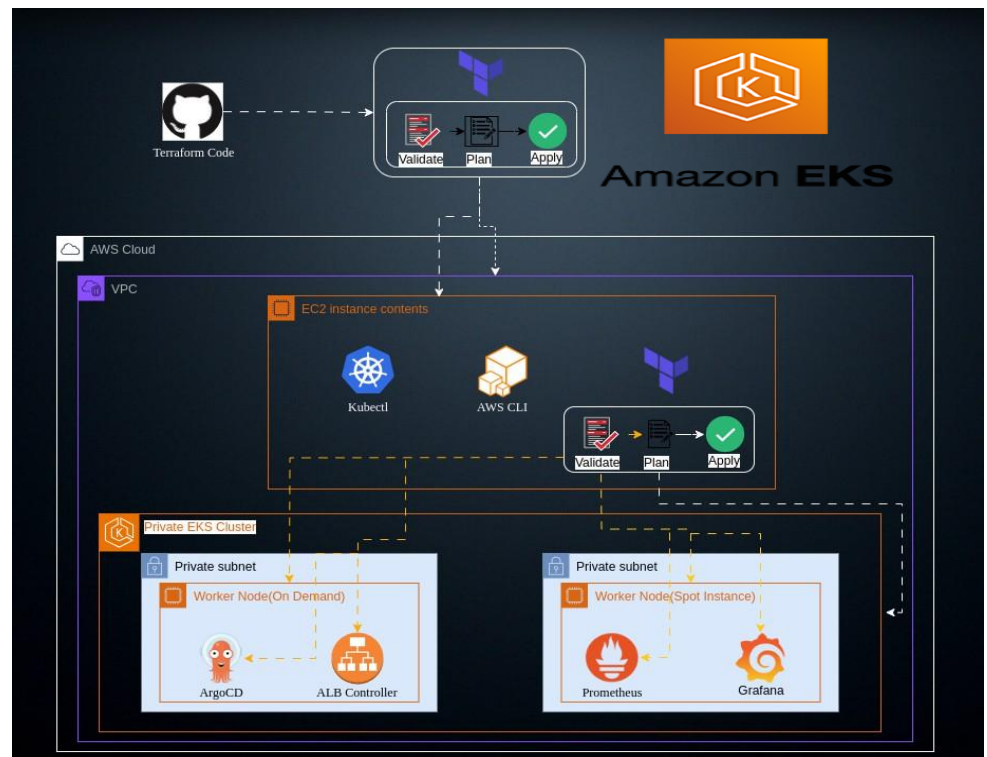




**Configure ArgoCD,
Prometheus, Grafana &**

AWS Load Balancer Controller on EKS Cluster using Terraform



Introduction

In today's DevOps-driven world, automating infrastructure deployment is crucial for maintaining efficiency and scalability. Setting up a secure and robust EKS (Elastic Kubernetes Service) cluster, complete with essential tools like ArgoCD, Prometheus, and Grafana, requires careful planning and execution. This guide will walk you through the entire process, from configuring your environment to deploying your infrastructure using Terraform, ensuring that your private EKS cluster is up and running smoothly with all the necessary resources.

Why Use Terraform for Configuring ArgoCD, Prometheus, and Other Tools?

Configuring tools like ArgoCD, Prometheus, and Grafana through Terraform offers several key advantages:

1. **Infrastructure as Code (IaC):** Terraform allows you to define your entire infrastructure as code, making it versionable, repeatable, and easier to manage. This approach minimizes the risk of manual configuration errors and ensures consistency across environments.
2. **Automated and Scalable Deployments:** Terraform automates the deployment process, enabling you to scale your infrastructure efficiently. Whether you're deploying in a development, staging, or production environment, Terraform ensures that all resources are provisioned and configured correctly with minimal manual intervention.
3. **Easier Maintenance and Updates:** With Terraform, updating or modifying configurations for tools like ArgoCD and Prometheus is straightforward. You can manage changes centrally, apply updates consistently, and track modifications through version control, reducing the complexity of managing dynamic infrastructures.
4. **Improved Security and Compliance:** By using Terraform, you can enforce security policies and compliance standards across your infrastructure. Terraform configurations can be audited, and changes can be tracked, helping to maintain a secure and compliant environment.

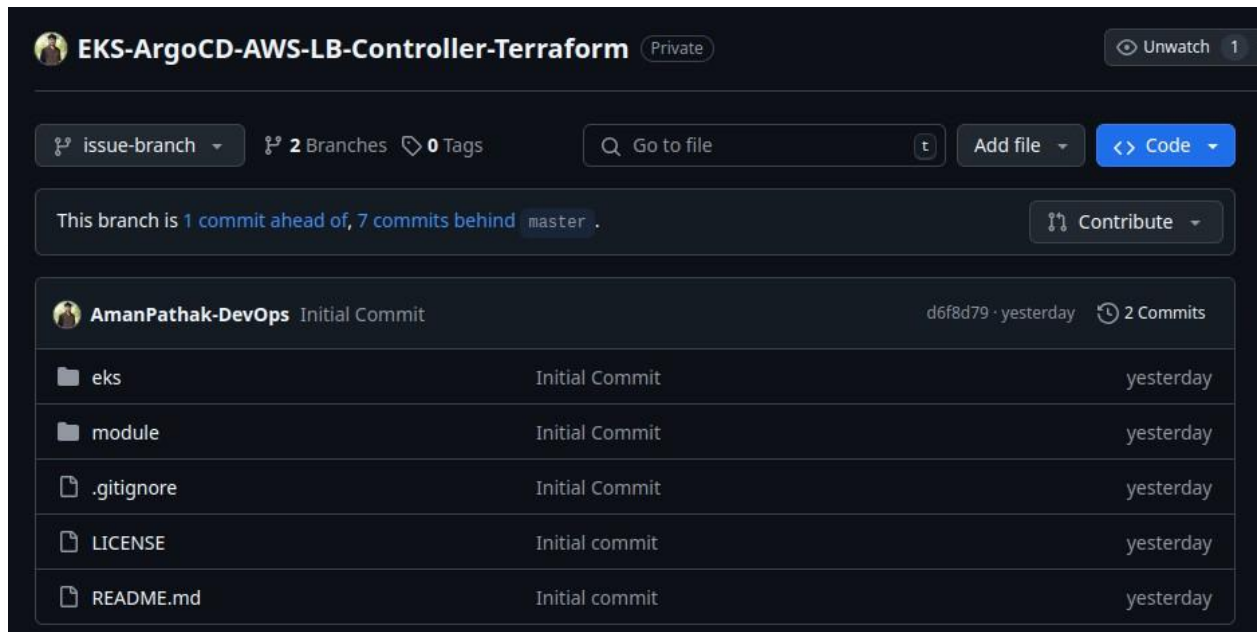
Prerequisites

Before diving into the deployment process, ensure that you have the following prerequisites in place:

1. **AWS Account:** You should have access to an AWS account with sufficient permissions to create and manage resources like VPCs, EC2 instances, and EKS clusters.
2. **Terraform Installed:** Terraform must be installed on your local machine or the server from which you will manage the infrastructure.
3. **Git:** Ensure Git is installed for cloning the necessary repositories.
4. **AWS CLI:** The AWS CLI tool should be installed and configured with appropriate credentials for deploying resources on AWS.
5. **Basic Knowledge of Terraform and Kubernetes:** Familiarity with Terraform and Kubernetes is essential to follow along with the steps and understand the infrastructure you're deploying.

Demonstration

So, I have created two branches Issue-branch and the other one is default named as master. Issue branch is to let you understand, why you can't deploy argoCD, Prometheus, and any other resources in your EKS Cluster.



This is because your Cluster is Private. You need to be in the same network to enter your cluster and make any changes or configure something.

So, the Issue branch will create everything related to Cluster only such as EKS Cluster, Node groups, EKS Add-ons, etc. But when it tries to deploy argocd and other resources inside your cluster It will throw the error.

You can check the error in the below snippet.

```
module.eks.aws_eks_addon.eks-addons["3"]: Still creating... [20s elapsed]
module.eks.aws_eks_addon.eks-addons["0"]: Still creating... [30s elapsed]
module.eks.aws_eks_addon.eks-addons["3"]: Still creating... [30s elapsed]
module.eks.aws_eks_addon.eks-addons["2"]: Still creating... [30s elapsed]
module.eks.aws_eks_addon.eks-addons["3"]: Still creating... [40s elapsed]
module.eks.aws_eks_addon.eks-addons["0"]: Still creating... [40s elapsed]
module.eks.aws_eks_addon.eks-addons["2"]: Still creating... [40s elapsed]
module.eks.aws_eks_addon.eks-addons["0"]: Still creating... [50s elapsed]
module.eks.aws_eks_addon.eks-addons["3"]: Still creating... [50s elapsed]
module.eks.aws_eks_addon.eks-addons["0"]: Creation complete after 51s [id=dev-medium-eks-cluster:vpc-cni]
module.eks.aws_eks_addon.eks-addons["3"]: Creation complete after 59s [id=dev-medium-eks-cluster:aws-ebs-csi-driver]
data.aws_eks_cluster.eks-cluster: Reading...
data.aws_eks_cluster.eks-cluster: Read complete after 0s [id=dev-medium-eks-cluster]
kubernetes_namespace.argocd: Creating...
kubernetes_service_account.example: Creating...
kubernetes_service_account.example: Still creating... [10s elapsed]
kubernetes_namespace.argocd: Still creating... [10s elapsed]
kubernetes_namespace.argocd: Still creating... [20s elapsed]
kubernetes_service_account.example: Still creating... [20s elapsed]
kubernetes_service_account.example: Still creating... [30s elapsed]
kubernetes_namespace.argocd: Still creating... [30s elapsed]

Error: Post "https://1606FBF9A7E7FC3417BDE71A310D655C.gr7.us-east-1.eks.amazonaws.com/api/v1/namespaces": dial tcp [64:ff9b:a10:8dfd]:443: i/o timeout

with kubernetes_namespace.argocd,
on argocd.tf line 1, in resource "kubernetes_namespace" "argocd":
1: resource "kubernetes_namespace" "argocd" {}

Error: Post "https://1606FBF9A7E7FC3417BDE71A310D655C.gr7.us-east-1.eks.amazonaws.com/api/v1/namespaces/default/serviceaccounts": context deadline exceeded

with kubernetes_service_account.example,
on kubernetes.tf line 70, in resource "kubernetes_service_account" "example":
70: resource "kubernetes_service_account" "example" {}

Releasing state lock. This may take a few moments...
aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ git branch
* issue-branch
master
```

