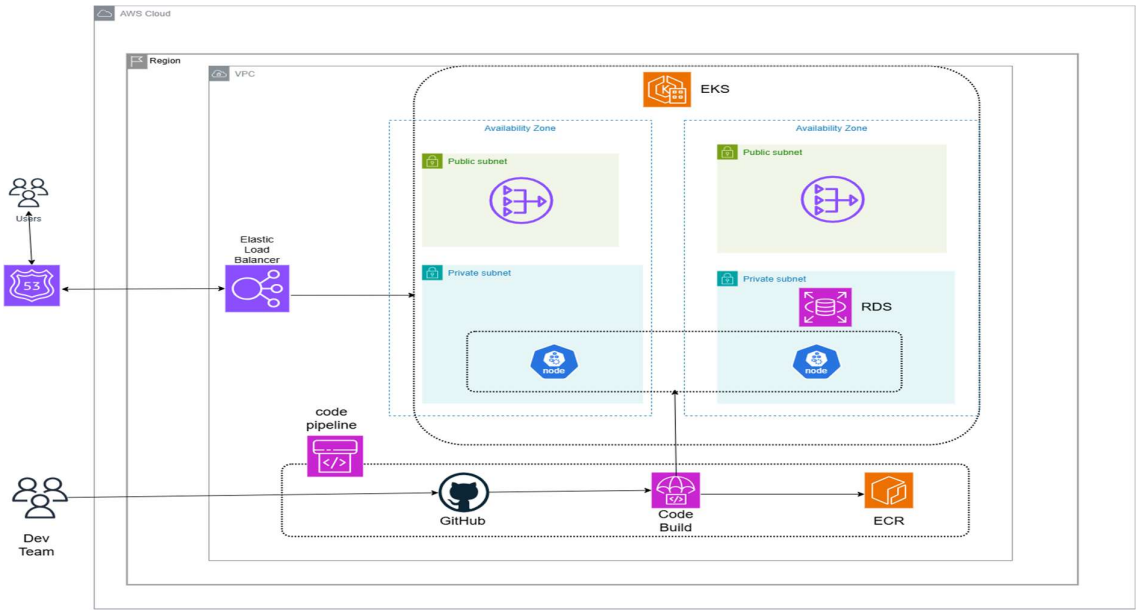
AWS Capstone Project – Deployment and Automation using CloudFormation and Terraform

📌 Project Overview

This project demonstrates a complete CI/CD pipeline deployment of a 3-tier web application (frontend, backend, database) using AWS services. The application is deployed in two different regions using CloudFormation and Terraform for infrastructure provisioning. Key DevOps tools integrated include AWS CodePipeline, CodeBuild, SonarQube, and CloudWatch.

🌐 Architecture Diagram



🌍 Region 1: eu-north-1 – Deployed via AWS CloudFormation

1️⃣ Infrastructure Provisioning

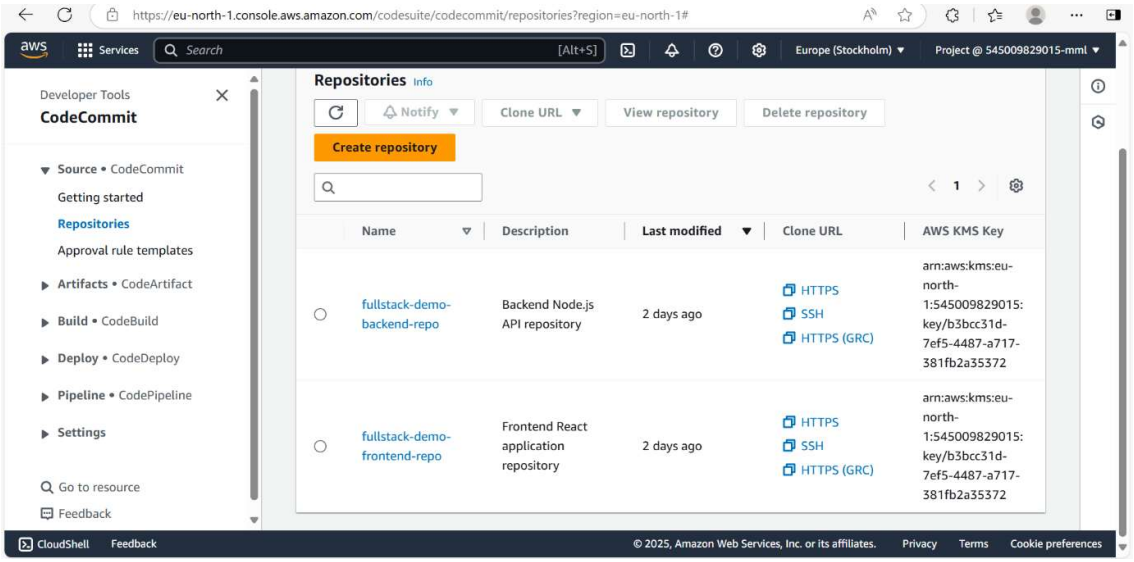
Infrastructure is provisioned using the following CloudFormation stack:
GitHub Repo:

https://github.com/AnushreeGM/AWS_CapstoneProject/blob/main/infrastructure/infrastructure.yaml

