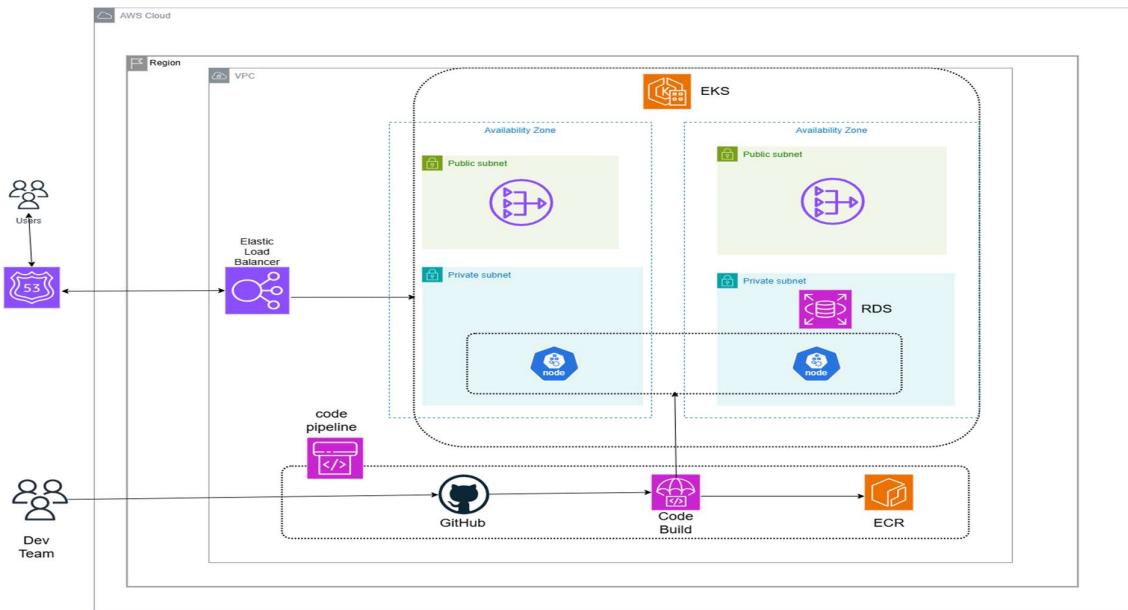


AWS Capstone Project – Deployment and Automation using CloudFormation and Terraform

🔗 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

1 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.

The screenshot shows the AWS CodeCommit service in the AWS Management Console. On the left, the navigation pane is open with the 'Source' section selected, specifically 'CodeCommit'. Under 'Repositories', there are two entries: 'fullstack-demo-backend-repo' and 'fullstack-demo-frontend-repo'. Both repositories are described as 'Backend Node.js API repository' and 'Frontend React application repository' respectively, both last modified '2 days ago'. The table includes columns for Name, Description, Last modified, Clone URL (HTTPS, SSH, HTTPS (GRC)), and AWS KMS Key. The AWS KMS Key for the backend repository is: arn:aws:kms:eu-north-1:545009829015:key/b3bcc31d-7ef5-4487-a717-381fb2a35372. The AWS KMS Key for the frontend repository is: arn:aws:kms:eu-north-1:545009829015:key/b3bcc31d-7ef5-4487-a717-381fb2a35372.

Create infra using cloud formation

The screenshot shows the AWS CloudFormation service in the AWS Management Console. The left sidebar shows 'Stacks' under 'CloudFormation'. A single stack named 'fullstack-demo-infrastructure' is listed with a status of 'CREATE_COMPLETE'. To the right, the 'Events' section displays a log of events for this stack, including the creation of nested stacks like 'EKSNodeGroup' and 'EKSCluster', and the creation of resources like 'RDSInstance'. All events show a status of 'CREATE_COMPLETE'.

