Bootcamp Project 1

Three Tier WebApp DevOps Implementation on AWS

Jatharthan Tharmanandasivam

Project Overview

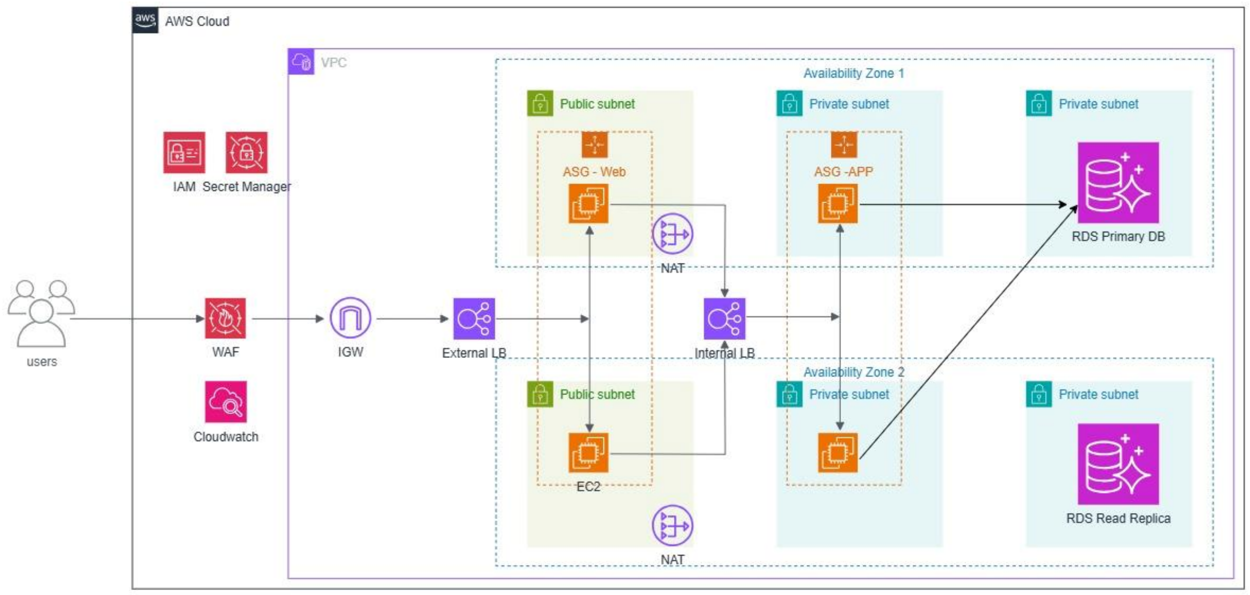
This project is for deploying a highly available three tier application on AWS cloud using terraform for Infrastructure as Code IaC. It includes automating the deployment leveraging CICD pipelines.

- Build and deploy a highly available three tier web application using terraform by automating the infrastructure provisioning using AWS pipeline and AWS CodeDeploy.
- Automating the deployment of Web and App Tier applications using CodePipline, CodeBuild and CodeDeploy

Technologies used

- Front End (Web Tier) Back End (App Tier) – Node.js
- Database – Amazon RDS Aurora (MySQL Compatible)
- Infrastructure as Code – Terraform
- CI/CD -AWS CodePipline, CodeBuild and CodeDeploy
- Logging – Cloud Watch
- Security – IAM ,AWS Secret Manager, WAF

Architecture of the deployed infrastructure



It involves web tier which will host the front end of the application where user will interact, and the app tier contains the execution logic, and the database tier includes the storage mechanism.

The infrastructure is load balanced and auto scalable. Auto scaling groups deploy the instances across two availability zones to ensure high availability and scalability of the architecture. The internal and internet facing load balancers direct the traffic across the instances in different availability zones thus ensuring the high availability of the application.

For security IAM roles with proper access were created. Secret Manager was used to save the database username and password. Web application Firewall (WAF) used to filter out traffic to the external load balancer.