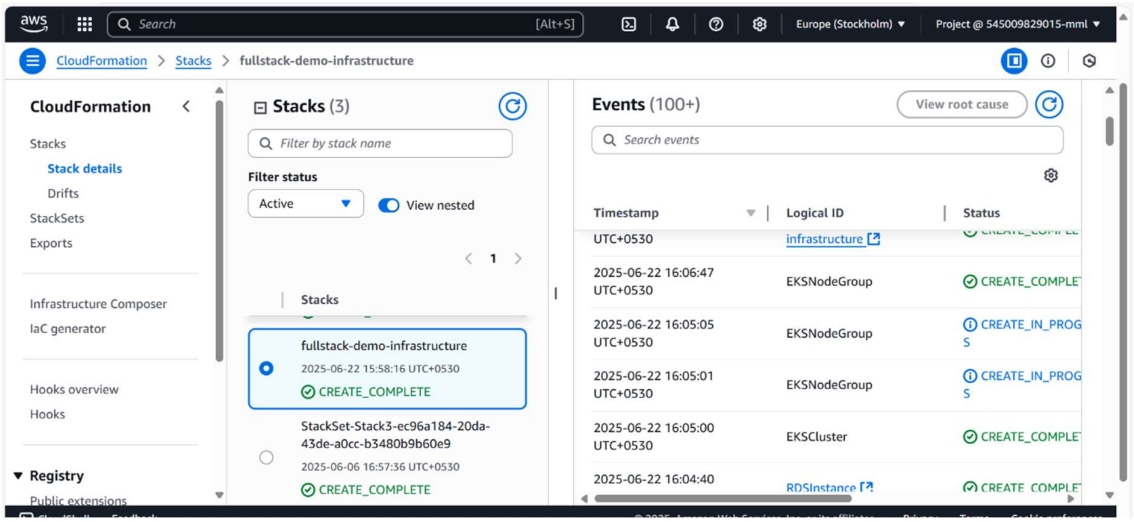
Resources Created:
- VPC with public/private subnets
- EKS Cluster
- RDS MySQL DB
- IAM roles and policies
- ALB and Security Groups

2 Source Control

All code is committed and maintained in AWS CodeCommit for source control.



Create infra using cloud formation



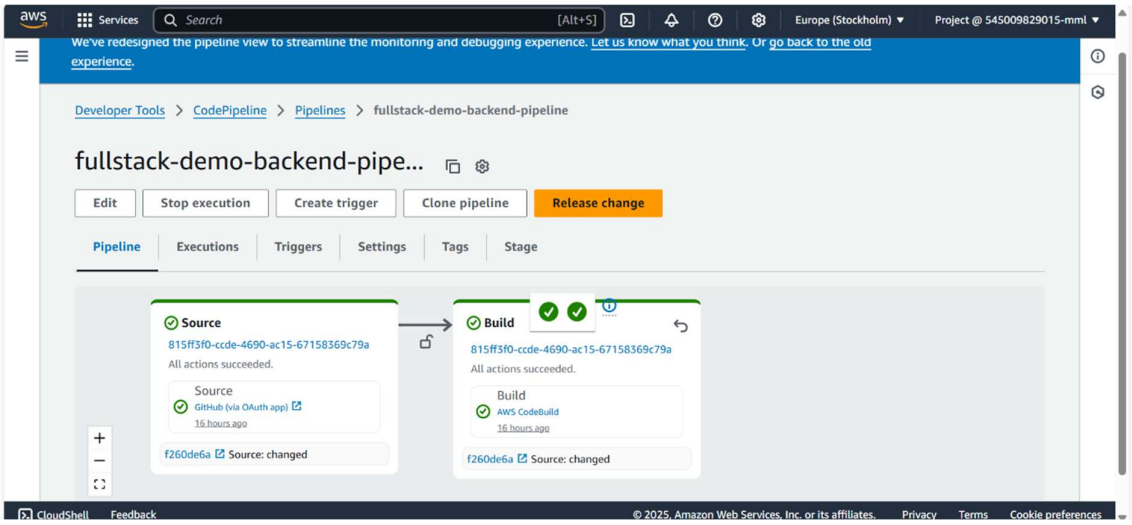
3 Application Deployment

Frontend Repo: https://github.com/AnushreeGM/AWS_CapstoneProject/tree/main/frontend
Backend Repo: https://github.com/AnushreeGM/AWS_CapstoneProject/tree/main/backend