P.S.: The EKS Cluster was configured as it was not visible in above snippet.


```

module.eks.aws_eks_node_group.ondemand-node: Still destroying... [id=dev-medium-eks-cluster:dev-medium-eks-cluster-on-demand-nodes, 2n30s elapsed]
module.eks.aws_eks_node_group.spot-node: Destruction complete after 2n31s
module.eks.aws_eks_node_group.ondemand-node: Still destroying... [id=dev-medium-eks-cluster:dev-medium-eks-cluster-on-demand-nodes, 2n40s elapsed]
module.eks.aws_eks_node_group.ondemand-node: Destruction complete after 2n42s
module.eks.aws_iam_role.eks-nodegroup-role[0]: Destroying... [id=dev-medium-eks-cluster-nodegroup-role-2377]
module.eks.aws_eks_cluster.eks[0]: Destroying... [id=dev-medium-eks-cluster]
module.eks.aws_iam_role.eks-nodegroup-role[0]: Destruction complete after 3s
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 10s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 20s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 30s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 40s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 50s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m0s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m10s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m20s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m30s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m40s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m50s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m0s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m10s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m20s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m30s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m40s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m50s elapsed]
module.eks.aws_eks_cluster.eks[0]: Destruction complete after 2m51s
module.eks.aws_iam_role.eks-cluster-role[0]: Destroying... [id=dev-medium-eks-cluster-role-2377]
module.eks.aws_subnet.private-subnet[0]: Destroying... [id=subnet-00204cf95bcd3cd1]
module.eks.aws_subnet.private-subnet[2]: Destroying... [id=subnet-0fde9b8584aa5a3a]
module.eks.aws_subnet.private-subnet[1]: Destroying... [id=subnet-07b6d658ba103d26f]
module.eks.aws_security_group.eks-cluster-sg: Destroying... [id=sg-0c8e072f99d14c4a3]
module.eks.aws_iam_role.eks-cluster-role[0]: Destruction complete after 2s
module.eks.random_integer.random_suffix: Destroying... [id=2377]
module.eks.aws_subnet.private-subnet[0]: Destruction complete after 3s
module.eks.aws_subnet.private-subnet[2]: Destruction complete after 3s
module.eks.aws_subnet.private-subnet[1]: Destruction complete after 4s
module.eks.aws_security_group.eks-cluster-sg: Destruction complete after 4s
module.eks.aws_vpc.vpc: Destroying... [id=vpc-0230937f80bc75d5e]
module.eks.aws_vpc.vpc: Destruction complete after 2s
Releasing state lock. This may take a few moments...

Destroy complete! Resources: 40 destroyed.


```

How to solve this?

To deploy anything inside your EKS Cluster, we need an instance or server to deploy those configurations that have the same VPC.


So, we are going to deploy our VPC and one EC2 server first. Then from the ec2 server, we will deploy everything related to the EKS Cluster.

In the master branch of the repository, you will find two modules. The first module is vpc-ec2 which we need to apply first.

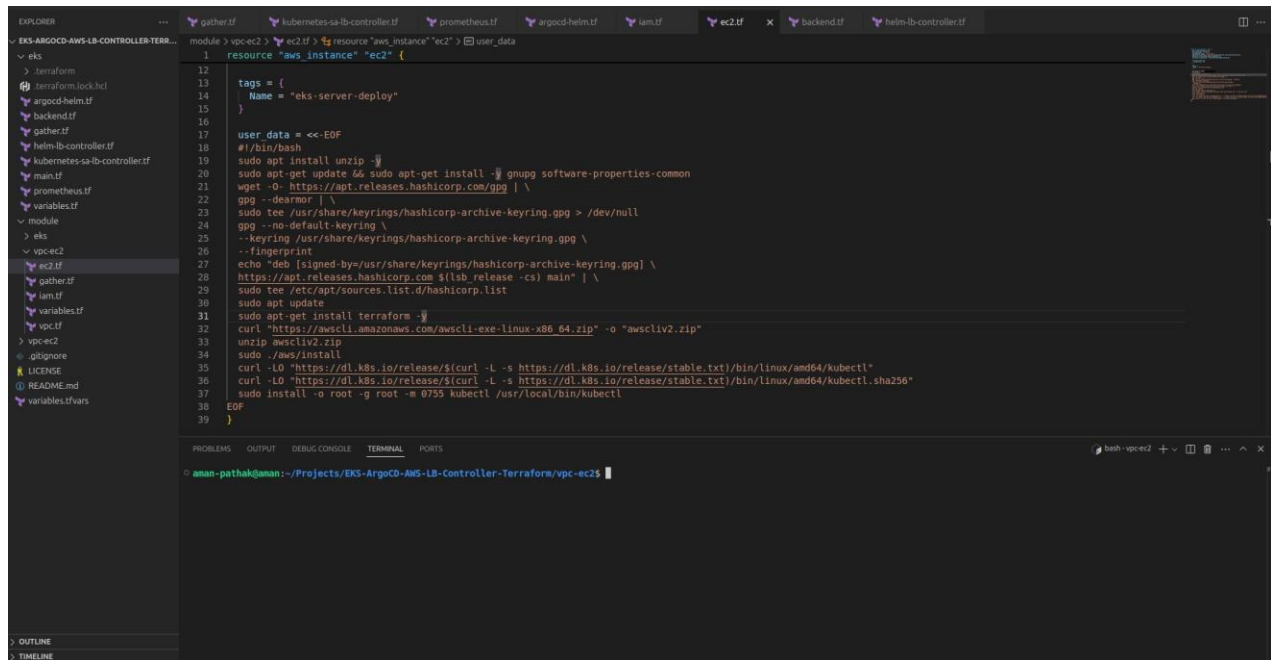

EKS-ArgoCD-AWS-LB-Controller-Terraform
Private
Unwatch 1

master
2 Branches
0 Tags

Add file
Code

	AmanPathak-DevOps File Structure updated	066af7d · now	8 Commits
eks	Updated naming	1 hour ago	
module	Updated policies for ec2	16 minutes ago	
vpc-ec2	File Structure updated	now	
.gitignore	Initial Commit	19 hours ago	
LICENSE	Initial commit	yesterday	
README.md	Initial commit	yesterday	
variables.tfvars	Added minor changes	7 hours ago	

So, clone the repository with the master branch and navigate to the vpc-ec2 directory.

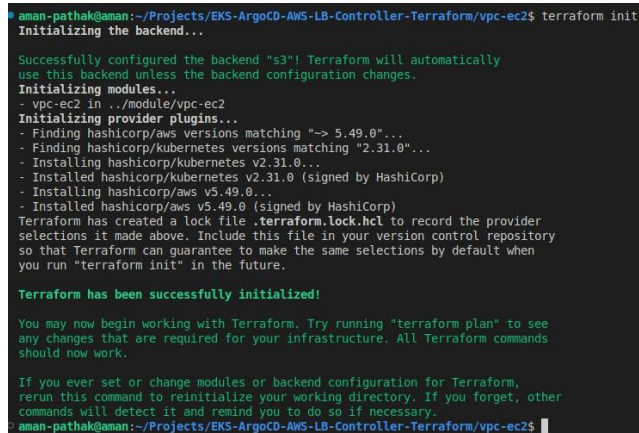


```
EXPLORER
  EKS-ARGOCD-AWS-LB-CONTROLLER-TERR...
    .terraform
    terraform.lock.hcl
    argocd-helm.tf
    backend.tf
    gather.tf
    helm-lb-controller.tf
    kubernetes-sa-lb-controller.tf
    main.tf
    prometheus.tf
    variables.tf
  module
    eks
    vpc-ec2
    vpc-ec2
      ec2.tf
        gather.tf
        iam.tf
        variables.tf
    vpc-ec2
      ignore
      LICENSE
      README.md
      variables.tfvars

module > vpc-ec2 > ec2.tf > resource "aws_instance" "ec2" > user_data
1 resource "aws_instance" "ec2" {
2
3   tags = {
4     Name = "eks-server-deploy"
5   }
6
7   user_data = <<- EOF
8   #!/bin/bash
9   sudo apt install unzip -y
10  sudo apt-get update && sudo apt-get install -y gnupg software-properties-common
11  wget -O- https://apt.releases.hashicorp.com/gpg | \
12  gpg --dearmor | \
13  sudo tee /usr/share/keyrings/hashicorp-archive-keyring.gpg > /dev/null
14  gpg --no-default-keyring \
15  --keyring /usr/share/keyrings/hashicorp-archive-keyring.gpg \
16  --fingerprint
17  echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] \
18  https://apt.releases.hashicorp.com $(lsb_release -cs) main" | \
19  sudo tee /etc/apt/sources.list.d/hashicorp.list
20  sudo apt update
21  sudo apt-get install terraform -y
22  curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
23  unzip awscliv2.zip
24  sudo ./aws/install
25  curl -LO "https://dl.k8s.io/release/${curl -L -s https://dl.k8s.io/release/stable.txt}/bin/linux/amd64/kubectl"
26  curl -LO "https://dl.k8s.io/release/${curl -L -s https://dl.k8s.io/release/stable.txt}/bin/linux/amd64/kubectl.sha256"
27  sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
28  EOF
29 }
```

Run the below commands to deploy VPC(Other networking resources) & EC2 Server on AWS.