Resources Deployed by terraform

1.Networking

VPC

6 Subnets across two availability zone. One Public and Two private subnets in each availability zones.

2 NAT gateways

Internet gateway

Route tables

Security Groups

2.Ec2 Related resources

Launch Templates for both app and web tier applications

Internal and External Loadbalancer

Target groups

Autoscaling groups

3.Security Resources

IAM roles

WAF

Secret Manager

4.Database

RDS Aurora MySQL database

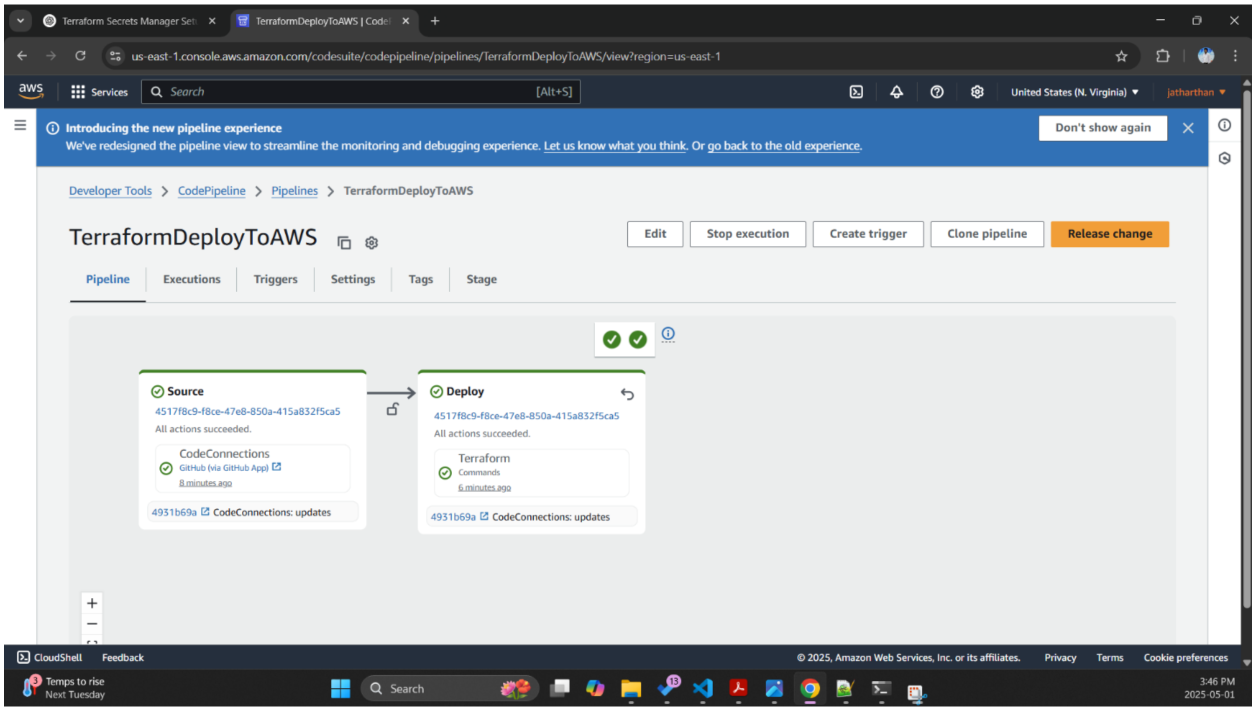
Terraform Code Used for the infrastructure provisioning

https://github.com/jatharthan/iac-terraform-aws-codepipeline-infra

Automating the infrastructure deployment

Infrastructure deployment was automated using GitHub, AWS CodePipline and CodeDeploy. When changes to the infrastructure are pushed to the GitHub it triggers the provisioning using the pipeline and the changes are deployed to the AWS cloud and track by the terraform state file stored in the S3 back end.

1.The below image shows the successful completion of the pipeline that deploys the infrastructure.