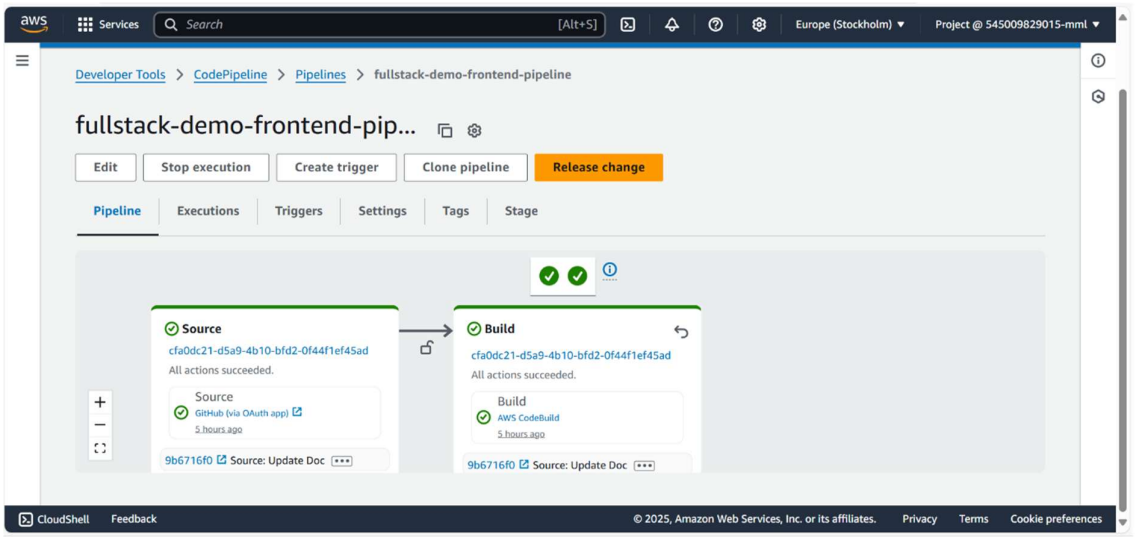
Deployment is triggered via AWS CodePipeline for both frontend and backend. Each pipeline includes CodeBuild, Docker image build, and EKS deployment stages.

Deploy the application through pipeline

Backend pipeline



Frontend pipeline



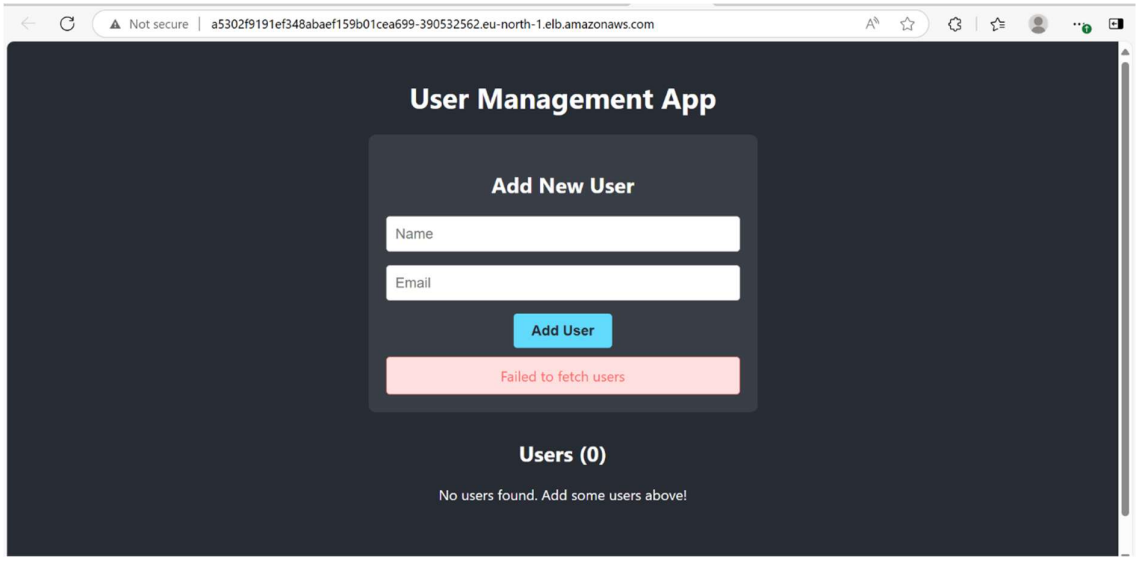
Application Validation

✅ Verification of Running Pods – Confirmed with `kubectl get pods`.