> terraform init



```
aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2$ terraform init
Initializing the backend...

Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.
Initializing modules...
- vpc-ec2 in ../module/vpc-ec2
Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.49.0"...
- Finding hashicorp/kubernetes versions matching "~2.31.0"...
- Installing hashicorp/kubernetes v2.31.0...
- Installed hashicorp/kubernetes v2.31.0 (signed by HashiCorp)
- Installing hashicorp/aws v5.49.0...
- Installed hashicorp/aws v5.49.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2$
```

> terraform validate

```
aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2$ terraform validate
Success! The configuration is valid.

aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2$
```

```
> terraform plan -var-file=../variables.tfvars
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
+ enable_dns_support           = true
+ enable_network_address_usage_metrics = (known after apply)
+ id                           = (known after apply)
+ instance_tenancy             = "default"
+ ipv6_association_id          = (known after apply)
+ ipv6_cidr_block              = (known after apply)
+ ipv6_cidr_block_network_border_group = (known after apply)
+ main_route_table_id          = (known after apply)
+ owner_id                     = (known after apply)
+ tags                         = {
+   "Env" = "dev"
+   "Name" = "dev-medium-vpc"
+ }
+ tags_all                     = {
+   "Env" = "dev"
+   "Name" = "dev-medium-vpc"
+ }
}

Plan: 24 to add, 0 to change, 0 to destroy.

Warning: Value for undeclared variable
The root module does not declare a variable named "ondemand_instance_types" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.
To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Value for undeclared variable
The root module does not declare a variable named "max_capacity_on_demand" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.
To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Values for undeclared variables
In addition to the other similar warnings shown, 11 other variable(s) defined without being declared.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
Releasing state lock. This may take a few moments...
aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2$
```

```
> terraform apply -auto-approve -var-file=../variables.tfvars
```



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
module.vpc-ec2.aws_instance.ec2: Still creating... [10s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [20s elapsed]
module.vpc-ec2.aws_instance.ec2: Still creating... [20s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [30s elapsed]
module.vpc-ec2.aws_instance.ec2: Still creating... [30s elapsed]
module.vpc-ec2.aws_instance.ec2: Creation complete after 37s [id=i-0d8a3bc509812b65b]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [40s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [50s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [1m0s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [1m0s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [1m20s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [1m30s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Still creating... [1m40s elapsed]
module.vpc-ec2.aws_nat_gateway.ngw: Creation complete after 1m50s [id=nat-0e920b64e401eb765]
module.vpc-ec2.aws_route_table.private-rt: Creating...
module.vpc-ec2.aws_route_table.private-rt: Creation complete after 3s [id=rtb-0b3db5efa7d41bab3]
module.vpc-ec2.aws_route_table_association.private-rt-association[1]: Creating...
module.vpc-ec2.aws_route_table_association.private-rt-association[0]: Creating...
module.vpc-ec2.aws_route_table_association.private-rt-association[2]: Creating...
module.vpc-ec2.aws_route_table_association.private-rt-association[0]: Creation complete after 2s [id=rtbassoc-0a46e70188fb044c]
module.vpc-ec2.aws_route_table_association.private-rt-association[2]: Creation complete after 2s [id=rtbassoc-0de5af7a0dd06803]
module.vpc-ec2.aws_route_table_association.private-rt-association[1]: Creation complete after 2s [id=rtbassoc-0b599d70a94f9b8fc]

Warning: Value for undeclared variable
The root module does not declare a variable named "max_capacity_on_demand" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Value for undeclared variable
The root module does not declare a variable named "min_capacity_spot" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Values for undeclared variables
In addition to the other similar warnings shown, 11 other variable(s) defined without being declared.

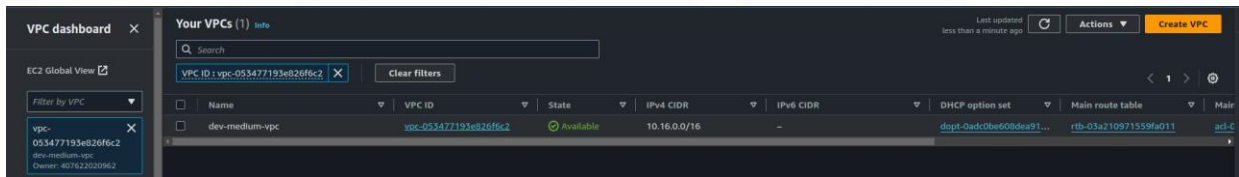
Releasing state lock. This may take a few moments...

Apply complete! Resources: 24 added, 0 changed, 0 destroyed.
aman-pathak@aman:~/Projects/EKS-ArgoCD-AWS-LB-Controller-Terraform/vpc-ec2$
```

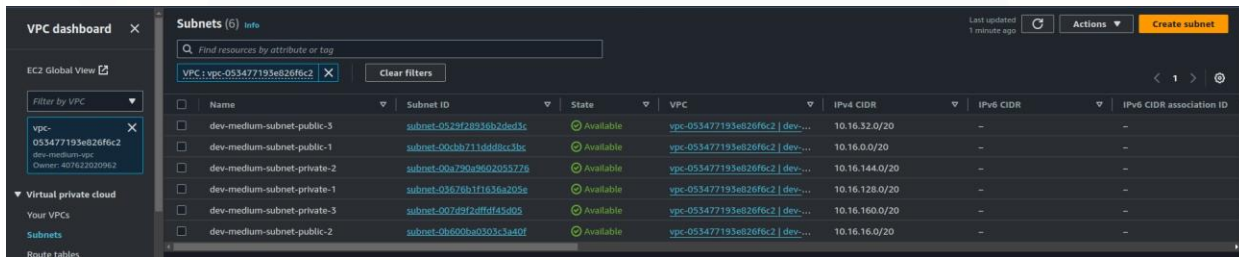
VPC & other services created through the above commands

Here are the snippets

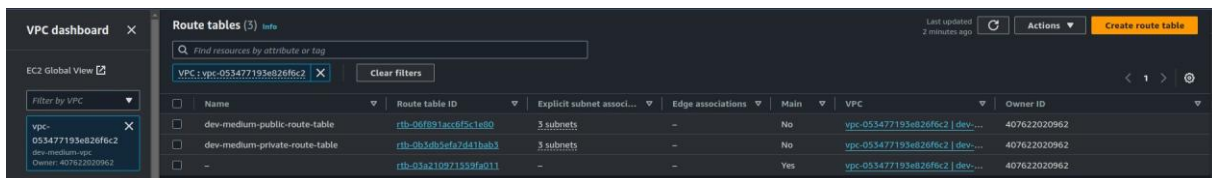
VPC



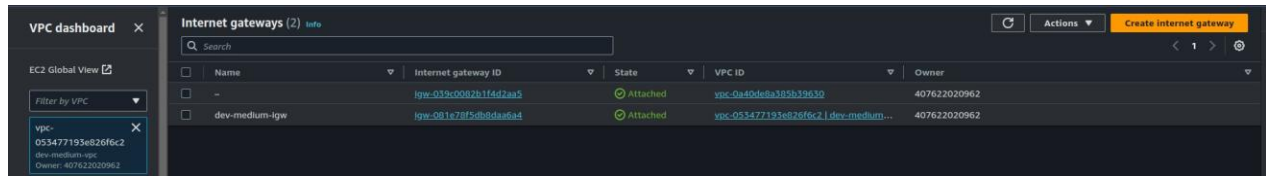
Subnets



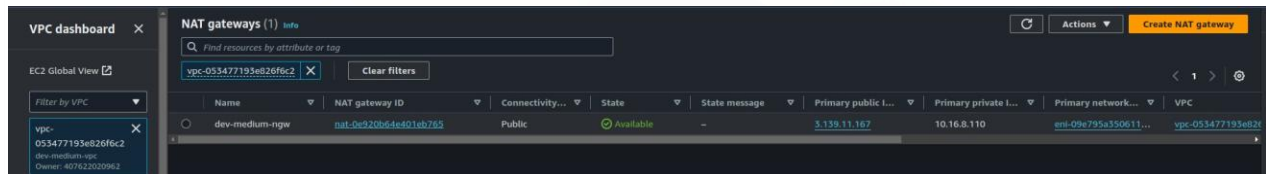
Route tables



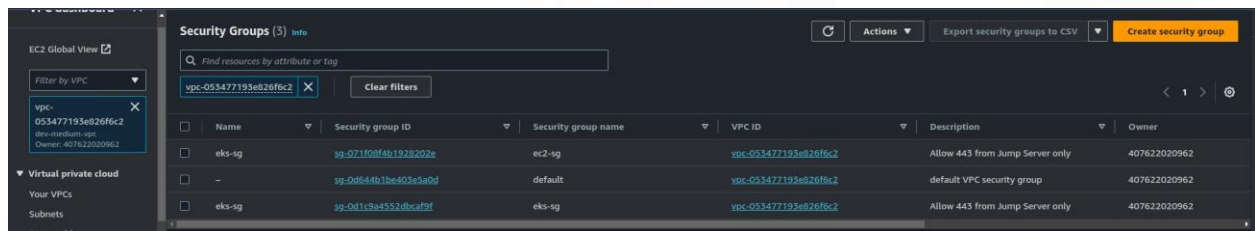
Internet Gateway



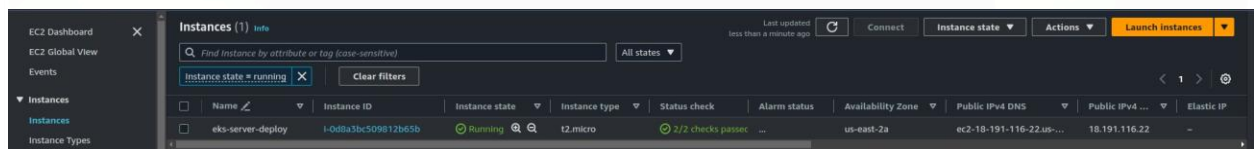
NAT Gateway



Security Group



EC2-Instance



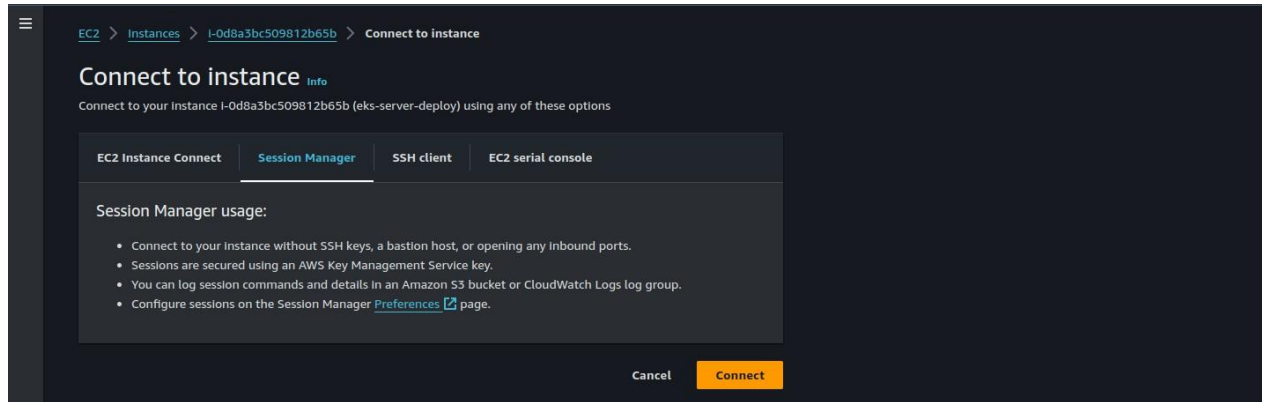
So, everything is created according to our requirements.

Now log in to the Server by selecting the created ec2 instance and click on

Connect

You can log in without the Pem file as we did not follow that.

If Session Manager is struggling to connect to the instance, you can use EC2 Instance Connect to login to the server.



I am logged In to my server and switched to the Ubuntu user with the help of the below command

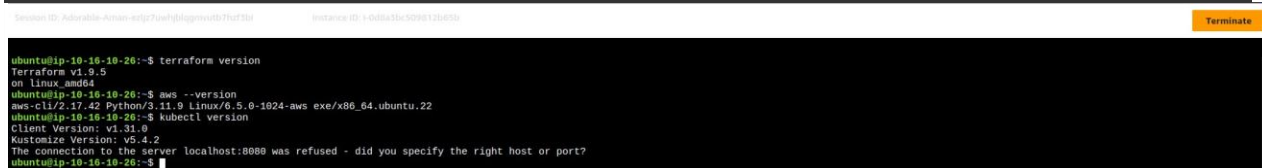
```
> sudo su ubuntu
```



We need to install a few tools as our pre-requisites. We installed it through user data in the Terraform script

Run the below command to validate whether it's installed or not. If not, wait for 5-10 minutes,

