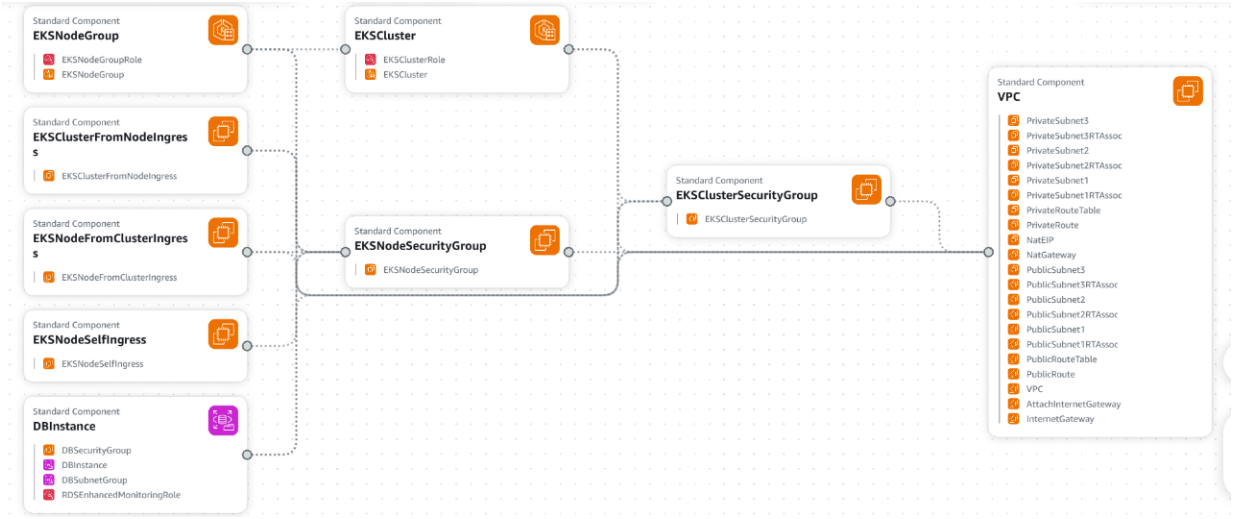
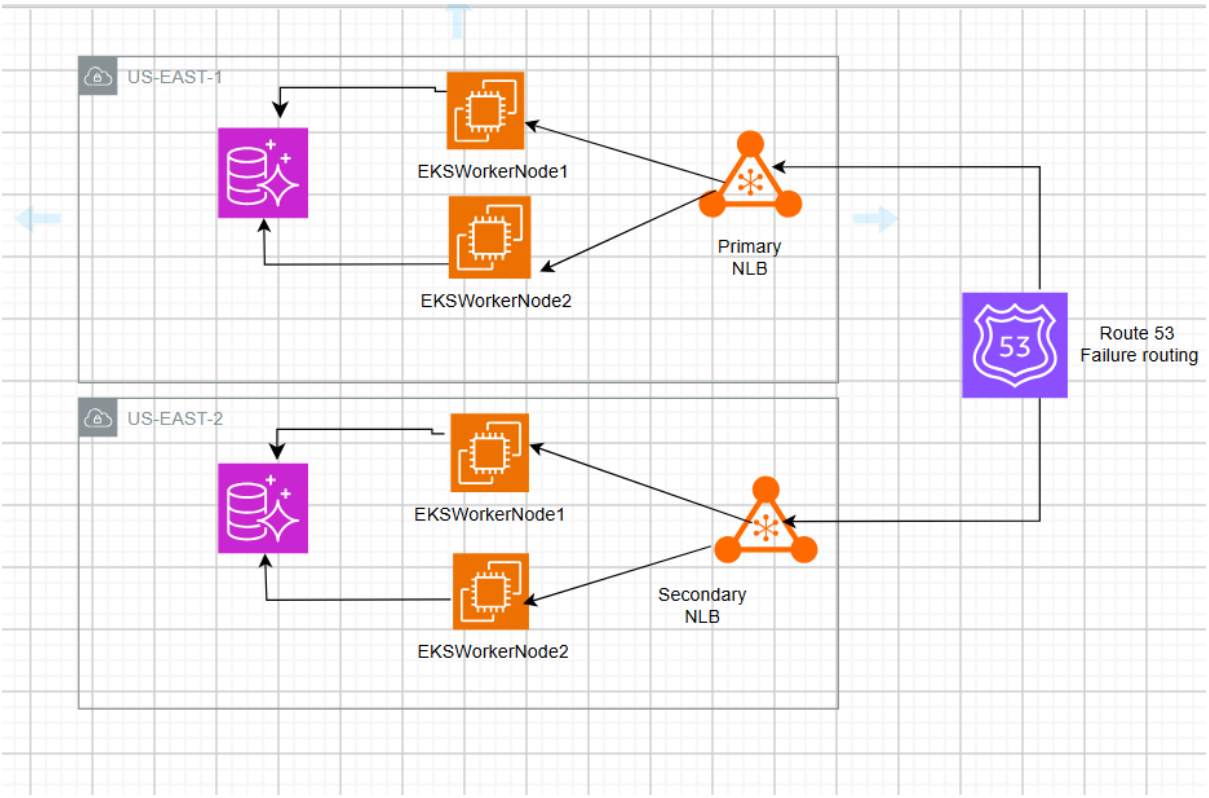
INFRASTRUCTURE

Infrastructure resources created through cloud formation and terraform (infracstructure seen in cloud formation infrastructure composer)



Overall infrastructure with route53 failover routing



Git repo containing the whole code

https://github.com/SelmiNazeeb/FinalProject-Devops.git

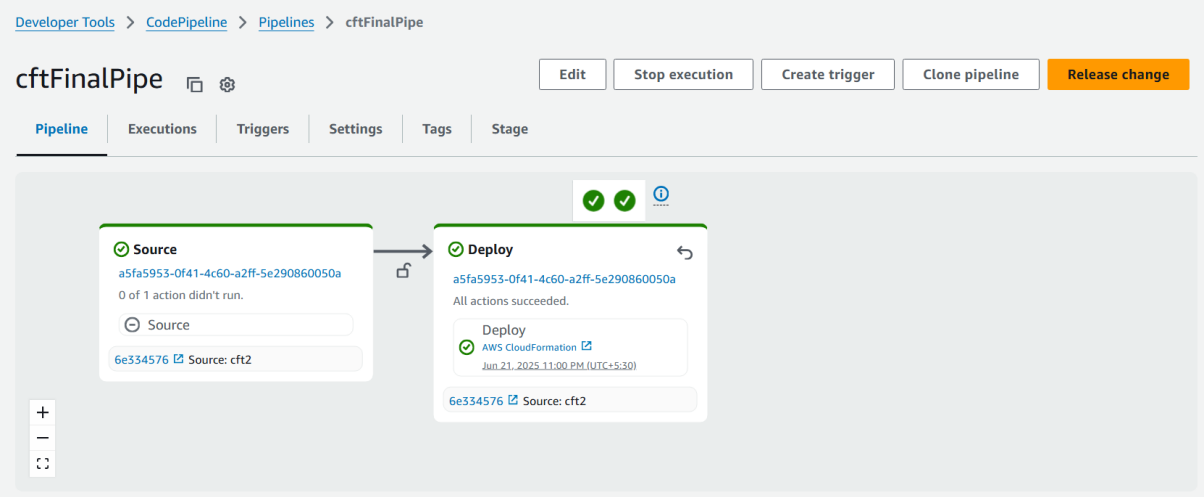
TASK1 :