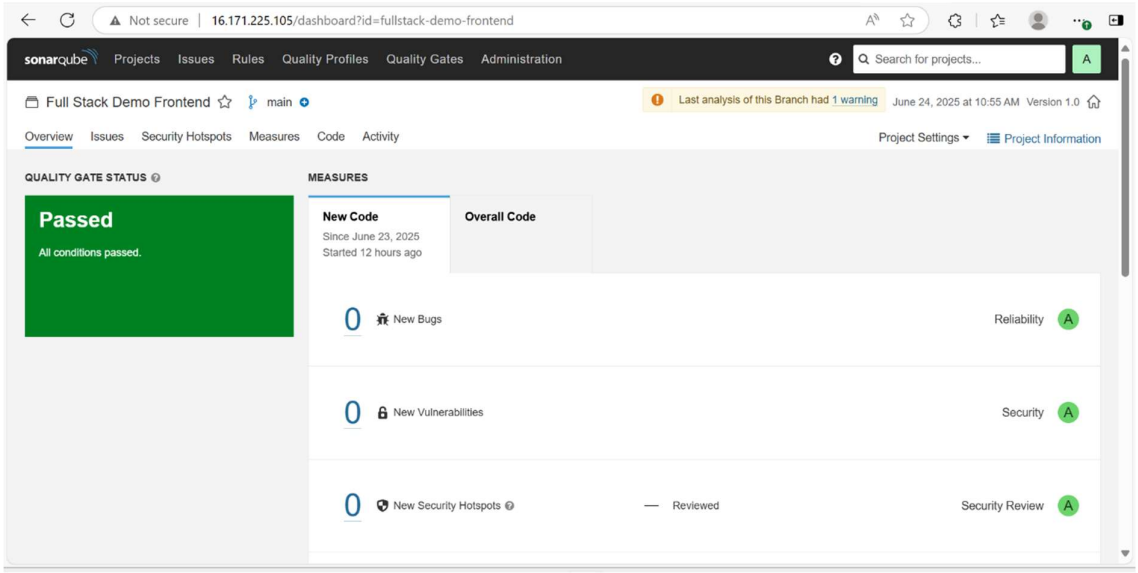
```
Default output format [None]:
ubuntu@ip-172-31-39-189:~$ aws eks update-kubeconfig --region eu-north-1 --name fullstack-demo-eks-cluster
Added new context arn:aws:eks:eu-north-1:545009829015:cluster/fullstack-demo-eks-cluster to /home/ubuntu/.kube/config
ubuntu@ip-172-31-39-189:~$ kubectl get nodes
NAME                                          STATUS   ROLES    AGE   VERSION
ip-10-0-3-182.eu-north-1.compute.internal     Ready    <none>   18h   v1.28.15-eks-473151a
ip-10-0-4-184.eu-north-1.compute.internal     Ready    <none>   18h   v1.28.15-eks-473151a
ubuntu@ip-172-31-39-189:~$ kubectl get pods
NAME                                  READY   STATUS      RESTARTS   AGE
backend-94d799575-5xz25               1/1     Running     0          16h
backend-94d799575-sm6tj               1/1     Running     0          16h
frontend-c7c658694-mljw4              1/1     Running     0          18h
frontend-c7c658694-tcq4w              1/1     Running     0          18h
nginx-loadbalancer-99f646b9-7kgm2     1/1     Running     0          16h
nginx-loadbalancer-99f646b9-kz9v8     1/1     Running     0          16h
test-backend                          0/1     Completed   0          16h
```

```
ubuntu@ip-172-31-39-189:~$ kubectl get svc
NAME                         TYPE           CLUSTER-IP      EXTERNAL-IP                                                              POR
T(S)        AGE
backend-service              ClusterIP      172.20.84.137   <none>                                                                   80/
TCP         18h
frontend-service             ClusterIP      172.20.28.202   <none>                                                                   80/
TCP         18h
kubernetes                   ClusterIP      172.20.0.1      <none>                                                                   443
/TCP        18h
nginx-loadbalancer-service   LoadBalancer   172.20.51.39    a5302f9191ef348abaef159b01cea699-390532562.eu-north-1.elb.amazonaws.com   80:
30121/TCP   18h
ubuntu@ip-172-31-39-189:~$
```