```
> terraform version
aws --version
kubectl version
```



Once all the tools are installed then, we need to configure AWS CLI as we need to deploy our infrastructure over AWS Cloud.

Run the below command to configure CLI and make sure you have sufficient permission with your credentials. For demonstration, you can utilize Administration Access user keys

P.S.: Don't use the same access keys as you will have your keys in your AWS Account

```
> aws configure
```

```
Session ID: Adorable-Aman-edp7ueh9lqgmrvub7Huf59a Instance ID: i-0d8a3bc506612b65b
ubuntu@ip-10-16-10-26:~$ aws configure
AWS Access Key ID [None]: AKIAVS2BKNSRFP2VCSR4H
AWS Secret Access Key [None]: ez58Pn3nWkHtAMPwB01/epCunsoomwqgQZf1fghk
Default region name [None]: us-east-2
Default output format [None]: json
ubuntu@ip-10-16-10-26:~$
```

Now, clone the same repository where our Terraform code is present for EKS

```
> git clone
https://github.com/AmanPathak-DevOps/EKS-ArgoCD-AWS-LB-Controller-Terraform.git
```

```
Session ID: Adorable-Aman-edp7ueh9lqgmrvub7Huf59a Instance ID: i-0d8a3bc506612b65b
ubuntu@ip-10-16-10-26:~$ git clone https://github.com/AmanPathak-DevOps/EKS-ArgoCD-AWS-LB-Controller-Terraform.git
Cloning into 'EKS-ArgoCD-AWS-LB-Controller-Terraform'...
Username for 'https://github.com': AmanPathak-DevOps
Password for 'https://AmanPathak-DevOps@github.com':
remote: Enumerating objects: 85, done.
remote: Counting objects: 100% (85/85), done.
remote: Compressing objects: 100% (68/68), done.
remote: Total 85 (delta 40), reused 64 (delta 23), pack-reused 0 (from 0)
Receiving objects: 100% (85/85), 19.01 KiB | 1.73 MiB/s, done.
Resolving deltas: 100% (48/40), done.
ubuntu@ip-10-16-10-26:~$
ubuntu@ip-10-16-10-26:~$ ls
EKS-ArgoCD-AWS-LB-Controller-Terraform
ubuntu@ip-10-16-10-26:~$
```

Now, we are ready to deploy our EKS Cluster, and other tools through Terraform. Navigate to the eks directory and run the below commands to deploy it

```
> terraform init
```

```
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ terraform init
Initializing the backend...

Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.
Initializing modules...
- eks in ../module/eks
Initializing provider plugins...
- Finding latest version of hashicorp/tls...
- Finding hashicorp/helm versions matching ">= 2.10.0"...
- Finding hashicorp/aws versions matching ">= 5.49.0"...
- Finding hashicorp/kubernetes versions matching "2.31.0"...
- Finding latest version of hashicorp/random...
- Installing hashicorp/random v3.6.2...
- Installed hashicorp/random v3.6.2 (signed by HashiCorp)
- Installing hashicorp/tls v4.0.5...
- Installed hashicorp/tls v4.0.5 (signed by HashiCorp)
- Installing hashicorp/helm v2.10.1...
- Installed hashicorp/helm v2.10.1 (signed by HashiCorp)
- Installing hashicorp/aws v5.49.0...
- Installed hashicorp/aws v5.49.0 (signed by HashiCorp)
- Installing hashicorp/kubernetes v2.31.0...
- Installed hashicorp/kubernetes v2.31.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

```
> terraform validate
```

```
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ terraform validate
Success! The configuration is valid.

ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

```
> terraform apply -auto-approve -var-file=../variables.tfvars
```

```
> terraform plan -var-file=../variables.tfvars
```

```
Session ID: Adorable-Arrow-edp7JowHqIggmVub7Uf3M Instance ID: i-0d8a3bc309b12b65b Terminate

}

# module.eks.aws_iam_role_policy_attachment.eks-AmazonWorkerNodePolicy[0] will be created
+ resource "aws_iam_role_policy_attachment" "eks-AmazonWorkerNodePolicy" {
  + id           = (known after apply)
  + policy_arn   = "arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy"
  + role        = (known after apply)
}

# module.eks.aws_iam_role_policy_attachment.eks-oidc-policy-attach will be created
+ resource "aws_iam_role_policy_attachment" "eks-oidc-policy-attach" {
  + id           = (known after apply)
  + policy_arn   = (known after apply)
  + role        = "eks-oidc"
}

# module.eks.random_integer.random_suffix will be created
+ resource "random_integer" "random_suffix" {
  + id           = (known after apply)
  + max         = 9999
  + min         = 1000
  + result      = (known after apply)
}

Plan: 25 to add, 0 to change, 0 to destroy.