CREATING INFRASTRUCTURE THROUGH CLOUDFORMATION IN REGION1 (US-EAST-1)

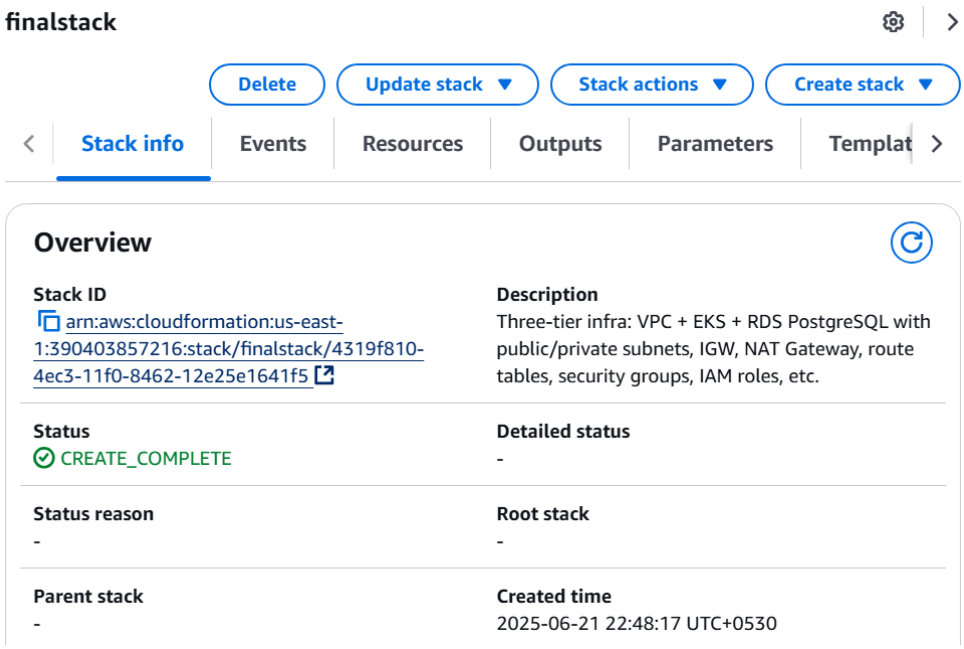
Permissions needed

- Cloud formation role – full admin access
- Pipeline role - AmazonEKSClusterPolicy, AWSCloudFormationFullAccess

Pipeline of infrastructure creation : success



Cloud formation stack created through pipeline



Resources created through stack

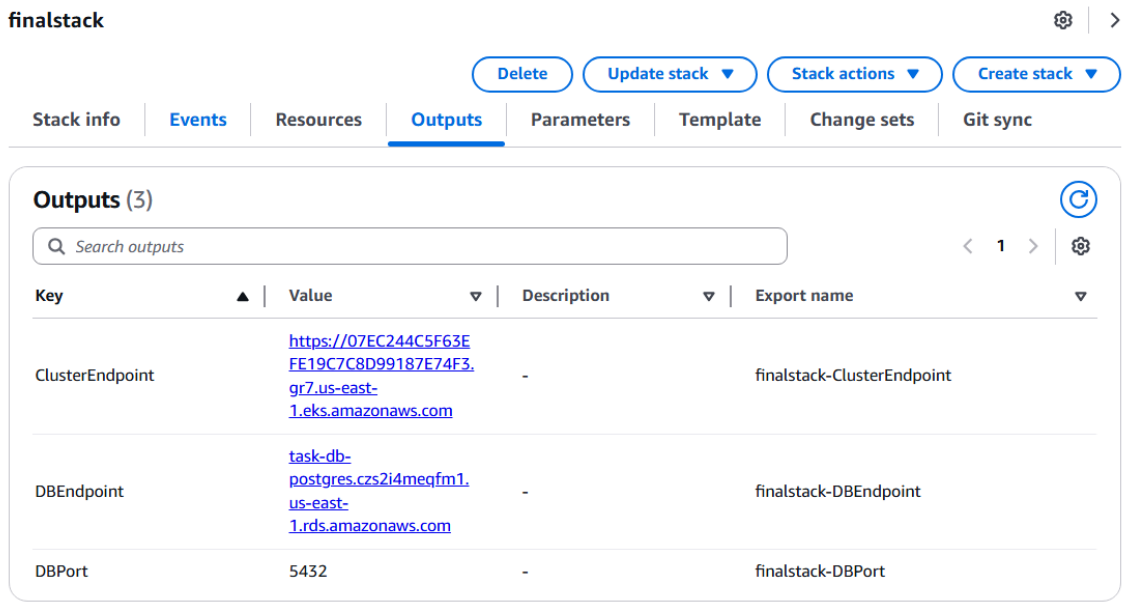
Resources (34)