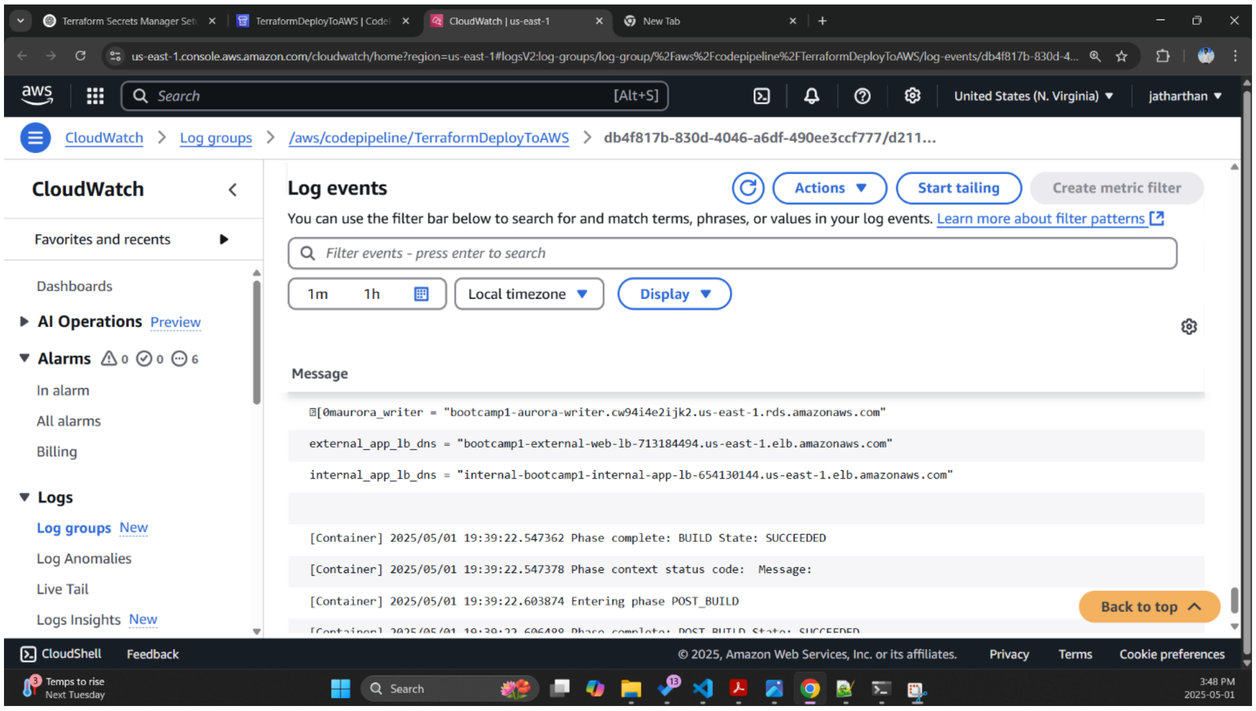
Introducing the new pipeline experience
We've redesigned the pipeline view to streamline the monitoring and debugging experience. Let us know what you think. Or go back to the old experience.

TerraformDeployToAWS

Source
4517f8c9-f8ce-47e8-850a-415a832f5ca5
All actions succeeded.
4931b69a CodeConnections: updates

Deploy
4517f8c9-f8ce-47e8-850a-415a832f5ca5
All actions succeeded.
4931b69a CodeConnections: updates