Warning: Value for undeclared variable

The root module does not declare a variable named "ec2-sg" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Value for undeclared variable

The root module does not declare a variable named "ec2-name" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Values for undeclared variables

In addition to the other similar warnings shown, 3 other variable(s) defined without being declared.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
ubuntu@ip-10-10-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

```
Session ID: Adorable-Arrow-edp7JowHqIggmVub7Uf3M Instance ID: i-0d8a3bc309b12b65b Terminate

module.eks.aws_eks_addon.eks-addons["2"]: Still creating... [50s elapsed]
module.eks.aws_eks_addon.eks-addons["0"]: Still creating... [50s elapsed]
module.eks.aws_eks_addon.eks-addons["3"]: Still creating... [50s elapsed]
module.eks.aws_eks_addon.eks-addons["3"]: Creation complete after 55s [id=dev-medium-eks-cluster:aws-eks-csi-driver]
module.eks.aws_eks_addon.eks-addons["2"]: Creation complete after 55s [id=dev-medium-eks-cluster:kube-proxy]
module.eks.aws_eks_addon.eks-addons["0"]: Creation complete after 55s [id=dev-medium-eks-cluster:vpc-cni]
data.aws_eks_cluster.eks-cluster: Reading...
data.aws_eks_cluster.eks-cluster: Read complete after 0s [id=dev-medium-eks-cluster]
kubernetes.service_account.lb-controller: Creating...
kubernetes.service_account.lb-controller: Creation complete after 0s [id=default/aws-load-balancer-controller]
helm.release.aws-load-balancer-controller: Creating...
helm.release.aws-load-balancer-controller: Still creating... [10s elapsed]
helm.release.aws-load-balancer-controller: Creation complete after 10s [id=aws-load-balancer-controller]
helm.release.argoctl: Creating...
helm.release.argoctl: Still creating... [10s elapsed]
helm.release.argoctl: Still creating... [20s elapsed]
helm.release.argoctl: Still creating... [30s elapsed]
helm.release.argoctl: Creation complete after 30s [id=argoctl]
helm.release.prometheus-helm: Creating...
helm.release.prometheus-helm: Still creating... [10s elapsed]
helm.release.prometheus-helm: Still creating... [20s elapsed]
helm.release.prometheus-helm: Still creating... [30s elapsed]
helm.release.prometheus-helm: Still creating... [40s elapsed]
helm.release.prometheus-helm: Still creating... [50s elapsed]
helm.release.prometheus-helm: Still creating... [1m0s elapsed]
helm.release.prometheus-helm: Still creating... [1m10s elapsed]
helm.release.prometheus-helm: Creation complete after 1m11s [id=prometheus]

Warning: Value for undeclared variable

The root module does not declare a variable named "ec2-name" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Value for undeclared variable

The root module does not declare a variable named "ec2-sg" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the -compact-warnings option.

Warning: Values for undeclared variables

In addition to the other similar warnings shown, 3 other variable(s) defined without being declared.

Apply complete! Resources: 25 added, 0 changed, 0 destroyed.
ubuntu@ip-10-10-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

EKS Cluster & other services created through the above commands

Here are the snippets

EKS Cluster

EKS > Clusters

Clusters (1) Info

Filter clusters

Cluster name	Status	Kubernetes version	Support period	Upgrade policy	Created	Provider
dev-medium-eks-cluster	Active	1.29 Upgrade now	Standard support until March 23, 2025	Extended	16 minutes ago	EKS

Node Groups

EKS > Clusters > dev-medium-eks-cluster

dev-medium-eks-cluster

End of standard support for Kubernetes version 1.29 is March 23, 2025. On that date, your cluster will enter the extended support period with additional fees. For more information, see the [pricing page](#).

Cluster info

Status: Active, Kubernetes version: 1.29, Support period: Standard support until March 23, 2025, Provider: EKS

Overview | Resources | **Compute** | Networking | Add-ons | Access | Observability | Upgrade insights | Update history | Tags

Nodes (2) Info

Filter Nodes by property or value

Node name	Instance type	Node group	Created	Status
ip-10-16-147-226.us-east-2.compute.internal	t3a.medium	dev-medium-eks-cluster-on-demand-nodes	Created 4 minutes ago	Ready
ip-10-16-157-126.us-east-2.compute.internal	t3a.large	dev-medium-eks-cluster-spot-nodes	Created 4 minutes ago	Ready

Node groups (2) Info

Edit | Delete | Add node group

To validate whether the other Kubernetes resources have been created or not. We need to update the kubconfig on our same ec2 server
Run the below command to update the kubeconfig

```
> aws eks update-kubeconfig --name dev-medium-eks-cluster --region us-east-2
```

```
Session ID: Aduvabde-Awren-ec2p7owh9l4ggmrvu71of736l Instance ID: i-00ba3bc3096123653b
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ aws eks update-kubeconfig --name dev-medium-eks-cluster --region us-east-2
Added new context arn:aws:eks:us-east-2:487622828962:cluster/dev-medium-eks-cluster to /home/ubuntu/.kube/config
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

Run the below command to validate whether it's our cluster or not

```
> kubectl get nodes
```

```
Session ID: Aduvabde-Awren-ec2p7owh9l4ggmrvu71of736l Instance ID: i-00ba3bc3096123653b
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ kubectl get nodes
NAME                                STATUS    ROLES    AGE    VERSION
ip-10-16-147-226.us-east-2.compute.internal Ready    <none>   5m36s   v1.29.6-eks-1552ad9
ip-10-16-157-126.us-east-2.compute.internal Ready    <none>   5m33s   v1.29.6-eks-1552ad9
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

Check whether the resources are running in argocd namespace or not

```
> kubectl get all -n argocd
```

```
Session ID: Atorabde-Jenen-efp7Jw6Hglpgmuvb7H7c730i Instance ID: i-fdb3a3bc300812b655b Terminate
```

```
ubuntu@ip-10-10-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ kubectl get all -n argocd
```

NAME	READY	STATUS	RESTARTS	AGE
pod/argocd-application-controller-0	1/1	Running	0	4m32s
pod/argocd-applicationset-controller-765cfb7674-dt9ct	1/1	Running	0	4m33s
pod/argocd-dex-server-6d8bcbf4db-ec8bt	1/1	Running	0	4m33s
pod/argocd-notifications-controller-7df6fd8fd4-wzsjs	1/1	Running	0	4m33s
pod/argocd-redis-57bd776b7d-sq2wx	1/1	Running	0	4m33s
pod/argocd-repo-server-8476665cb-bcbbc	1/1	Running	0	4m33s
pod/argocd-server-84db774f8b-bh5mv	1/1	Running	0	4m33s

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/argocd-applicationset-controller	ClusterIP	172.20.173.9	<none>	7080/TCP	4m33s
service/argocd-dex-server	ClusterIP	172.20.32.78	<none>	5556/TCP, 5557/TCP	4m33s
service/argocd-redis	ClusterIP	172.20.140.141	<none>	6379/TCP	4m33s
service/argocd-repo-server	ClusterIP	172.20.109.123	<none>	8081/TCP	4m33s
service/argocd-server	LoadBalancer	172.20.232.82	a6368afb5259146559274c90b23d69be-105956850.us-east-2.elb.amazonaws.com	80:31297/TCP, 443:31884/TCP	4m33s

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/argocd-applicationset-controller	1/1	1	1	4m33s
deployment.apps/argocd-dex-server	1/1	1	1	4m33s
deployment.apps/argocd-notifications-controller	1/1	1	1	4m33s
deployment.apps/argocd-redis	1/1	1	1	4m33s
deployment.apps/argocd-repo-server	1/1	1	1	4m33s
deployment.apps/argocd-server	1/1	1	1	4m33s

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/argocd-applicationset-controller-765cfb7674	1	1	1	4m33s
replicaset.apps/argocd-dex-server-6d8bcbf4db	1	1	1	4m33s
replicaset.apps/argocd-notifications-controller-7df6fd8fd4	1	1	1	4m33s
replicaset.apps/argocd-redis-57bd776b7d	1	1	1	4m33s
replicaset.apps/argocd-repo-server-8476665cb	1	1	1	4m33s
replicaset.apps/argocd-server-84db774f8b	1	1	1	4m33s

```
NAME READY AGE
statefulset.apps/argocd-application-controller 1/1 4m33s
ubuntu@ip-10-10-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

Check whether the resources are running in Prometheus namespace or not

```
> kubectl get all -n prometheus
```

```
Session ID: Atorabde-Jenen-efp7Jw6Hglpgmuvb7H7c730i Instance ID: i-fdb3a3bc300812b655b Terminate
```

```
ubuntu@ip-10-10-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ kubectl get all -n prometheus
```

NAME	READY	STATUS	RESTARTS	AGE
pod/alertmanager-prometheus-kube-prometheus-alertmanager-0	2/2	Running	0	4m49s
pod/prometheus-grafana-6cc4484fc-8p4fd	3/3	Running	0	4m52s
pod/prometheus-kube-prometheus-operator-76647f58db-t58k2	1/1	Running	0	4m52s
pod/prometheus-kube-state-metrics-5b787f976b-59jz4	1/1	Running	0	4m52s
pod/prometheus-prometheus-kube-prometheus-prometheus-0	2/2	Running	0	4m48s
pod/prometheus-prometheus-node-exporter-5nqj7	1/1	Running	0	4m52s
pod/prometheus-prometheus-node-exporter-64rgd	1/1	Running	0	4m52s

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/alertmanager-operated	ClusterIP	None	<none>	9093/TCP, 9094/TCP, 9094/UDP	4m50s
service/prometheus-grafana	LoadBalancer	172.20.108.21	a345375fbc34543d490e55622e7afc64-89423726.us-east-2.elb.amazonaws.com	80:30249/TCP	4m53s
service/prometheus-kube-prometheus-alertmanager	ClusterIP	172.20.109.191	<none>	9093/TCP, 8080/TCP	4m53s
service/prometheus-kube-prometheus-operator	ClusterIP	172.20.18.255	<none>	443/TCP	4m53s
service/prometheus-kube-prometheus-prometheus	LoadBalancer	172.20.221.37	abd7eeb944f274e5d8289dd7489e9175-1643275437.us-east-2.elb.amazonaws.com	9090:31872/TCP, 8080:32243/TCP	4m53s
service/prometheus-kube-state-metrics	ClusterIP	172.20.131.125	<none>	8080/TCP	4m53s
service/prometheus-operated	ClusterIP	None	<none>	9090/TCP	4m49s
service/prometheus-prometheus-node-exporter	ClusterIP	172.20.199.114	<none>	9100/TCP	4m53s

NAME	DESIRED	CURRENT	READY	UP-TO-DATE	AVAILABLE	NODE SELECTOR	AGE
daemonset.apps/prometheus-prometheus-node-exporter	2	2	2	2	2	kubernetes.io/os=linux	4m53s

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/prometheus-grafana	1/1	1	1	4m53s
deployment.apps/prometheus-kube-prometheus-operator	1/1	1	1	4m53s
deployment.apps/prometheus-kube-state-metrics	1/1	1	1	4m53s