Search resources

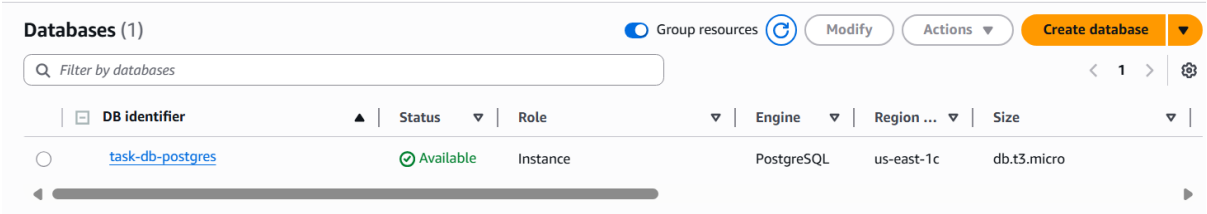
1

Logical ID	Physical ID	Type	Status	Module
AttachInternetGateway	IGW|vpc-0a15dad6dfa654dd5	AWS::EC2::VPCGatewayAttachment	CREATE_COMPLETE	-
DBInstance	task-db-postgres	AWS::RDS::DBInstance	CREATE_COMPLETE	-
DBSecurityGroup	sg-0b884508425083b44	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-
DBSubnetGroup	task-db-postgres-subnet-group	AWS::RDS::DBSubnetGroup	CREATE_COMPLETE	-
EKSCluster	three-tier-eks-cluster	AWS::EKS::Cluster	CREATE_COMPLETE	-
EKSClusterFromNodeIngress	sgr-05c25fc400cd957fb	AWS::EC2::SecurityGroupIngress	CREATE_COMPLETE	-
EKSClusterRole	finalstack-EKSClusterRole-3nVJOcCjuPVL	AWS::IAM::Role	CREATE_COMPLETE	-
EKSClusterSecurityGroup	sg-03d366fdb055a8abf	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-
EKSNodeFromClusterIngress	sgr-0a9b238f8163ef5b0	AWS::EC2::SecurityGroupIngress	CREATE_COMPLETE	-
EKSNodeGroup	three-tier-eks-cluster/app-node-group	AWS::EKS::Nodegroup	CREATE_COMPLETE	-
EKSNodeGroupRole	finalstack-EKSNodeGroupRole-pgB2ZjUW4tVs	AWS::IAM::Role	CREATE_COMPLETE	-
EKSNodeSecurityGroup	sg-079b9e86378fbb1af	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-
EKSNodeSelfIngress	sgr-0aa0fad27f71bd69e	AWS::EC2::SecurityGroupIngress	CREATE_COMPLETE	-
InternetGateway	igw-021f3dab7cfea1f0b	AWS::EC2::InternetGateway	CREATE_COMPLETE	-
PrivateSubnet2RTAssoc	rtbassoc-0ed8252df0690f5f0	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE	-
PrivateSubnet3	subnet-0bce62677f3a6f609	AWS::EC2::Subnet	CREATE_COMPLETE	-
PrivateSubnet3RTAssoc	rtbassoc-00c3d839b8c3b5875	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE	-
PublicRoute	rtb-04f9c8025c03e2860|0.0.0.0/0	AWS::EC2::Route	CREATE_COMPLETE	-
PublicRouteTable	rtb-04f9c8025c03e2860	AWS::EC2::RouteTable	CREATE_COMPLETE	-
PublicSubnet1	subnet-06ee189155b033a24	AWS::EC2::Subnet	CREATE_COMPLETE	-
PublicSubnet1RTAssoc	rtbassoc-0ea7d0da7a5b0ca1f	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE	-
PublicSubnet2	subnet-04b6cd1ed6eacdbbd	AWS::EC2::Subnet	CREATE_COMPLETE	-
PublicSubnet2RTAssoc	rtbassoc-0fe1c507ce20148ed	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE	-
PublicSubnet3	subnet-02e0c106269a659a6	AWS::EC2::Subnet	CREATE_COMPLETE	-
PublicSubnet3RTAssoc	rtbassoc-0aee15f2ed39589a5	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE	-
RDSEnhancedMonitoringRole	finalstack-RDSEnhancedMonitoringRole-m7vEBzEzbQLT	AWS::IAM::Role	CREATE_COMPLETE	-
VPC	vpc-0a15dad6dfa654dd5	AWS::EC2::VPC	CREATE_COMPLETE	-

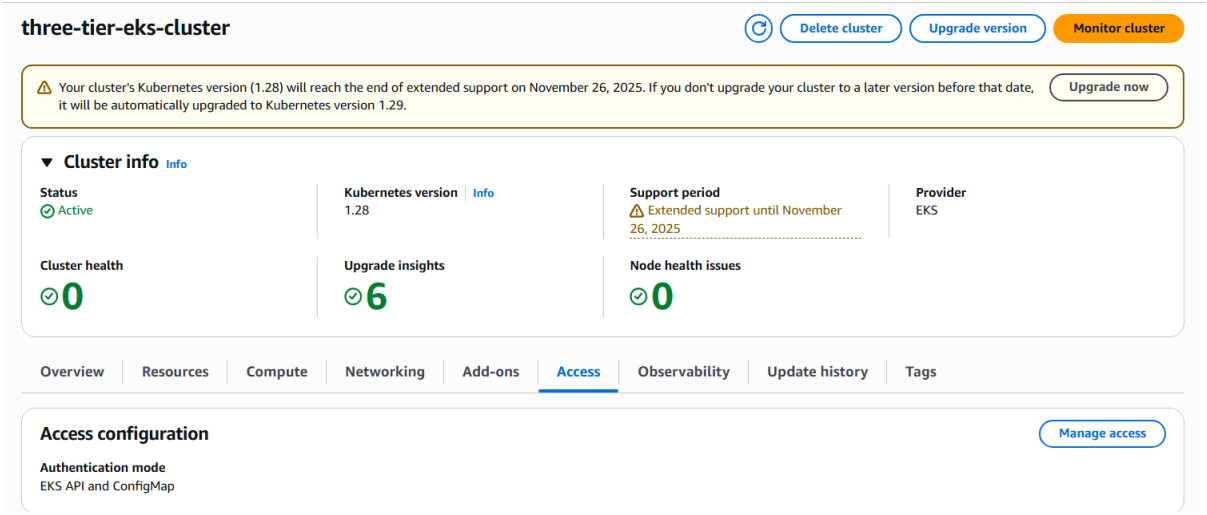
Output in cloud formation stack



Postgresql Database



EKS cluster



Permission to user to access EKS cluster

Select EKS cluster -> Access -> Add access entry -> give user arn -> next -> add “AmazonEKSClusterAdminPolicy “ -> save

arn:aws:iam::390403857216:user/terraform-user Standard arn:aws:iam::390403857216:user/terraform-user - AmazonEKSClusterAdminPolicy

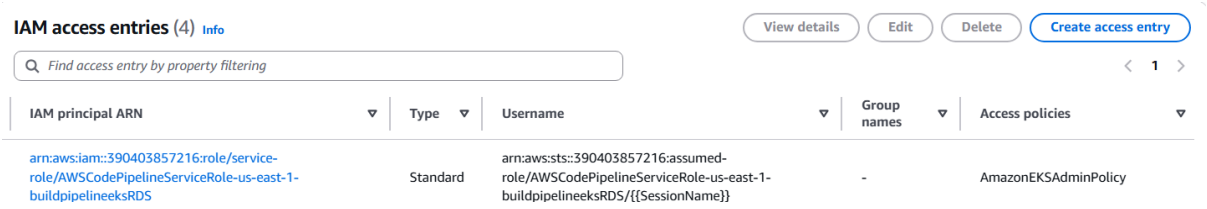
TASK 2:

DEPLOYING APPLICATION ON THE INFRASTUCTURE CREATED BY CLOUD FORMATION

Permissions needed

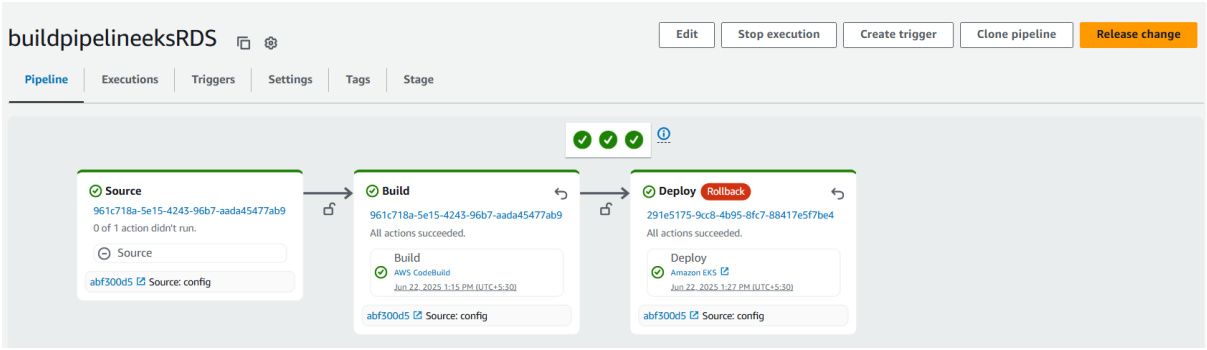
- pipeline role – Access entry in EKS cluster