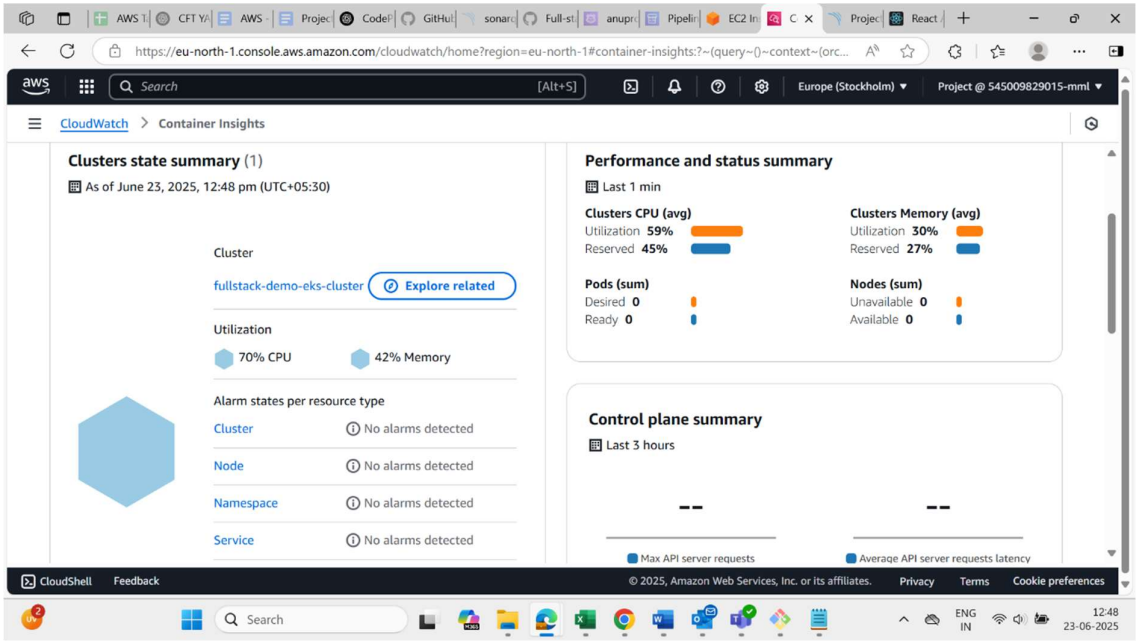
✅ Load Balancer Test – Verified frontend application access via ALB DNS.



Static Code Analysis – SonarQube Integration



CloudWatch Monitoring



🌍 Region 2: us-east-1 – Deployed via Terraform

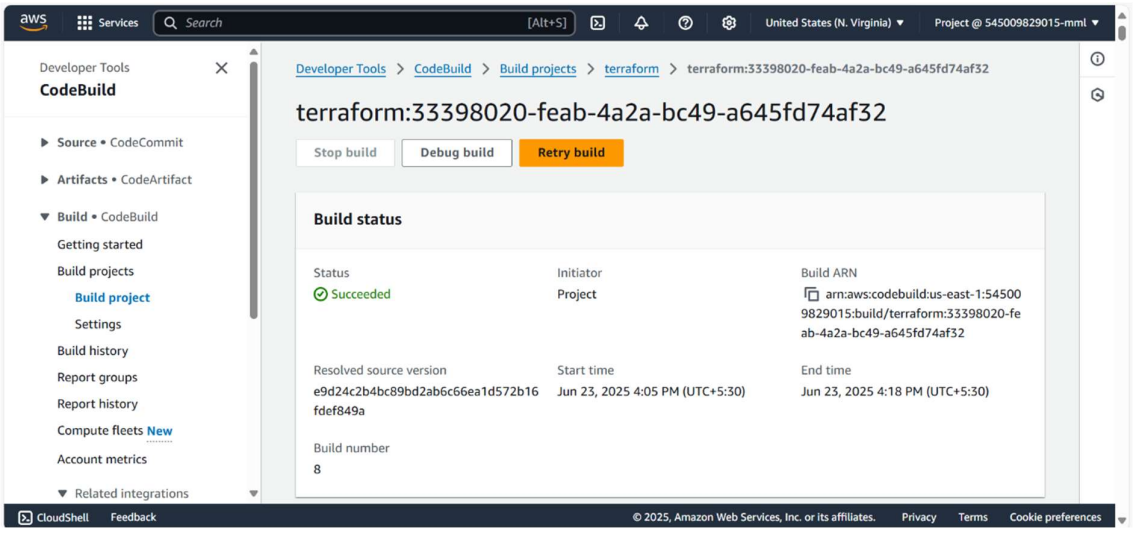
1️⃣ Infrastructure Provisioning using Terraform