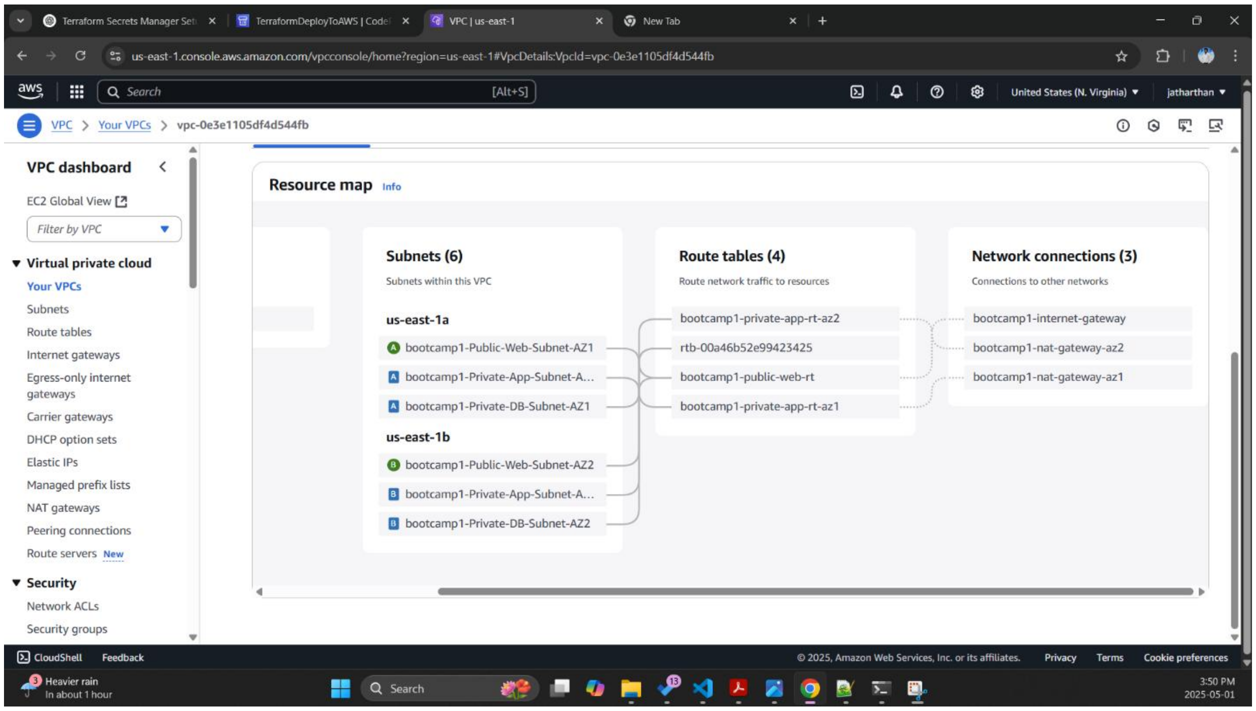
2.CloudWatch logs shows the successful completion of the deployment and you can see that the RDS writer endpoint and the internal load balancer DNS which will be required for the web and app tier application deployment.



```
[0maurora_writer = "bootcamp1-aurora-writer.cw94i4e2ijk2.us-east-1.rds.amazonaws.com"
external_app_lb_dns = "bootcamp1-external-web-lb-713184494.us-east-1.elb.amazonaws.com"
internal_app_lb_dns = "internal-bootcamp1-internal-app-lb-654130144.us-east-1.elb.amazonaws.com"

[Container] 2025/05/01 19:39:22.547362 Phase complete: BUILD State: SUCCEEDED
[Container] 2025/05/01 19:39:22.547378 Phase context status code:  Message:
[Container] 2025/05/01 19:39:22.603874 Entering phase POST_BUILD
```

03.Deployed VPC infrastructure



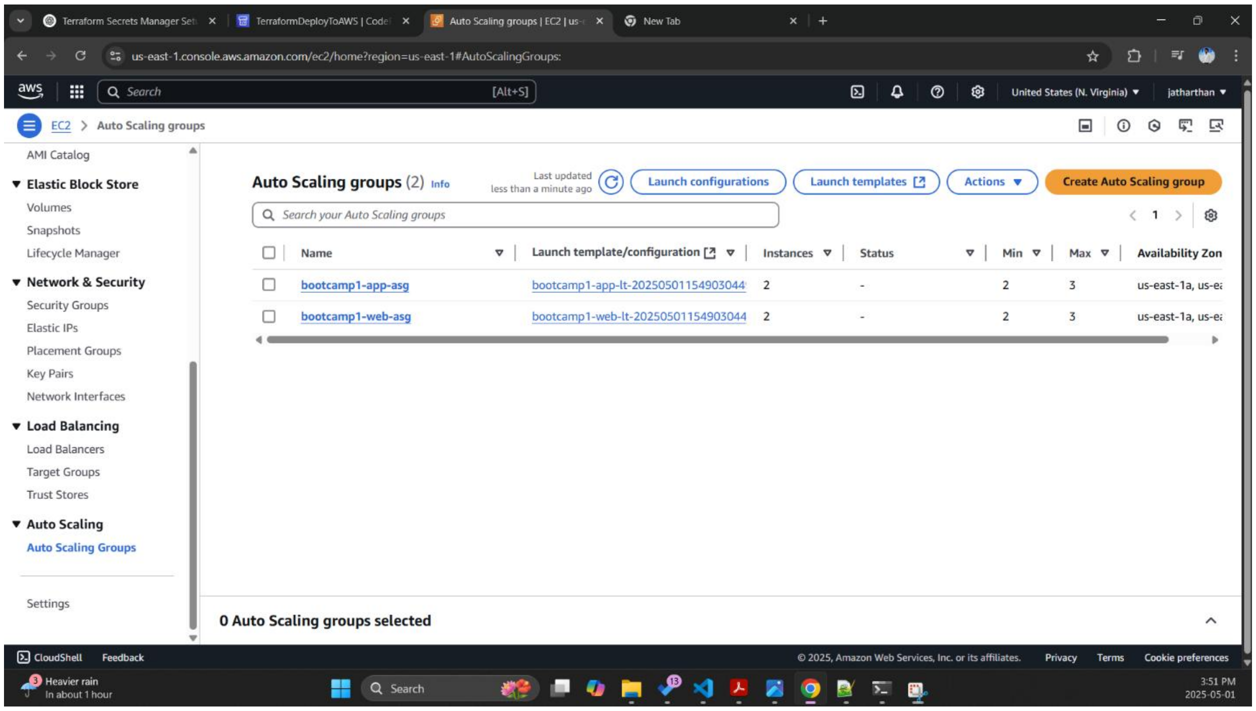
Subnets (6)
Subnets within this VPC
us-east-1a
- bootcamp1-Public-Web-Subnet-AZ1
- bootcamp1-Private-App-Subnet-A...
- bootcamp1-Private-DB-Subnet-AZ1

