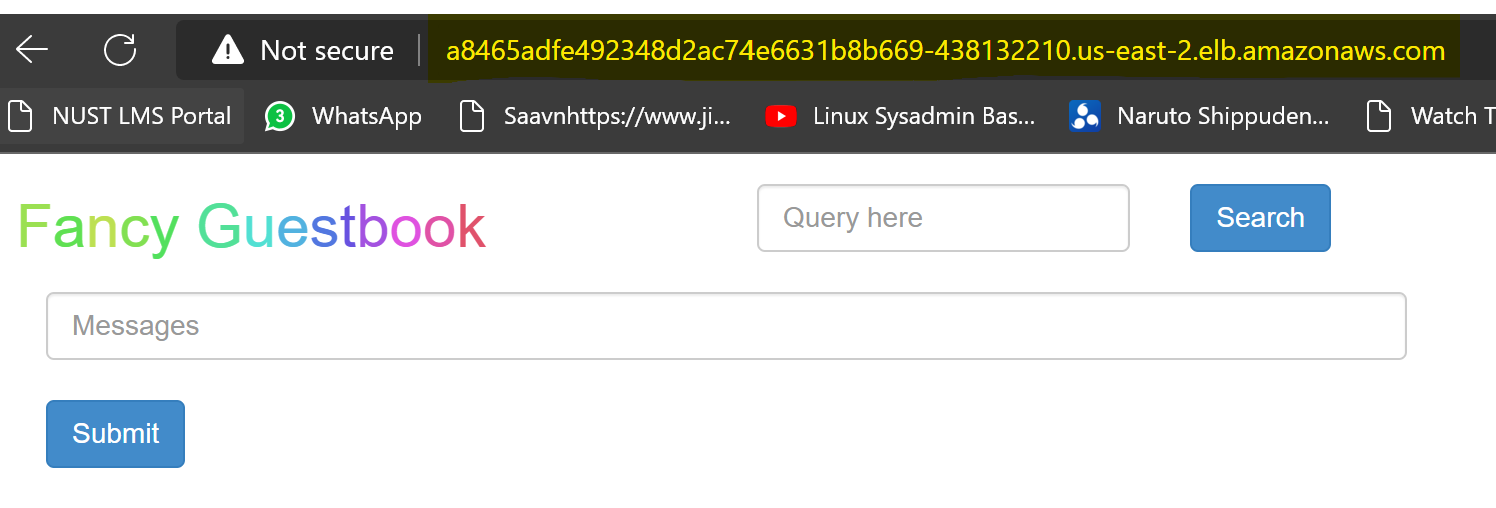
Use Traefik for Ingress

Traefik can be enabled as an addon. Team-riker application in the script we are creating an ingress for the team riker application to use traefik as ingress.



“guestbook-ui” is the name of the service created by team-riker project. In Load balancer section of AWS, we can see 4 load balancers. 2 of them are created by team-geordie application, 1 by team-riker and 1 by treafik. Opening endpoint of the traefik loadbalancer will show team-riker app which can be confirmed by opening endpoint of team-riker alb. Team-riker is using alb as default.





# Future Improvements

* Use advance configuration for managed node groups I.e custom IAM role , custom security groups etc
* Use custom configuration for application deployment using helm
* Create an ingress for ArgoCD or use ALB service to access it directly