GitHub Repo for terraform Infrastructure

https://github.com/AnushreeGM/AWS_CapstoneProject/tree/main/Terraform-main

Creation of infrastructure using terraform

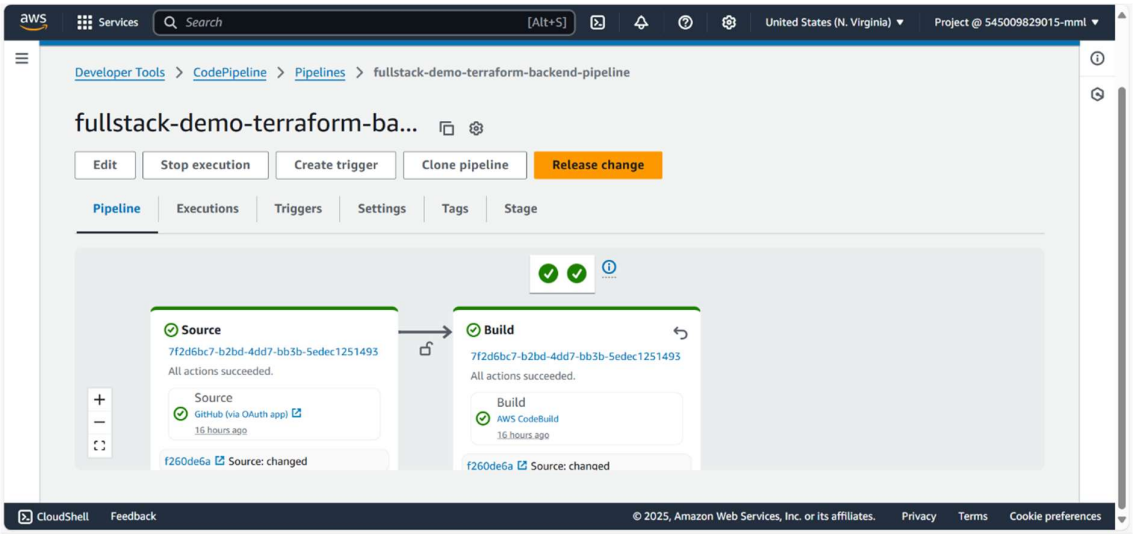
Resources Created:
- VPC with public/private subnets
- EKS Cluster
- RDS MySQL DB
- IAM roles and policies
- ALB and Security Groups



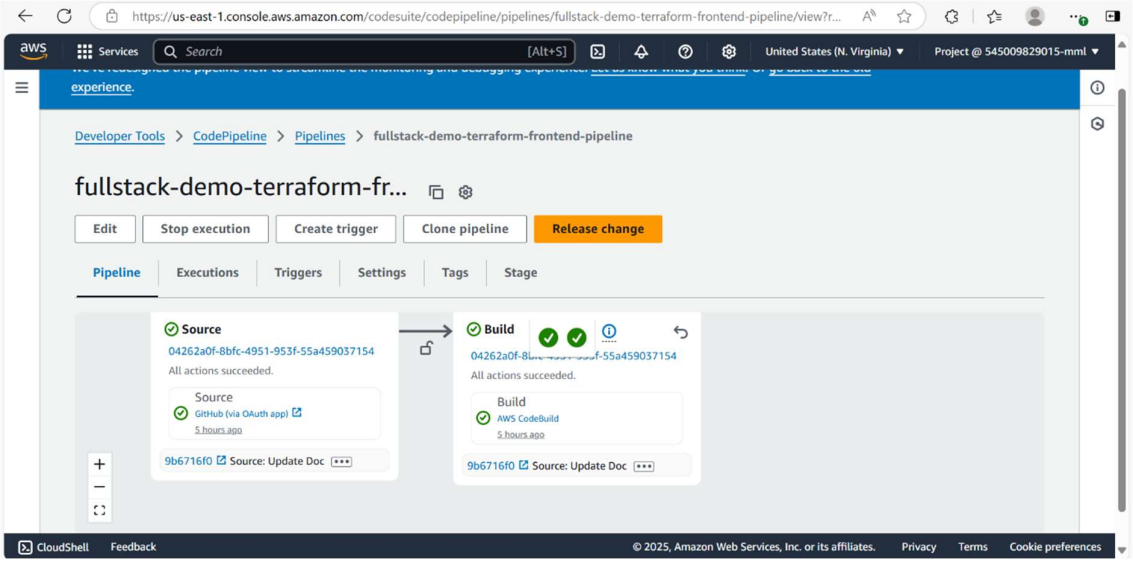
2 Application Deployment

Application deployed through CodePipeline using similar steps as in eu-north-1.