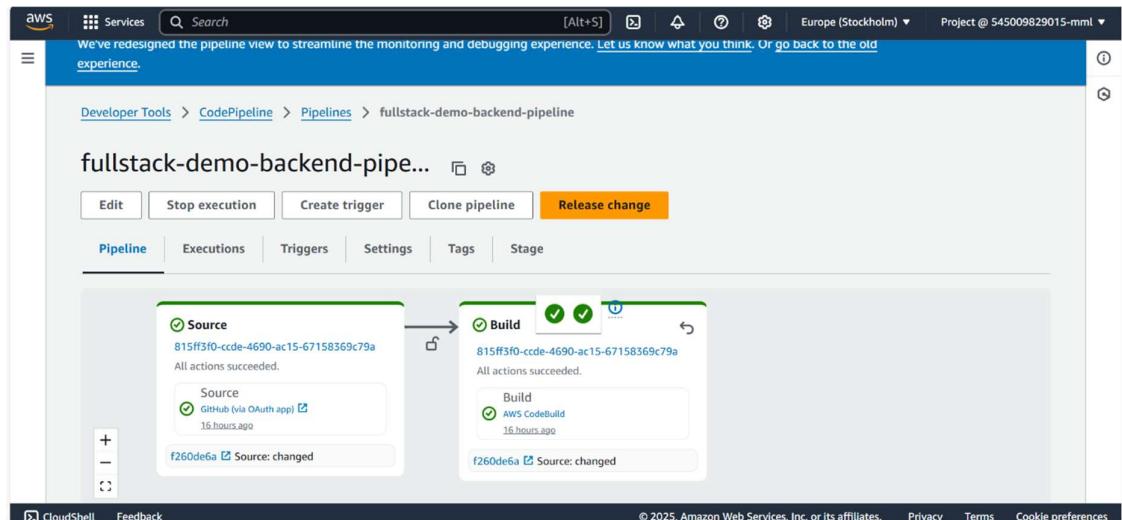
3 Application Deployment

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

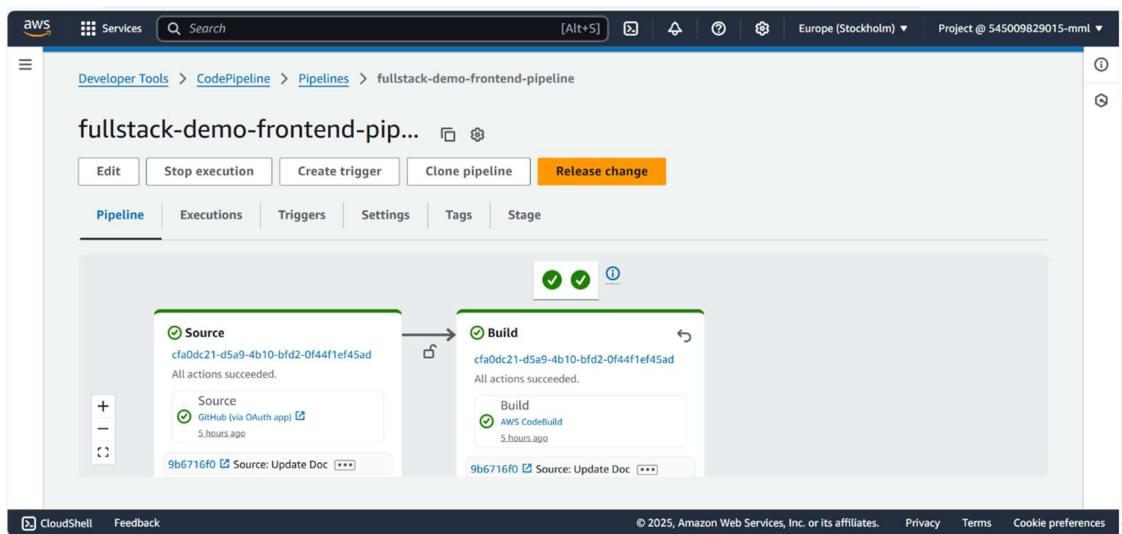
Deployment is triggered via AWS CodePipeline for both frontend and backend. Each pipeline includes CodeBuild, Docker image build, and EKS deployment stages.