us-east-1b
- bootcamp1-Public-Web-Subnet-AZ2
- bootcamp1-Private-App-Subnet-A...
- bootcamp1-Private-DB-Subnet-AZ2

Route tables (4)
Route network traffic to resources
- bootcamp1-private-app-rt-az2
- rtb-00a46b52e99423425
- bootcamp1-public-web-rt
- bootcamp1-private-app-rt-az1

Network connections (3)
Connections to other networks
- bootcamp1-internet-gateway
- bootcamp1-nat-gateway-az2
- bootcamp1-nat-gateway-az1

04.Autoscaling groups



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#AutoScalingGroups:

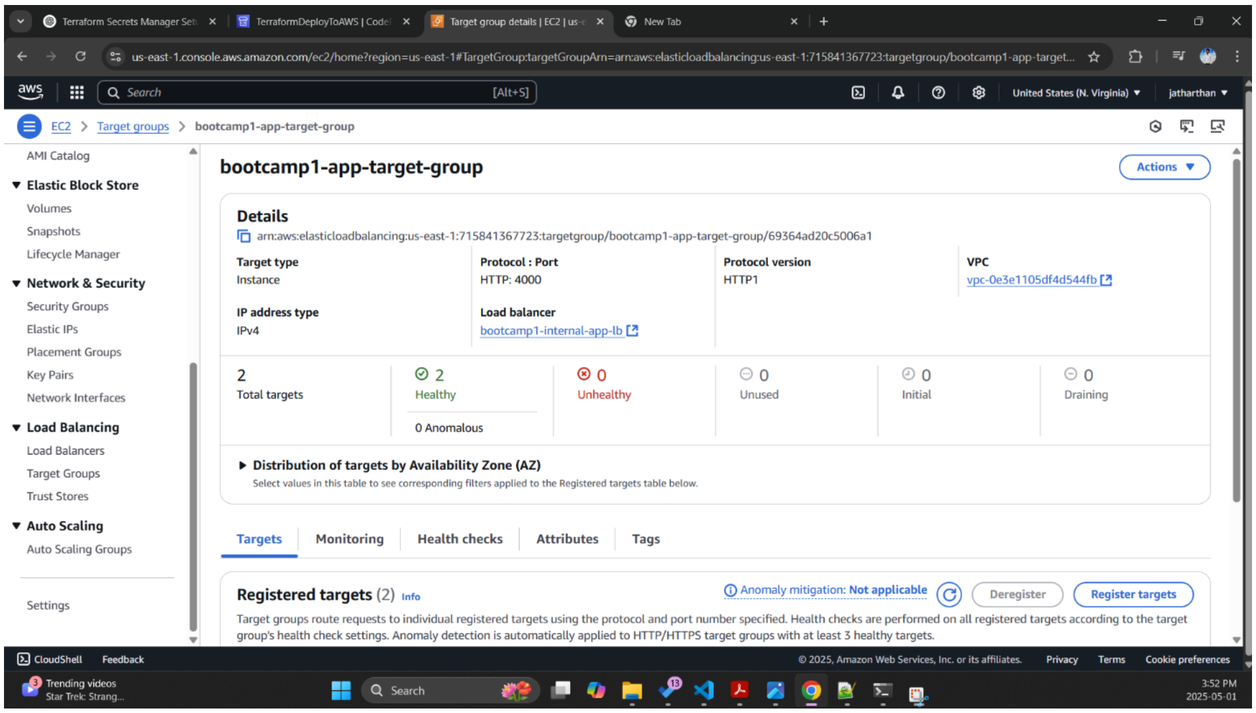
EC2 > Auto Scaling groups

Auto Scaling groups (2) Info — Last updated less than a minute ago

Name	Launch template/configuration	Instances	Status	Min	Max	Availability Zon
bootcamp1-app-asg	bootcamp1-app-lt-20250501154903044	2	-	2	3	us-east-1a, us-ea
bootcamp1-web-asg	bootcamp1-web-lt-20250501154903044	2	-	2	3	us-east-1a, us-ea