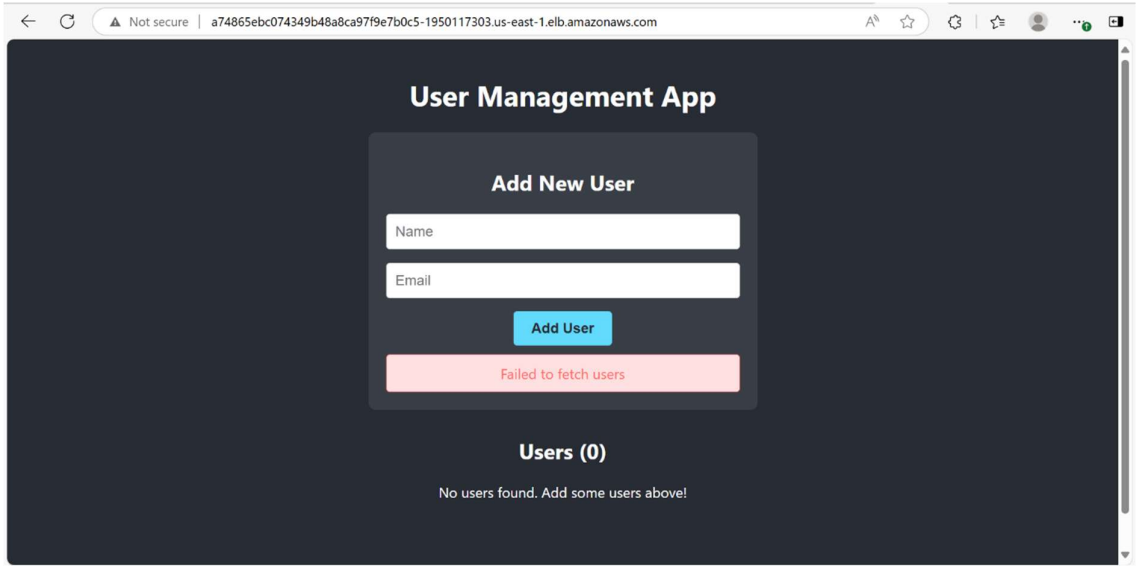
Backend pipeline



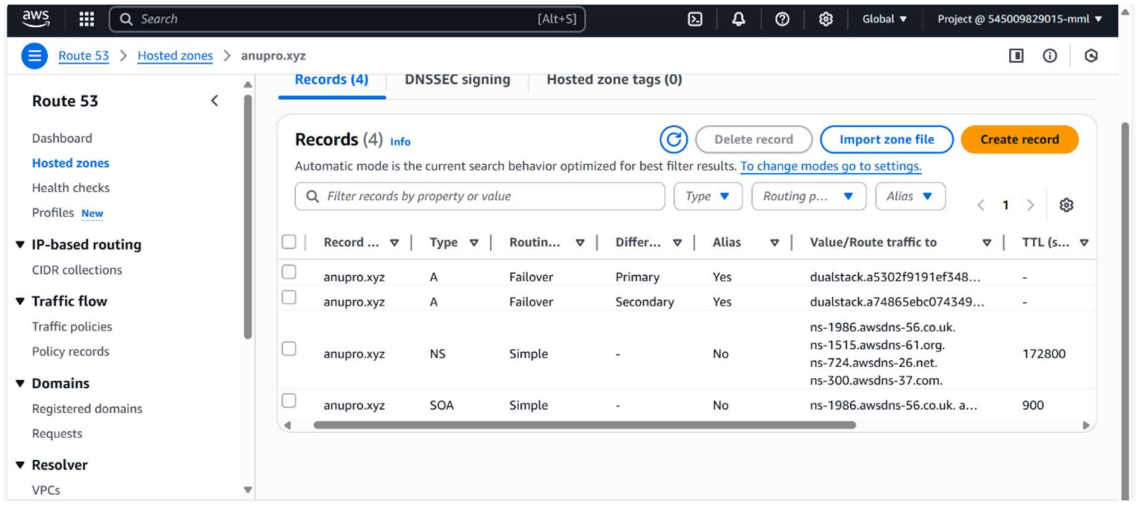
Frontend pipeline



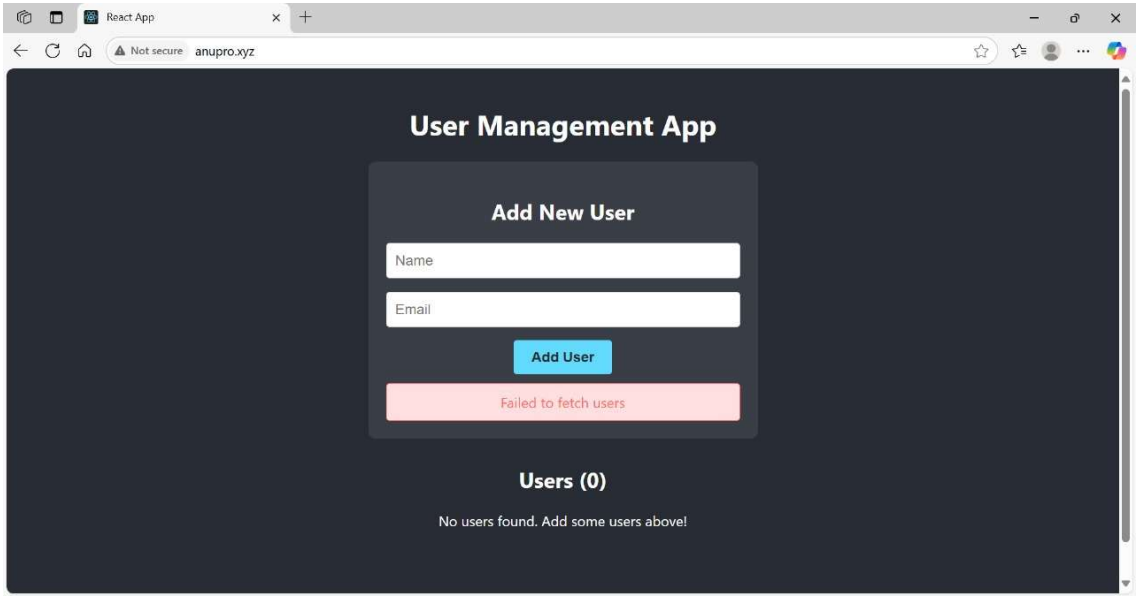
✅ Load Balancer Test – Verified frontend application access via ALB DNS.



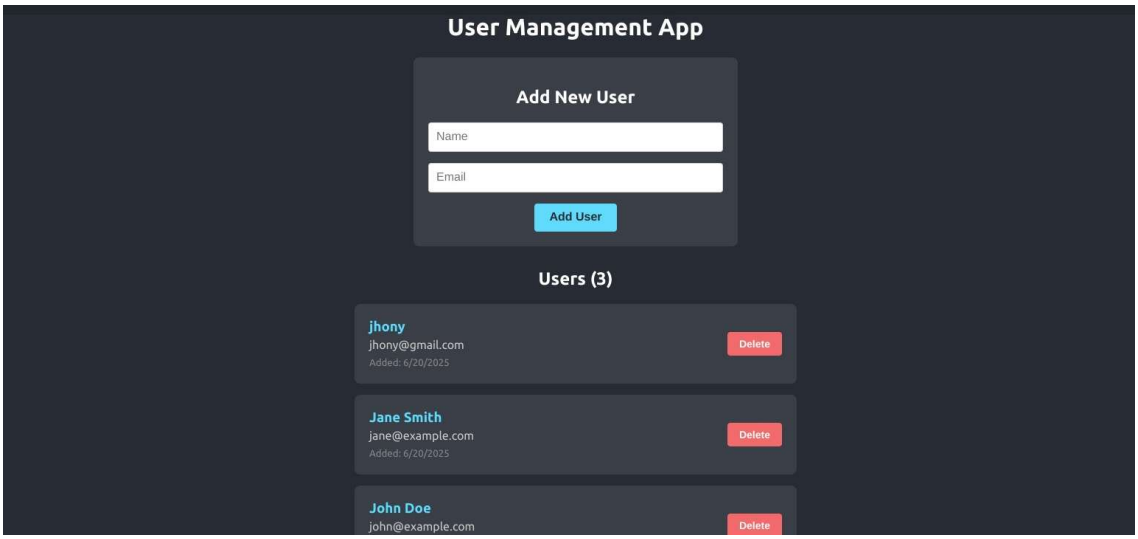
✅ Route 53 & Failover Routing Setup – Route 53 hosted zone created with:
- Primary: eu-north-1 ALB
- Secondary: us-east-1 ALB



✅ Domain Verification – Domain mapped successfully and tested failover.



After adding users



Summary and Key Outcomes

- CI/CD automation using CodePipeline and CodeBuild
- High availability via multi-region deployment
- Code quality via SonarQube
- Monitoring with CloudWatch
- Secure, scalable infrastructure using best practices

Appendix

- Architecture Diagram
- Pipeline Screenshots
- CloudWatch Dashboard
- Route 53 DNS Configuration