Deploy the application through pipeline

Backend pipeline



Frontend pipeline

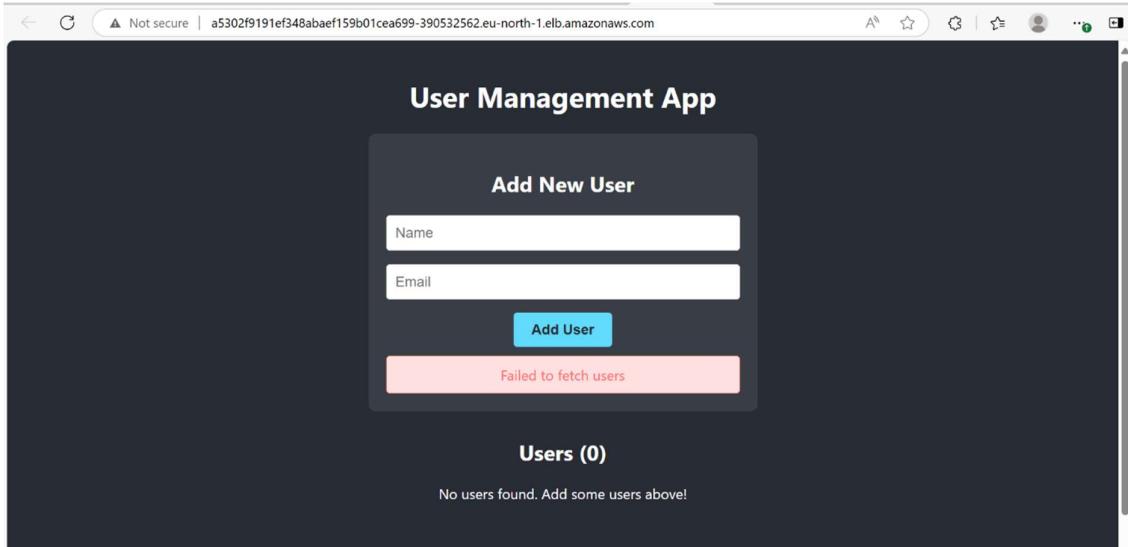


📝 Application Validation

- ✓ 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-1:1545009829015: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 PORT(S)  
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 a5302f9191ef348abaeef159b01cea699-390532562.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

The screenshot shows the SonarQube dashboard for the 'Full Stack Demo Frontend' project. The top navigation bar includes links for Projects, Issues, Rules, Quality Profiles, Quality Gates, and Administration. A search bar at the top right allows searching for projects. The main content area displays the 'Quality Gate Status' which is 'Passed'. Below this, under 'MEASURES', there are three sections: 'New Code' (since June 23, 2025, started 12 hours ago), 'Overall Code' (0 New Bugs, Reliability A), 'New Vulnerabilities' (0, Security A), and 'New Security Hotspots' (0, Security Review A). A note at the top right indicates 'Last analysis of this Branch had 1 warning' on June 24, 2025 at 10:55 AM.

CloudWatch Monitoring

The screenshot shows the AWS CloudWatch Container Insights dashboard for the 'fullstack-demo-eks-cluster'. The left sidebar lists 'Clusters state summary (1)' with one item: 'fullstack-demo-eks-cluster' (As of June 23, 2025, 12:48 pm (UTC+05:30)). Below this, 'Utilization' shows 70% CPU and 42% Memory. 'Alarm states per resource type' shows no alarms detected for Cluster, Node, Namespace, and Service. The right side features a 'Performance and status summary' section with metrics for Clusters CPU and Memory utilization (avg) and Pods (sum). It also includes a 'Control plane summary' section showing Max API server requests and Average API server requests latency over the last 3 hours. The bottom of the screen shows the AWS navigation bar and a taskbar with various application icons.