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/prometheus-grafana-6cc4484fc	1	1	1	4m53s
replicaset.apps/prometheus-kube-prometheus-operator-76647f58db	1	1	1	4m53s
replicaset.apps/prometheus-kube-state-metrics-5b787f976b	1	1	1	4m53s

NAME	READY	AGE
statefulset.apps/alertmanager-prometheus-kube-prometheus-alertmanager	1/1	4m50s
statefulset.apps/prometheus-prometheus-kube-prometheus-prometheus	1/1	4m49s

```
ubuntu@ip-10-10-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

Check whether the AWS Load Balancer controller pods are running in a aws-loadbalancer-controller namespace or not

```
> kubectl get all -n aws-loadbalancer-controller
```

```
Session ID: Atorabde-Jenen-efp7Jw6Hglpgmuvb7H7c730i Instance ID: i-fdb3a3bc300812b655b Terminate
```

```
ubuntu@ip-10-10-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ kubectl get all -n aws-loadbalancer-controller
```

NAME	READY	STATUS	RESTARTS	AGE
pod/aws-load-balancer-controller-7cb85f99d7-5cv4p	1/1	Running	0	9m26s
pod/aws-load-balancer-controller-7cb85f99d7-9gb0f	1/1	Running	0	9m26s

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/aws-load-balancer-webhook-service	ClusterIP	172.20.224.219	<none>	443/TCP	9m26s

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/aws-load-balancer-controller	2/2	2	2	9m26s

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/aws-load-balancer-controller-7cb85f99d7	2	2	2	9m26s

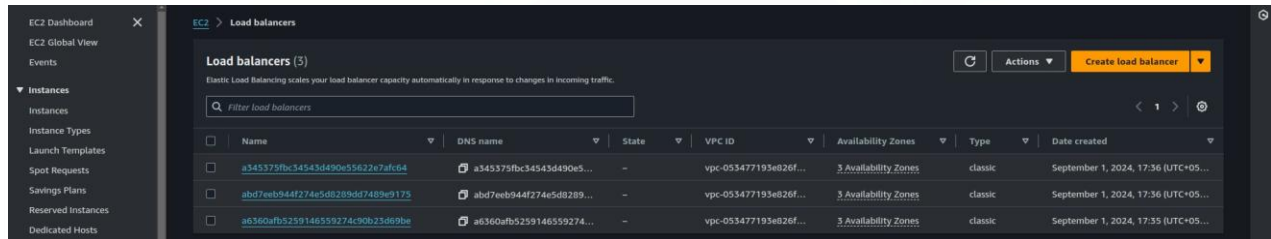
```
ubuntu@ip-10-10-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

So, we have configured the AWS Load Balancer Controller, ArgoCD, Prometheus, and Grafana through Terraform.

According to the terraform script, we have updated the service type for ArgoCD, Prometheus, and Grafana from ClusterIP to LoadBalancer.

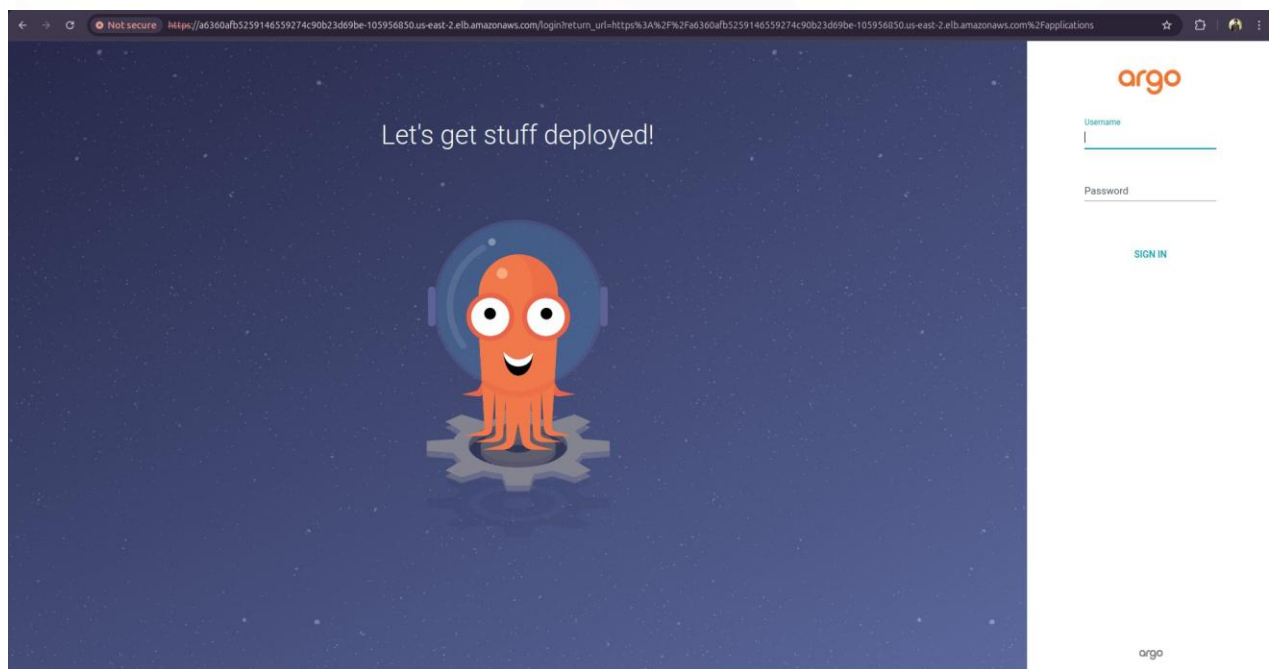
Hence, navigate to Load Balancer on your AWS account.

You can check which LB is for which tool through the kubectl command that we ran in the above steps.



Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
a345375fbc345434490e55622e7af6d4	a345375fbc345434490e5...	-	vpc-053477193e826f...	3 Availability Zones	classic	September 1, 2024, 17:36 (UTC+05...
abd7eeb944f274e5d8289dd7489e9175	abd7eeb944f274e5d8289...	-	vpc-053477193e826f...	3 Availability Zones	classic	September 1, 2024, 17:36 (UTC+05...
a6360afb5259146559274c90b23d69be	a6360afb5259146559274...	-	vpc-053477193e826f...	3 Availability Zones	classic	September 1, 2024, 17:35 (UTC+05...

So, let's try to access argoCD first. Copy the DNS and paste it into your favorite browser

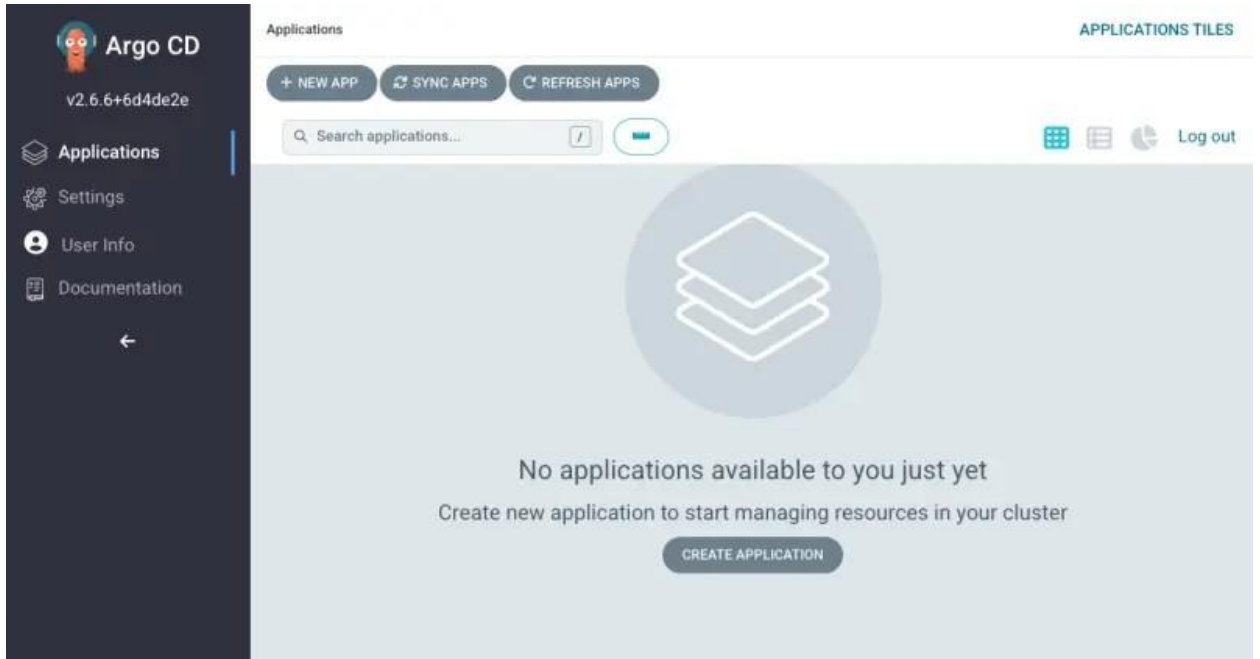


The username is admin but it needs the password which we don't know. So, we need to run a command on our ec2 server to get the password for the login

```
kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath="{.data.password}" | base64 -d
```