0 Auto Scaling groups selected

05.App-tier target groups with 2 healthy instances



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#TargetGroup:targetGroupArn=arn:aws:elasticloadbalancing:us-east-1:715841367723:targetgroup/bootcamp1-app-target...

EC2 > Target groups > bootcamp1-app-target-group

bootcamp1-app-target-group

Details

arn:aws:elasticloadbalancing:us-east-1:715841367723:targetgroup/bootcamp1-app-target-group/69364ad20c5006a1

Target type	Protocol : Port	Protocol version	VPC
Instance	HTTP: 4000	HTTP1	vpc-0e3e1105df4d544fb

IP address type	Load balancer
IPv4	bootcamp1-internal-app-lb

2	2	0	0	0	0
Total targets	Healthy	Unhealthy	Unused	Initial	Draining
	0 Anomalous				

Distribution of targets by Availability Zone (AZ)

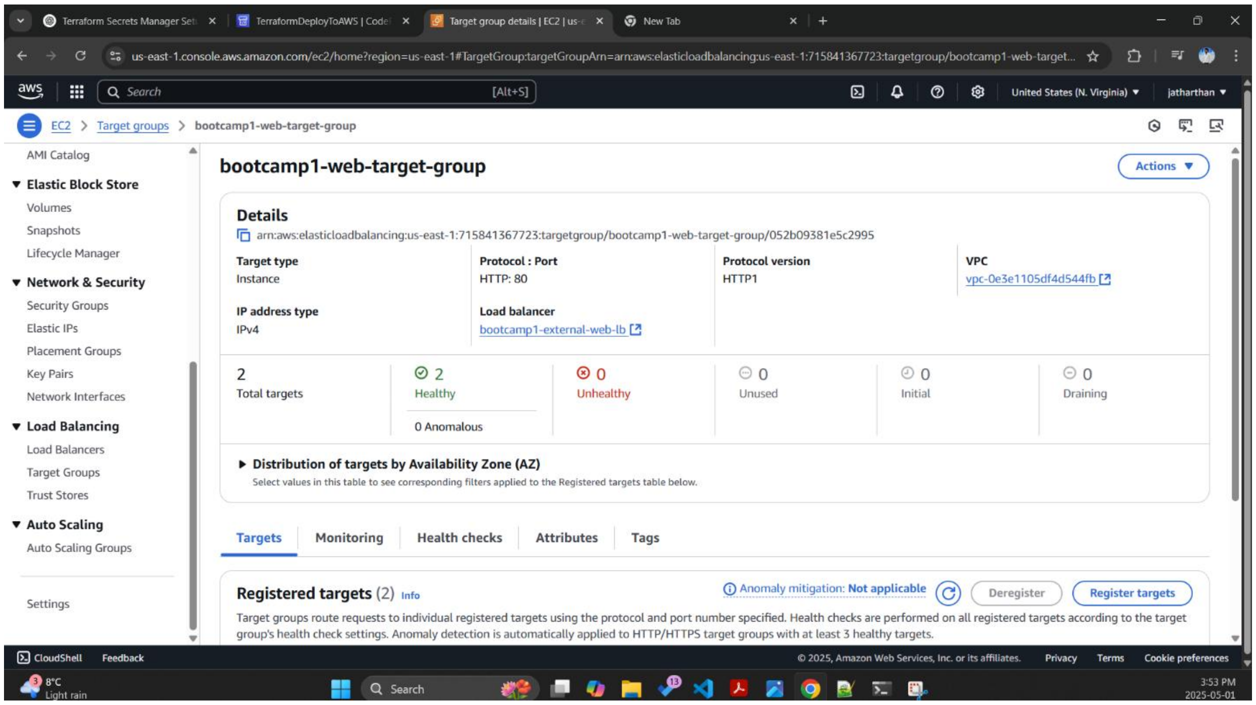
Select values in this table to see corresponding filters applied to the Registered targets table below.