Select EKS cluster -> Access -> Add access entry -> give pipeline role arn -> next -> add “AmazonEKSAdminPolicy “ -> save



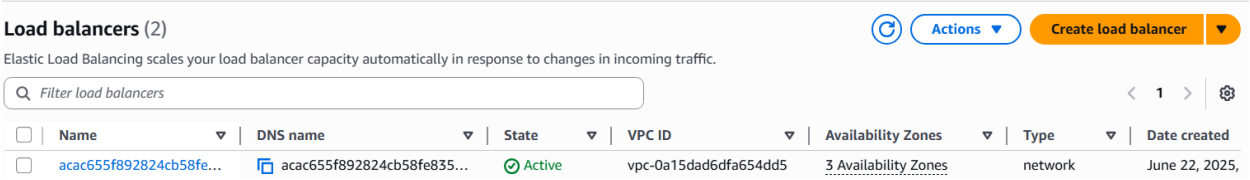
- Code build role - AmazonEC2ContainerRegistryPowerUser

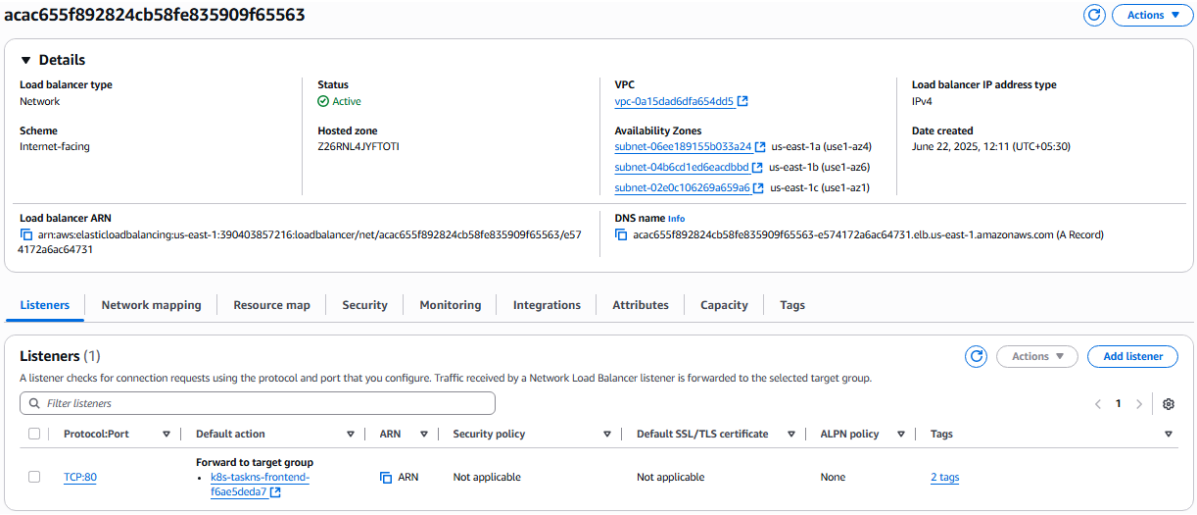
Success pipeline of application deployment



>>> Now all containers, pods, deployment and services are running

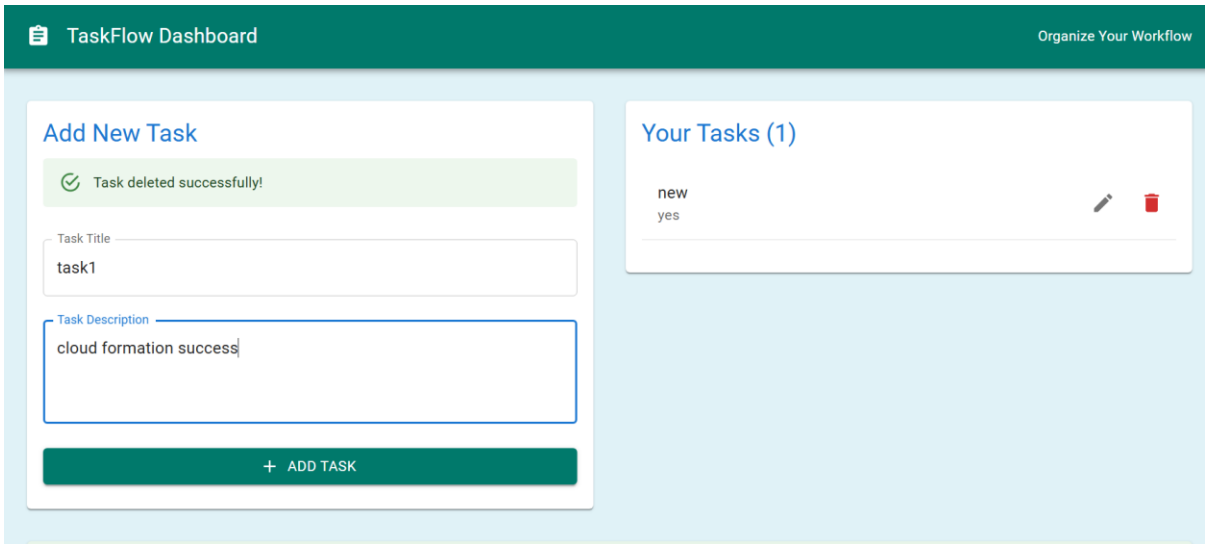
Load Balancer created for the application