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

1 Infrastructure Provisioning using Terraform

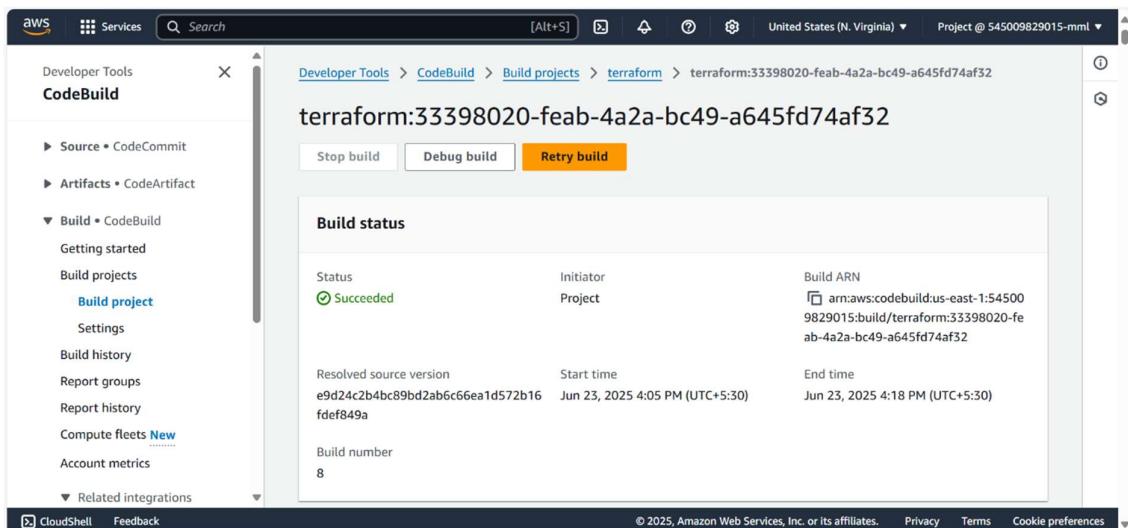
GitHub Repo for terraform Infrastructure

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

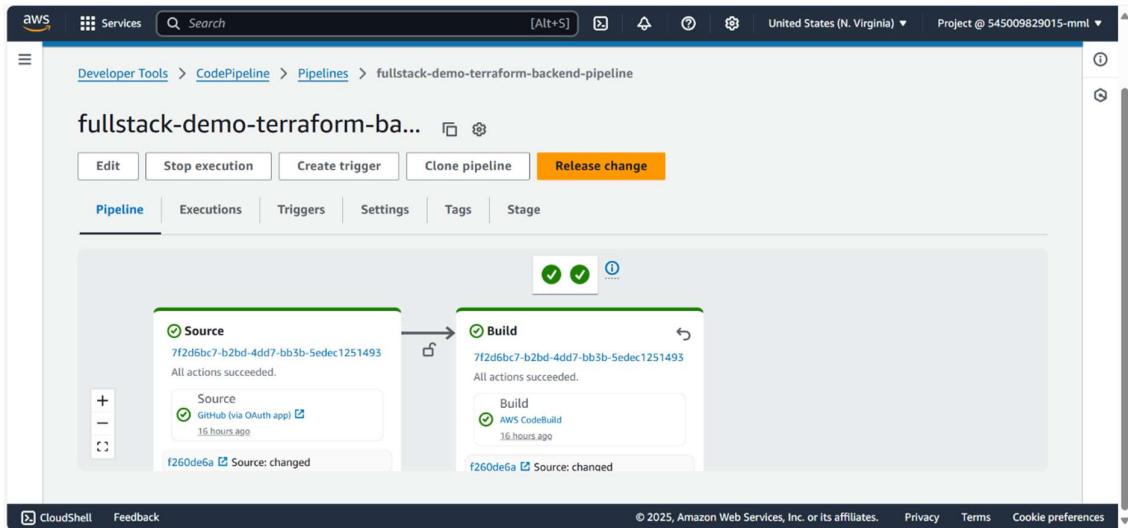


The screenshot shows the AWS CodeBuild console. The left sidebar is titled "CodeBuild" and includes options like "Source", "Artifacts", "Build", "Report groups", "Report history", "Compute fleets", and "Account metrics". Under "Build", "Build project" is selected. The main content area shows a build named "terraform:33398020-feab-4a2a-bc49-a645fd74af32". The "Build status" section indicates the build has "Succeeded". It provides details such as the resolved source version (e9d24c2b4bc89bd2ab6c66ea1d572b16 fdef849a), start time (Jun 23, 2025 4:05 PM (UTC+5:30)), end time (Jun 23, 2025 4:18 PM (UTC+5:30)), and build number (8). The "Build ARN" is listed as arn:aws:codebuild:us-east-1:545009829015:build/terraform:33398020-feab-4a2a-bc49-a645fd74af32.