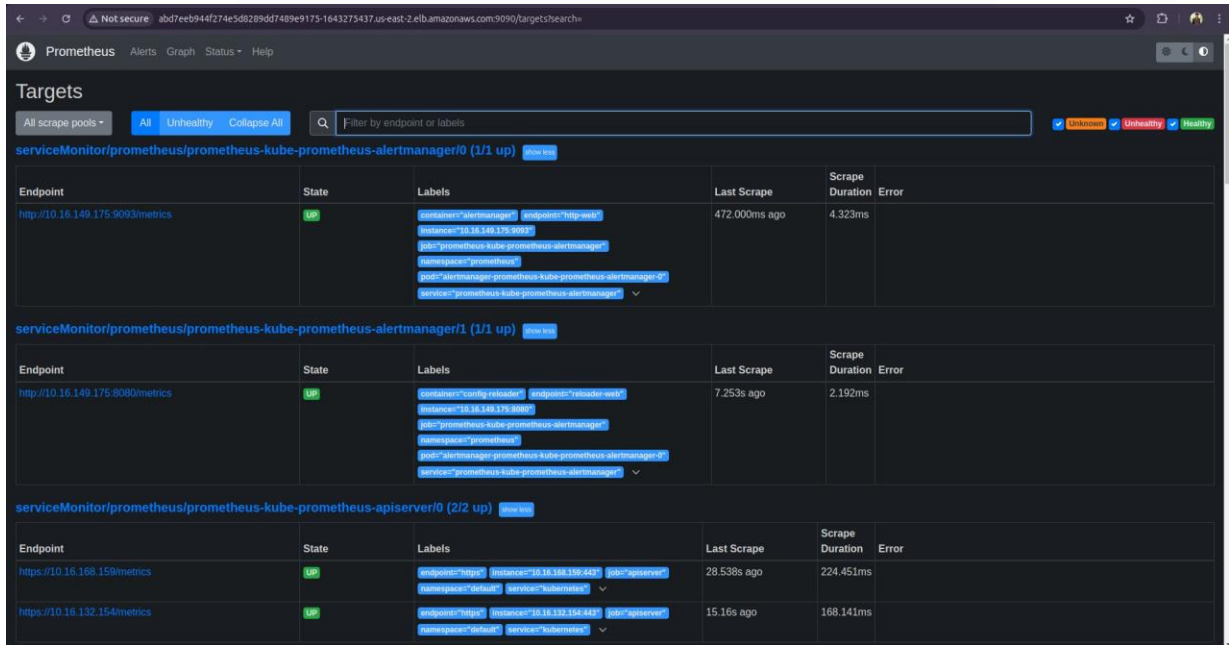
```
Session ID: Adorable-Arrow-wsg7uwnblggymvub7uof3d4 Instance ID: i-0b8a0c508e12d479c
ubuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$ kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath="{.data.password}" | base64 -d
F8fbFBxsEjHjwHmUbuntu@ip-10-16-10-26:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

It is accessible



Now, let's try to access Prometheus. Copy the DNS and paste it into your favorite browser with port 9090.

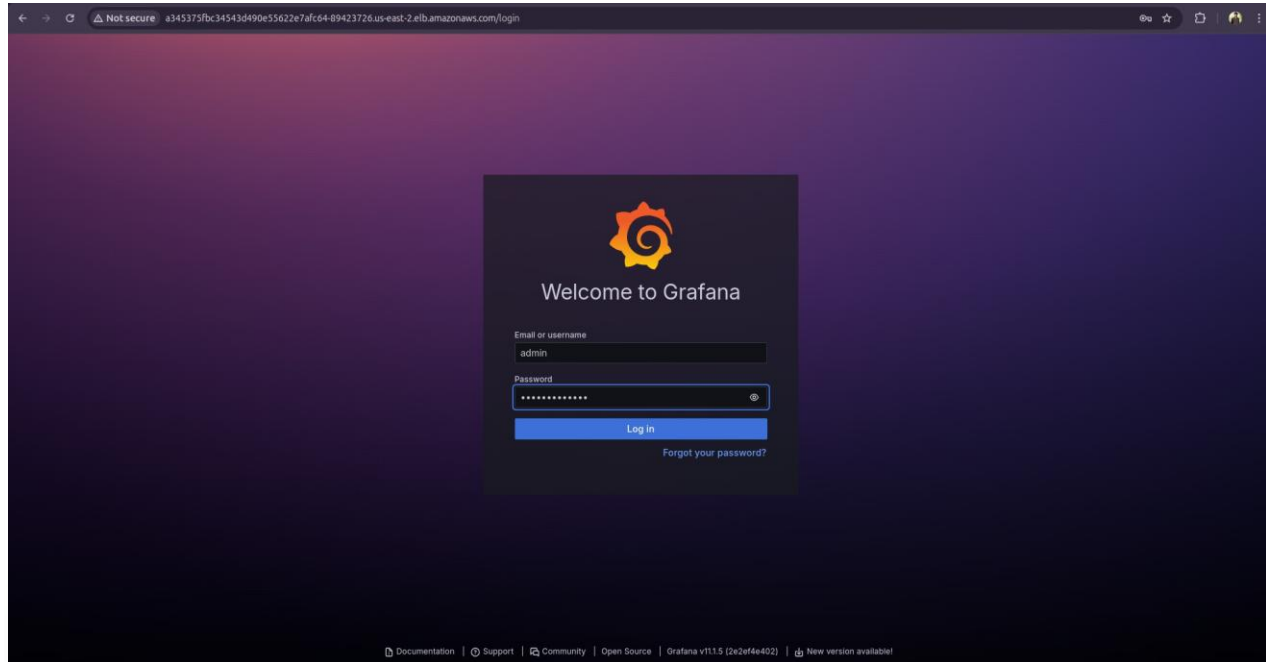
It is accessible



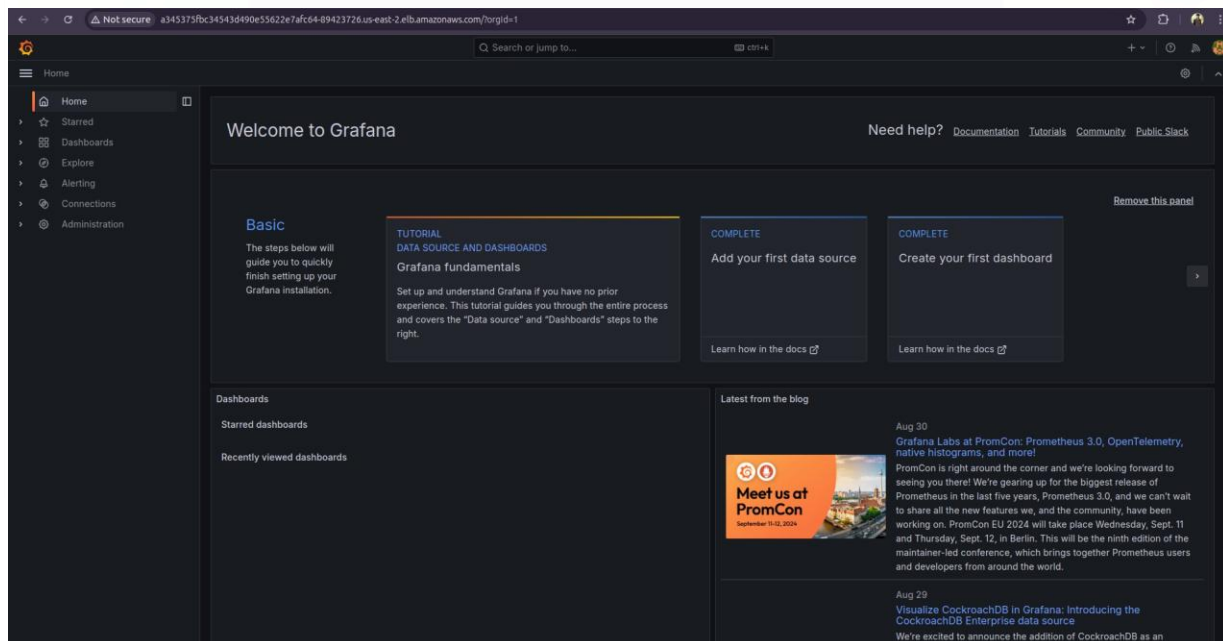
In the end, let's try to access our Grafana dashboard. Copy the DNS and paste it into your favorite browser

It is accessible.

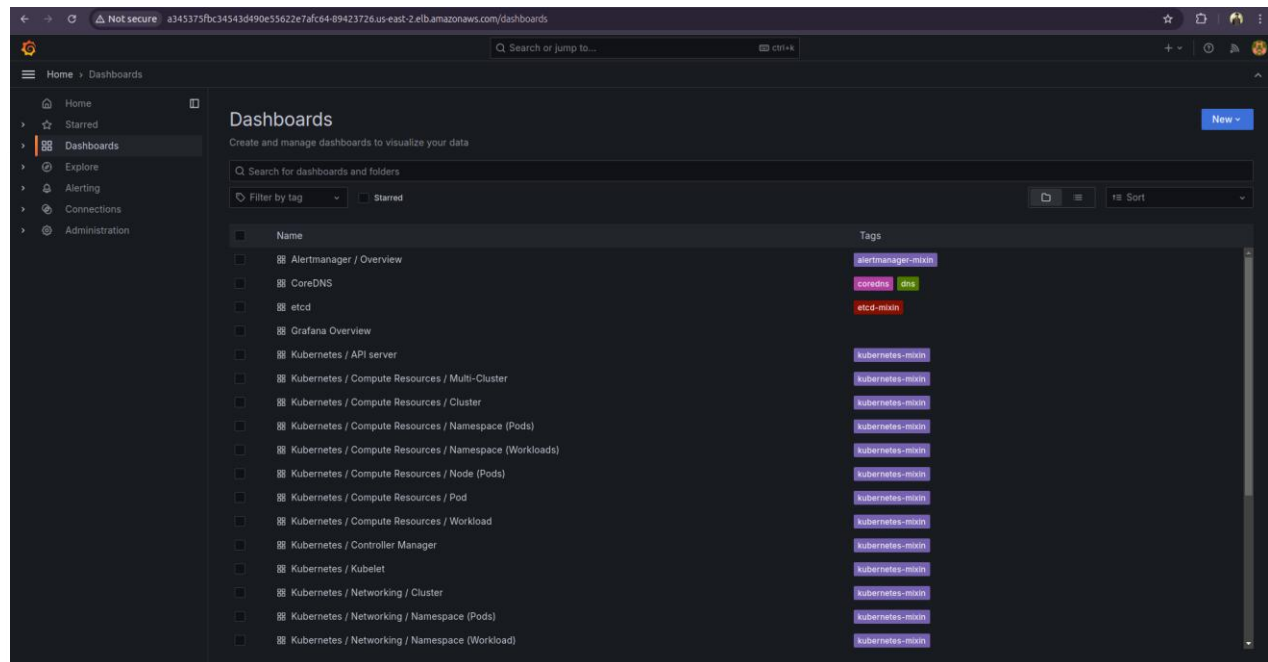
The username is admin and the password is prom-operator for the Grafana login



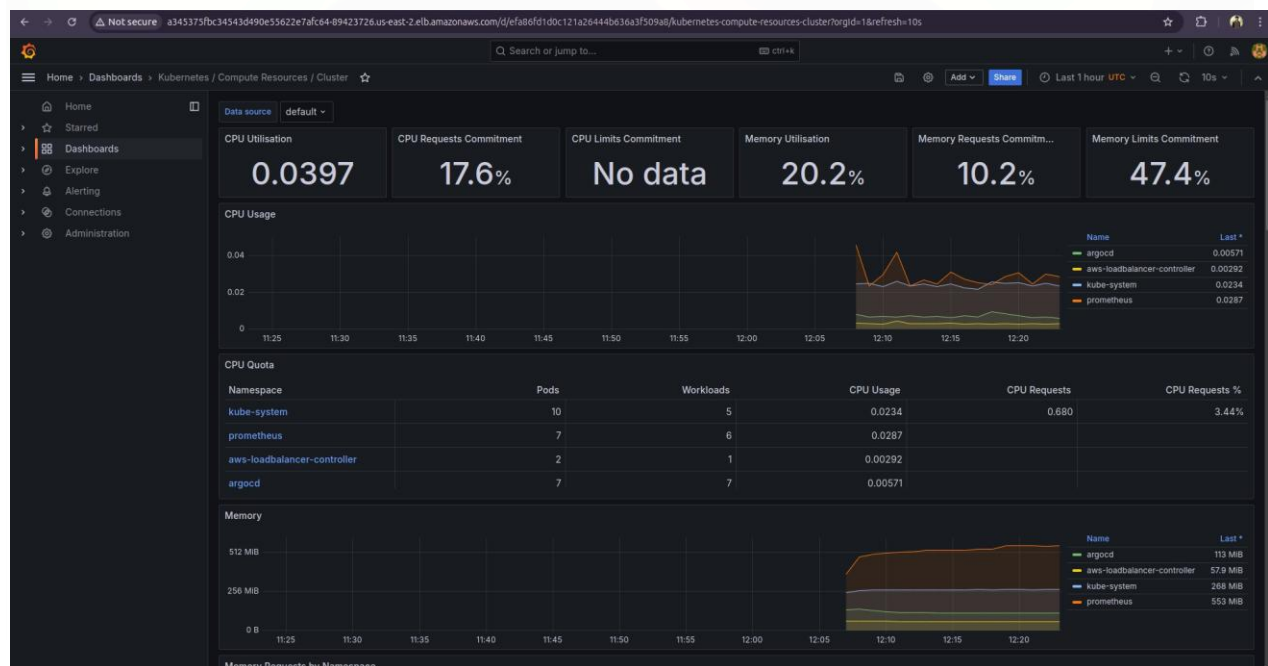
After login, the Grafana dashboard



There are a lot of dashboards already imported. You can click any of them to get insight about the EKS Cluster resources accordingly.



Here is one of them



So, we have completed our demonstration for today.

Hope you learn something new today.

To prevent the cost of cloud. Don't forget to destroy all the resources.

To do that, we need to run EKS Cluster first.