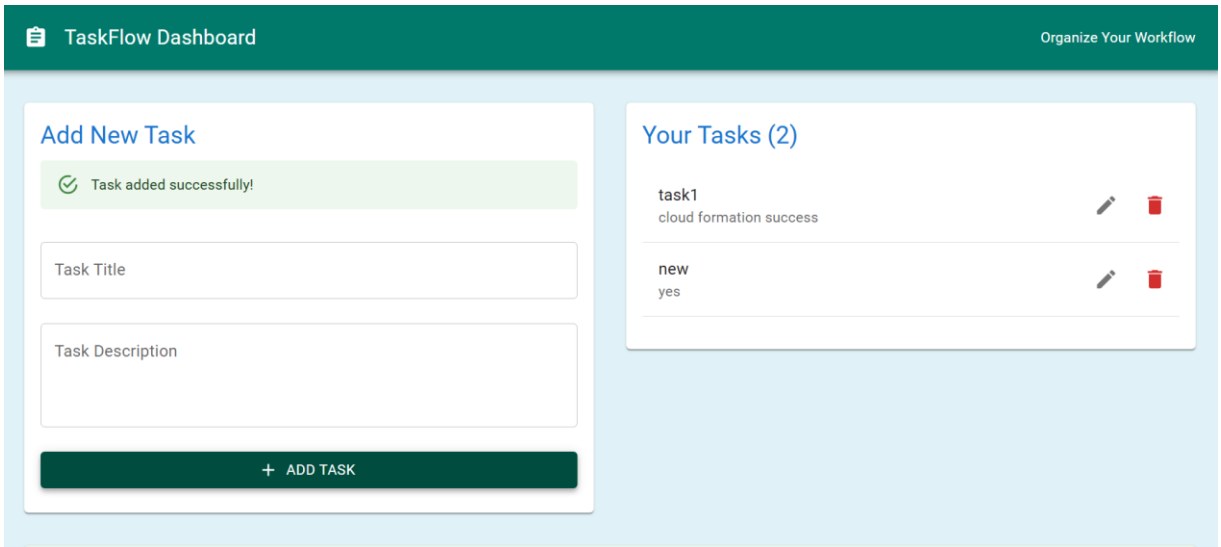


In Browser – paste load balancer arn **(Application is working)**

Frontend of Task management APP – Adding a new task (task1 – give title and description -> add task)



Added task and stored in Database



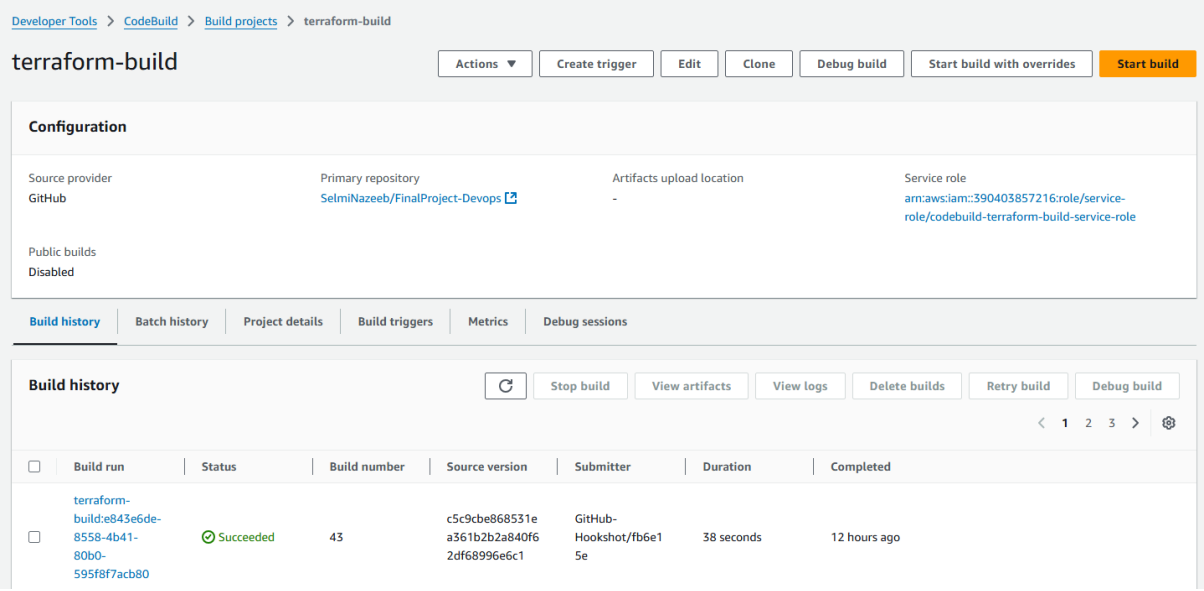
TASK3 :

CREATING INFRASTRUCTURE THROUGH TERRAFORMIN REGION2 (US-EAST-2)

Permissions needed

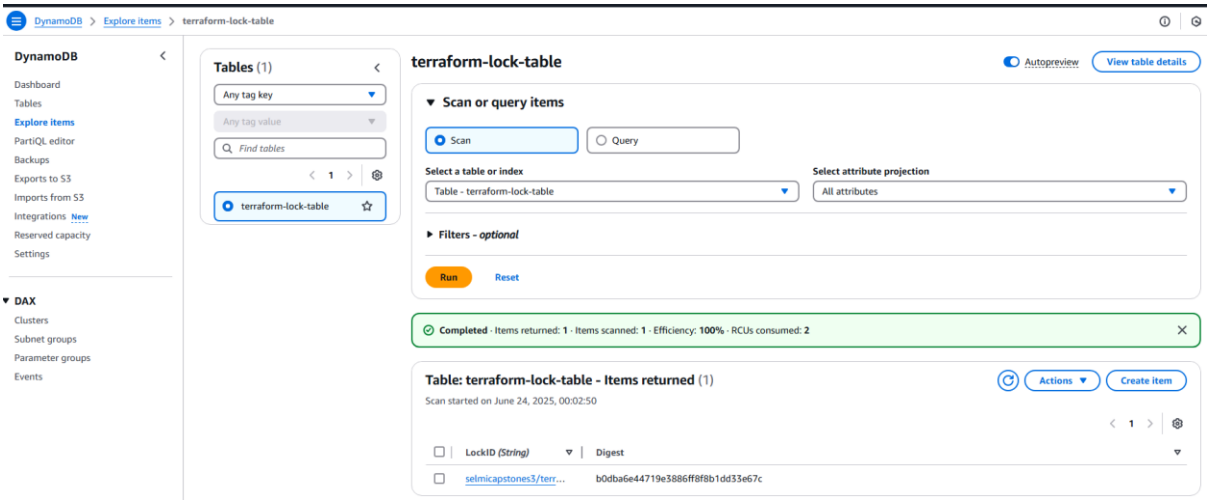
- Code build role - policy should be attached (added in github -> permissions/terraform-build-policy)

Infrastructure creation : success

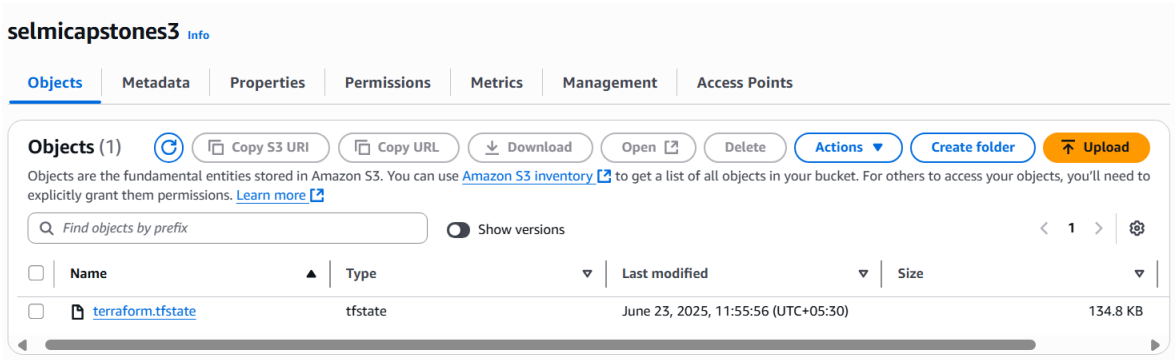


Dynamo DB table and S3 bucket for storing terraform tf state storage of code build of infrastructure