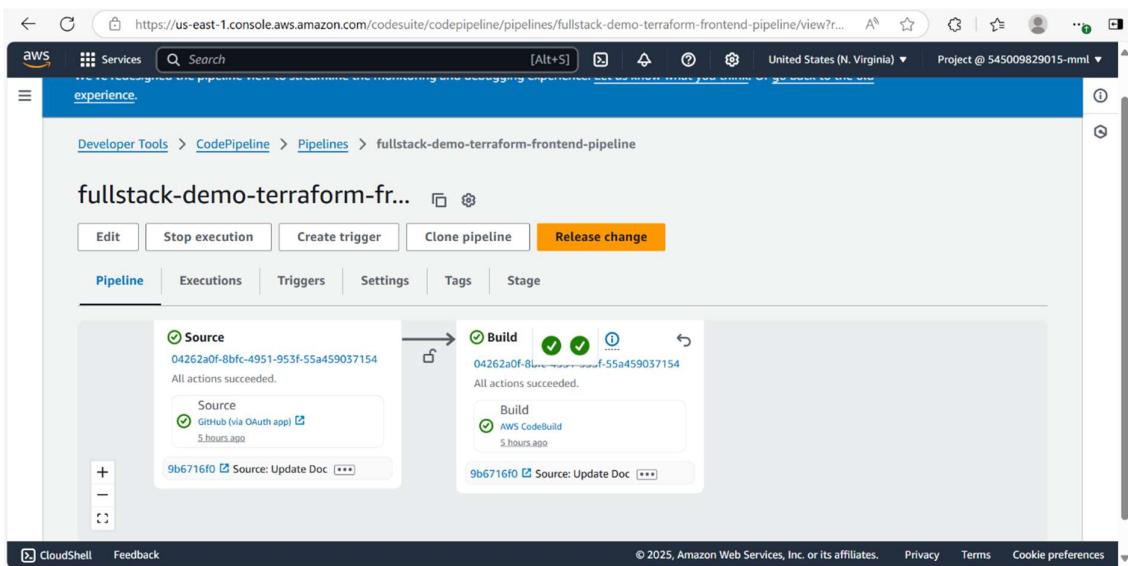
2 Application Deployment

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

Backend pipeline



Frontend pipeline

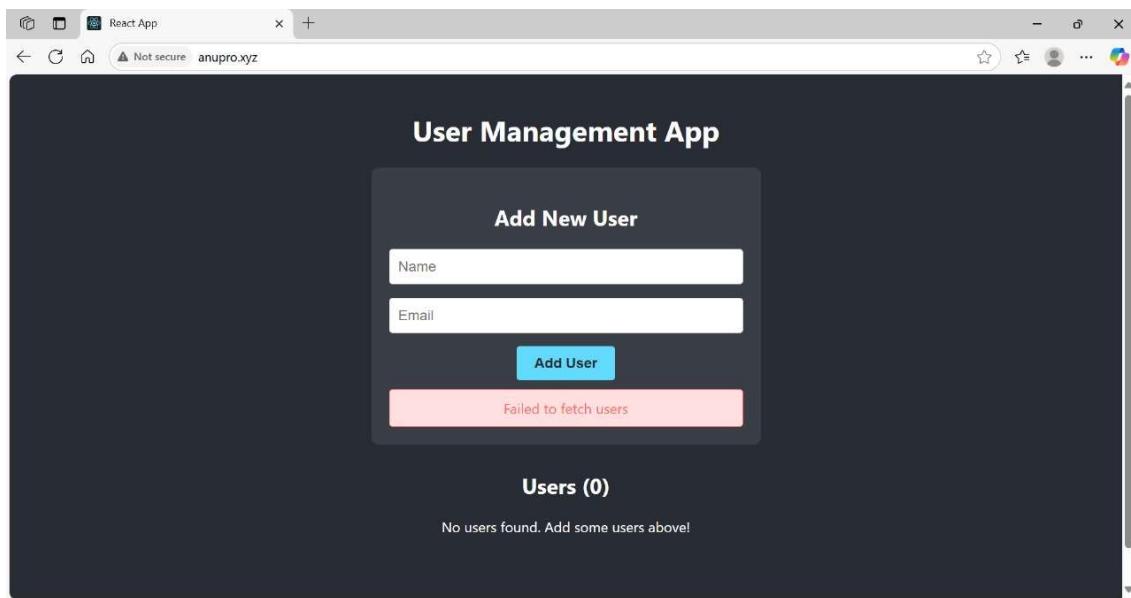


- Load Balancer Test – Verified frontend application access via ALB DNS.

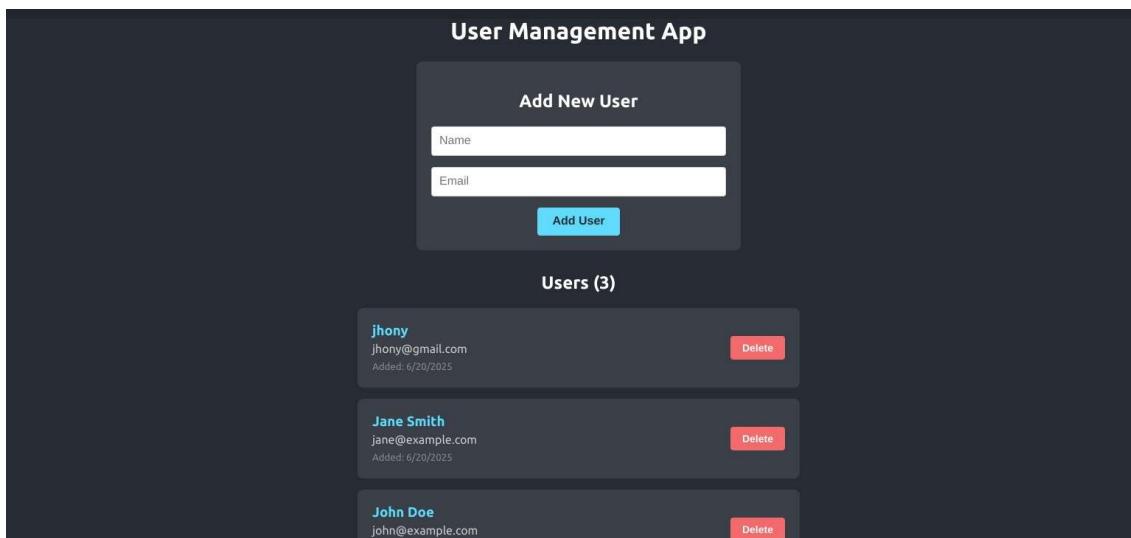
- Route 53 & Failover Routing Setup – Route 53 hosted zone created with:
 - Primary: eu-north-1 ALB
 - Secondary: us-east-1 ALB

	Record ...	Type	Routing...	Differ...	Alias	Value/Route traffic to	TTL (s...)
<input type="checkbox"/>	anupro.xyz	A	Failover	Primary	Yes	dualstack.a5302f9191ef348...	-
<input type="checkbox"/>	anupro.xyz	A	Failover	Secondary	Yes	dualstack.a74865ebc074349...	-
<input type="checkbox"/>	anupro.xyz	NS	Simple	-	No	ns-1986.awsdns-56.co.uk. ns-1515.awsdns-61.org. ns-724.awsdns-26.net. ns-300.awsdns-37.com.	172800
<input type="checkbox"/>	anupro.xyz	SOA	Simple	-	No	ns-1986.awsdns-56.co.uk. a...	900

- 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