```
> terraform destroy -auto-approve -var-file=../variables.tfvars
```

```
Session ID: Aduabale-Aman-edp70wqfkggmwrb7uaf3bi Instance ID: i-0d8a3bc500812b400b Terminate

module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 40s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 50s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m0s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m10s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m20s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m30s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m40s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 1m50s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m0s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m10s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m20s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m30s elapsed]
module.eks.aws_eks_cluster.eks[0]: Still destroying... [id=dev-medium-eks-cluster, 2m40s elapsed]
module.eks.aws_eks_cluster.eks[0]: Destruction complete after 2m58s
module.eks.aws_iam_role.eks-cluster-role[0]: Destroying... [id=dev-medium-eks-cluster-role-4949]
module.eks.aws_iam_role.eks-cluster-role[0]: Destruction complete after 0s
module.eks.random_integer.random_suffix: Destroying... [id=4949]
module.eks.random_integer.random_suffix: Destruction complete after 0s

Warning: Value for undeclared variable

The root module does not declare a variable named "ec2-iam-instance-profile" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the
-compact-warnings option.

Warning: Value for undeclared variable

The root module does not declare a variable named "ec2-iam-role-policy" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the
-compact-warnings option.

Warning: Values for undeclared variables

In addition to the other similar warnings shown, 3 other variable(s) defined without being declared.

Warning: Helm uninstall returned an information message

These resources were kept due to the resource policy:
[CustomResourceDefinition] applications.argoproj.io
[CustomResourceDefinition] applications.argoproj.io
[CustomResourceDefinition] appprojects.argoproj.io

Destroy complete! Resources: 25 destroyed.
ubuntu@ip-10-10-10-20:~/EKS-ArgoCD-AWS-LB-Controller-Terraform/eks$
```

Now, destroy the vpc and an ec2 server. To do it, run the below command on your local server

```
> terraform destroy -auto-approve -var-file=../variables.tfvars
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS bash vpc-ec2 + + + + +

module.vpc-ec2.aws_instance.ec2: Destruction complete after 54s
module.vpc-ec2.aws_iam_instance_profile.ec2-instance-profile: Destroying... [id=ec2-ssm-instance-profile]
module.vpc-ec2.aws_security_group.ec2-sg: Destroying... [id=sg-071f6874b192820e]
module.vpc-ec2.aws_nat_gateway.ngw: Still destroying... [id=nat-0e920b64e401eb765, 50s elapsed]
module.vpc-ec2.aws_internet_gateway.igw: Still destroying... [id=igw-081e78f5db8daa6a4, 50s elapsed]
module.vpc-ec2.aws_iam_instance_profile.ec2-instance-profile: Destruction complete after 1s
module.vpc-ec2.aws_iam_role.iam-role: Destroying... [id=ec2-ssm-role]
module.vpc-ec2.aws_security_group.ec2-sg: Destruction complete after 2s
module.vpc-ec2.aws_iam_role.iam-role: Destruction complete after 2s
module.vpc-ec2.aws_nat_gateway.ngw: Still destroying... [id=nat-0e920b64e401eb765, 1m0s elapsed]
module.vpc-ec2.aws_internet_gateway.igw: Still destroying... [id=igw-081e78f5db8daa6a4, 1m0s elapsed]
module.vpc-ec2.aws_internet_gateway.igw: Destruction complete after 1m4s
module.vpc-ec2.aws_nat_gateway.ngw: Destruction complete after 1m10s
module.vpc-ec2.aws_elb.ngw-elb: Destroying... [id=elb-altoe-085eda06ddc0bf559]
module.vpc-ec2.aws_subnet.public-subnet[0]: Destroying... [id=subnet-00cbb711ddd8cc3bc]
module.vpc-ec2.aws_subnet.public-subnet[1]: Destroying... [id=subnet-0b600ba0303c3a40f]
module.vpc-ec2.aws_subnet.public-subnet[2]: Destroying... [id=subnet-0529f28936b2ded3c]
module.vpc-ec2.aws_subnet.public-subnet[0]: Destruction complete after 1s
module.vpc-ec2.aws_subnet.public-subnet[1]: Destruction complete after 2s
module.vpc-ec2.aws_subnet.public-subnet[2]: Destruction complete after 3s
module.vpc-ec2.aws_vpc.vpc: Destroying... [id=vpc-053477193e826fec2]
module.vpc-ec2.aws_vpc.vpc: Destruction complete after 1s

Warning: Value for undeclared variable

The root module does not declare a variable named "endpoint-public-access" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the
configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the
-compact-warnings option.

Warning: Value for undeclared variable

The root module does not declare a variable named "desired_capacity_on_demand" but a value was found in file "../variables.tfvars". If you meant to use this value, add a "variable" block to the
configuration.

To silence these warnings, use TF_VAR_... environment variables to provide certain "global" settings to all configurations in your organization. To reduce the verbosity of these warnings, use the
-compact-warnings option.

Warning: Values for undeclared variables

In addition to the other similar warnings shown, 11 other variable(s) defined without being declared.

Releasing state lock. This may take a few moments...

Destroy complete! Resources: 24 destroyed.
```

I would recommend you to read the eks terraform files to get a better understanding of the resources.

Conclusion

In this comprehensive guide, we've explored how to deploy a private EKS cluster on AWS and configure essential Kubernetes tools such as ArgoCD, Prometheus, and Grafana using Terraform. By following these steps, you can efficiently manage your infrastructure and ensure that your applications are running smoothly in a secure, scalable environment. Remember to clean up your resources after the demonstration to avoid unnecessary costs. Continuous learning and hands-on practice are key to mastering these DevOps practices, so keep experimenting with different configurations and tools to enhance your skills.