Dynamo DB



S3 bucket

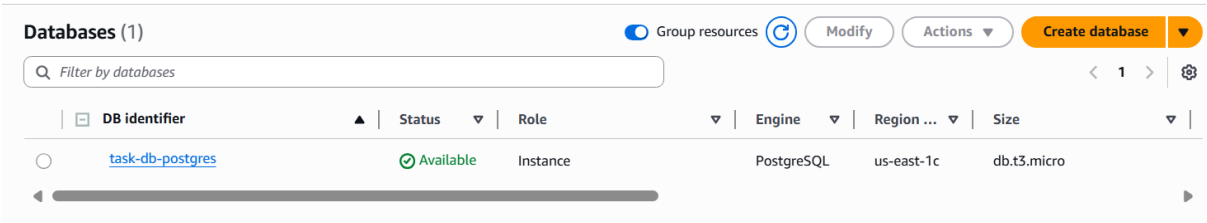


Outputs

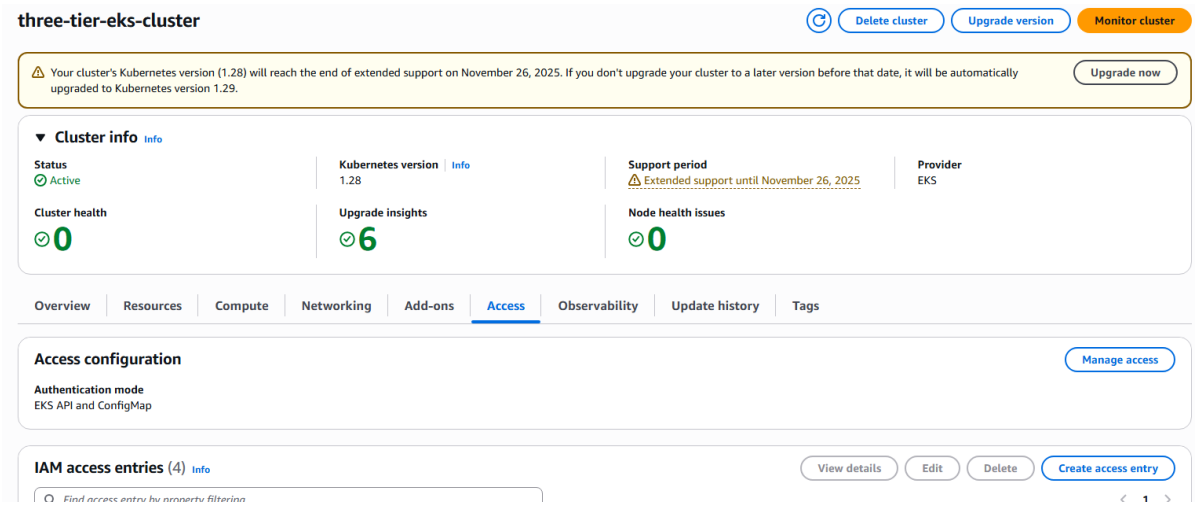
```
Outputs:

eks_cluster_endpoint = "https://7627E37233A3CF883ECD615F9D063276.gr7.us-east-2.eks.amazonaws.com"
eks_cluster_security_group_id = "sg-020b2b5f3047cfc66"
rds_endpoint = "task-db-postgres.czus4mukos81.us-east-2.rds.amazonaws.com:5432"
vpc_id = "vpc-02b566a9f4f50daec"
```

Resources created

RDS



EKS cluster



Permission to user to access EKS cluster

Select EKS cluster -> Access -> Add access entry -> give user arn -> next -> add “AmazonEKSClusterAdminPolicy “ -> save

arn:aws:iam::390403857216:user/terraform-user	Standard	arn:aws:iam::390403857216:user/terraform-user	-	AmazonEKSClusterAdminPolicy

TASK 4:

DEPLOYING APPLICATION ON THE INFRASTUCTURE CREATED BY TERRAFORM

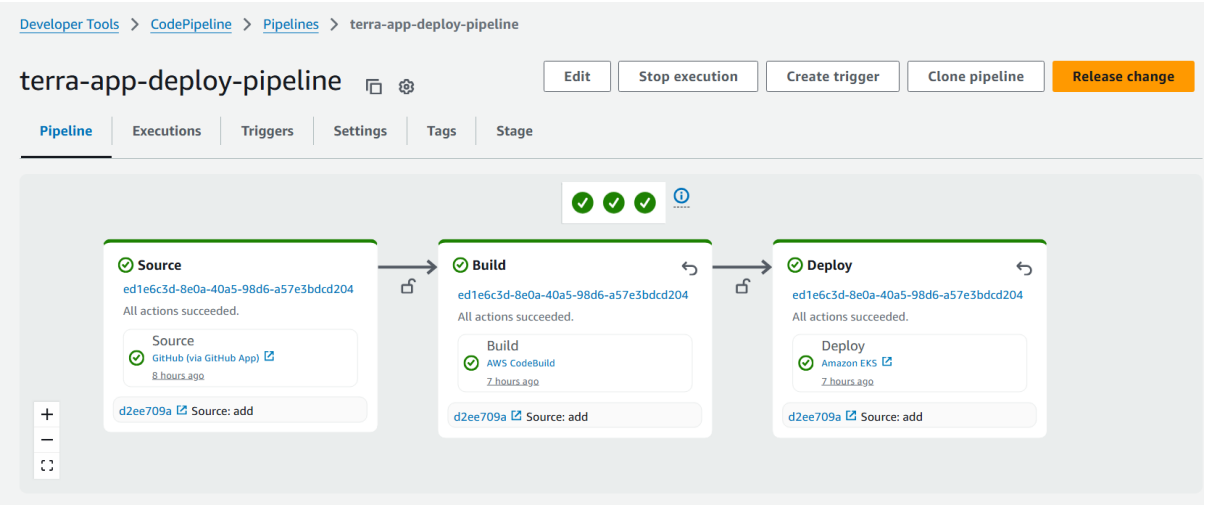
Permissions needed

- pipeline role – Access entry in EKS cluster

Select EKS cluster -> Access -> Add access entry -> give pipeline role arn -> next -> add “AmazonEKSAdminPolicy “ -> save

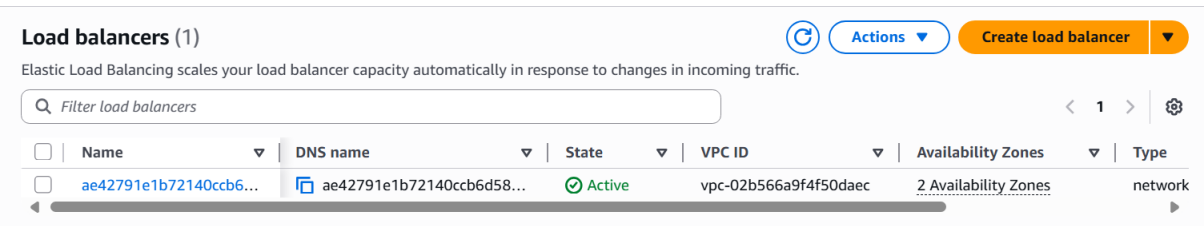
- Code build role - AmazonEC2ContainerRegistryPowerUser

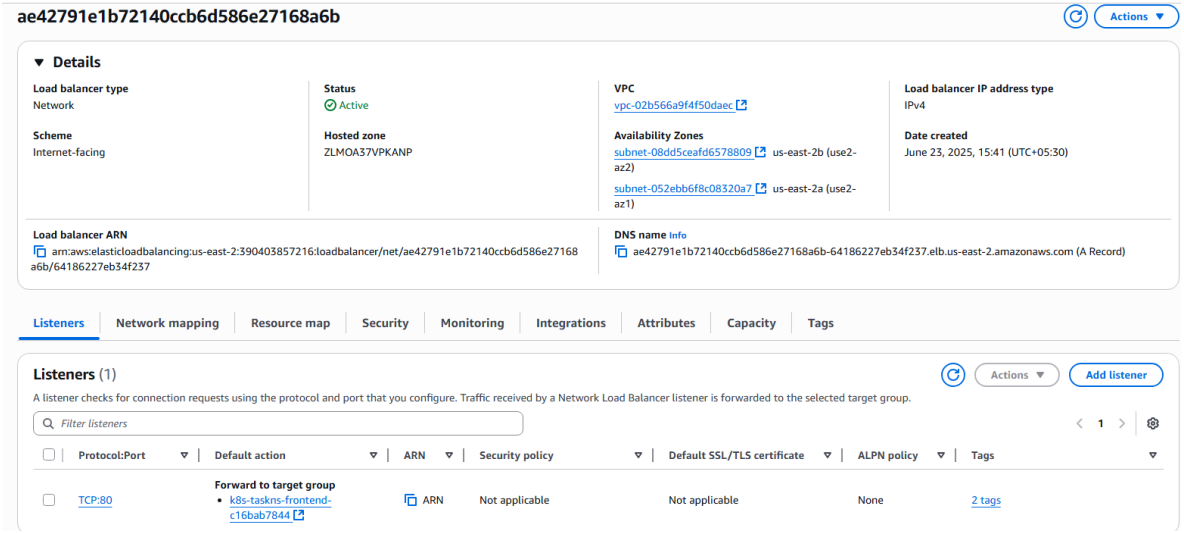
Success pipeline of application deployment



>>> Now all containers, pods, deployment and services are running

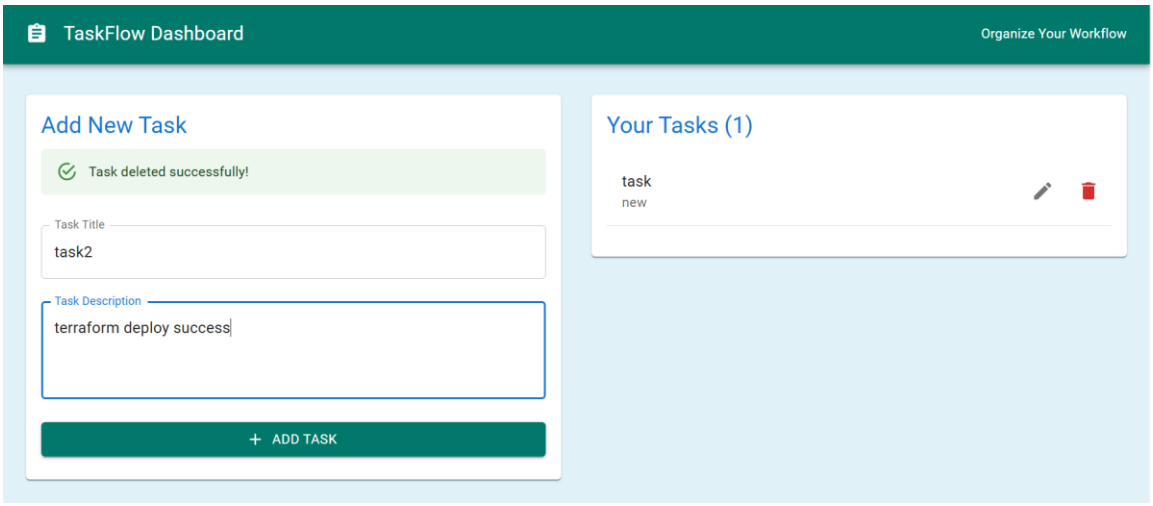
Load Balancer created for the application



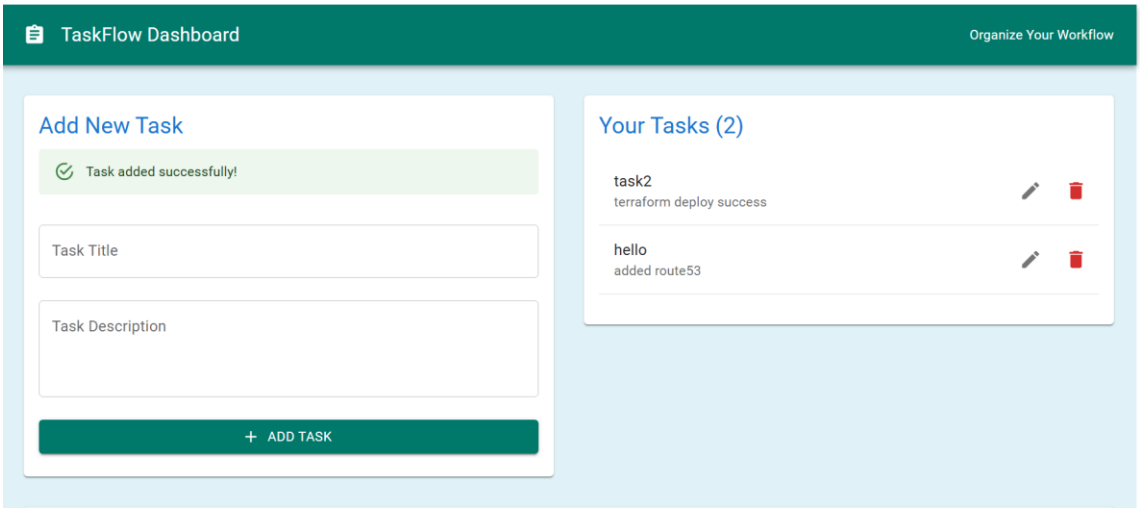


In Browser – paste load balancer arn **(Application is working)**

Frontend of Task management APP – Adding a new task (task2 – give title and description -> add task)



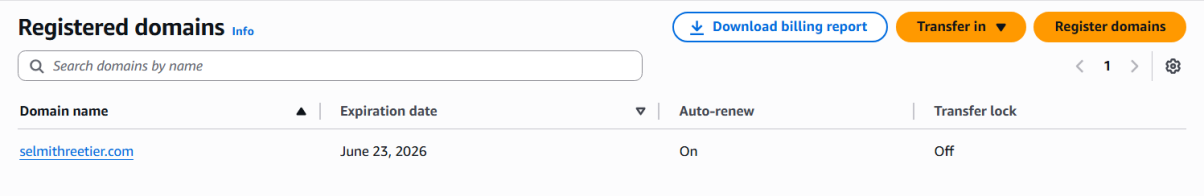
Added task and stored in Database



TASK 5 :

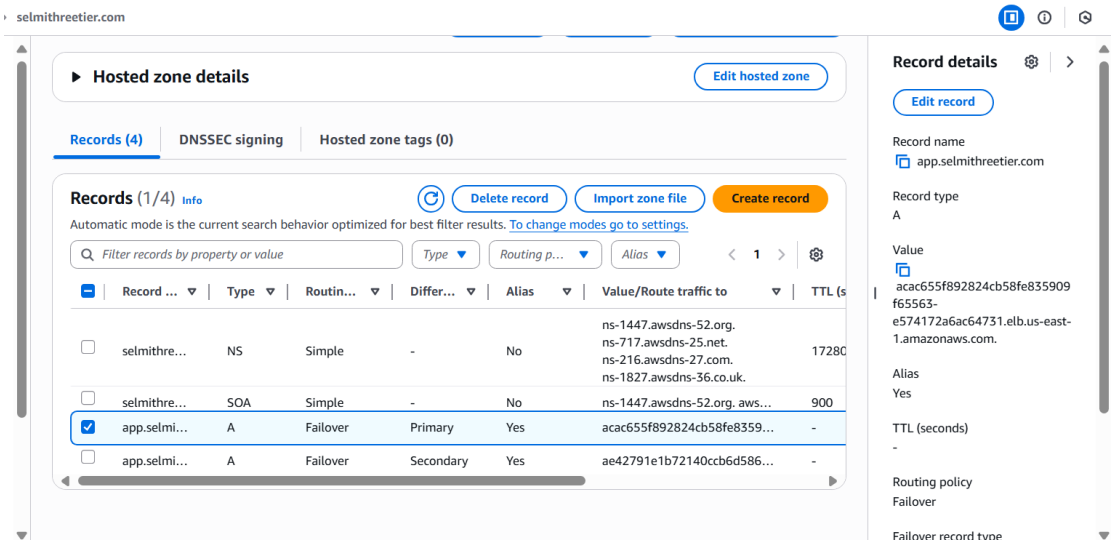
ROUTE 53 FOR FAILURE ROUTING

Domain Name : selmithreetier.com



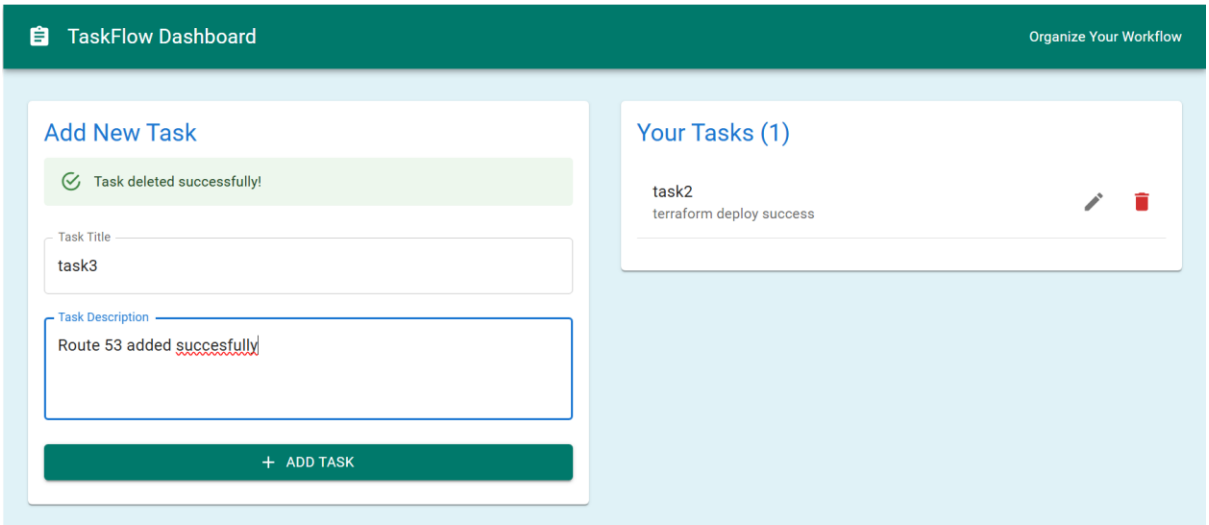
Creating a failure routing A record to public hosted zone

- Set primary load balancer as LB in region1 and secondary as LB in region2
- If primary load balancer fails routes the traffic to secondary load balancer

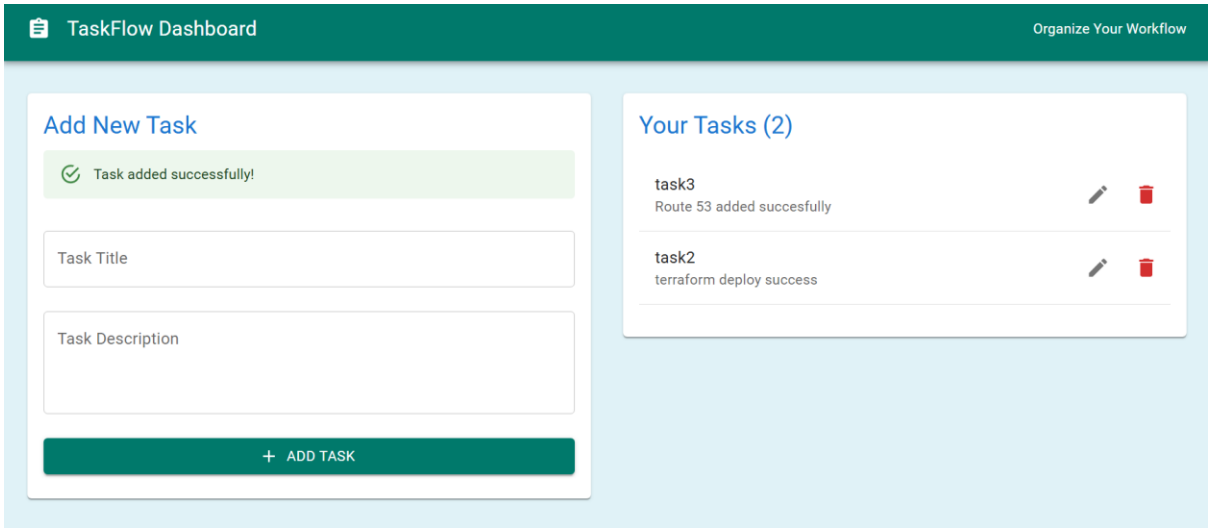


Browse – “app.selmithreetier.com”

Frontend of Task management APP – Adding a new task (task3 – give title and description -> add task)



Added task and stored in Database



Deleted one stored value