Targets | Monitoring | Health checks | Attributes | Tags

Registered targets (2) Info — Anomaly mitigation: Not applicable

Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

06.web-tier target groups with 2 healthy instances


us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#TargetGroup:targetGroupArn=arn:aws:elasticloadbalancing:us-east-1:715841367723:targetgroup/bootcamp1-web-target...

EC2 > Target groups > bootcamp1-web-target-group

bootcamp1-web-target-group

Details

arn:aws:elasticloadbalancing:us-east-1:715841367723:targetgroup/bootcamp1-web-target-group/052b09381e5c2995

Target type Instance	**Protocol : Port** HTTP: 80	**Protocol version** HTTP1	**VPC** vpc-0e3e1105df4d544fb
IP address type IPv4	**Load balancer** bootcamp1-external-web-lb		

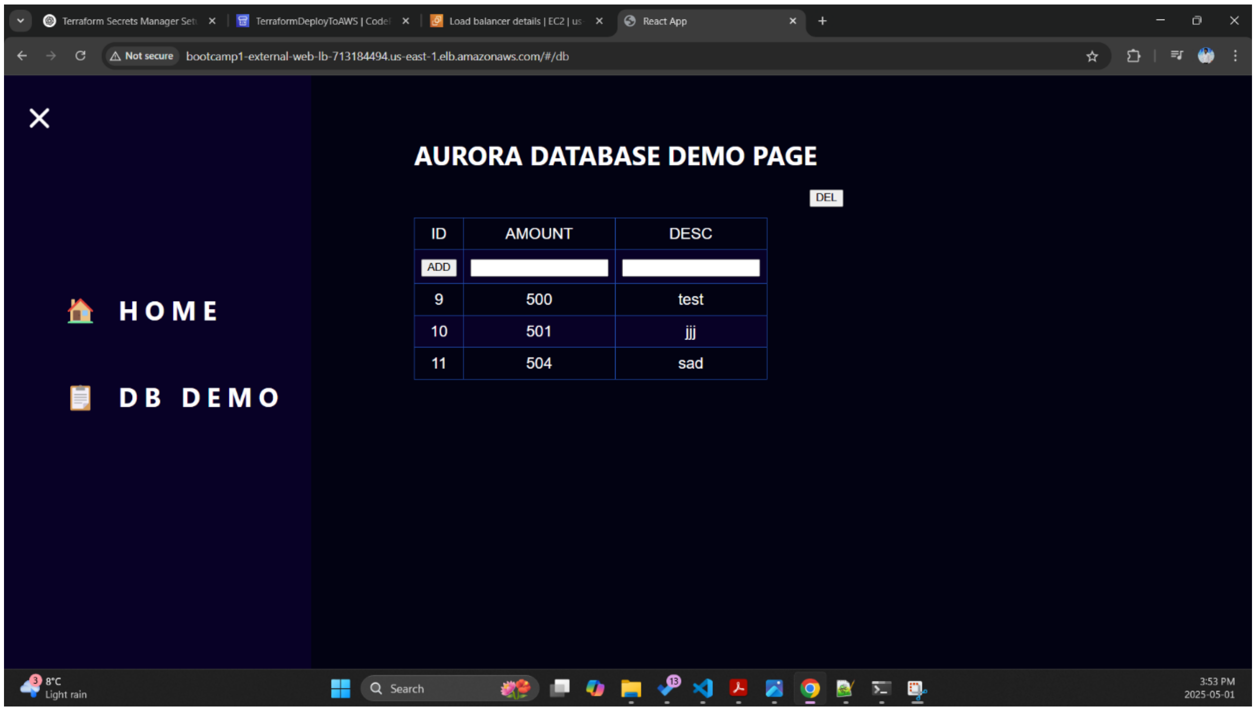
2 Total targets	✓ 2 Healthy	0 Unhealthy	0 Unused	0 Initial	0 Draining
	0 Anomalous				

Distribution of targets by Availability Zone (AZ)
Select values in this table to see corresponding filters applied to the Registered targets table below.

Targets | Monitoring | Health checks | Attributes | Tags

Registered targets (2) Info
Anomaly mitigation: **Not applicable**
Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

07.accessing the application via external load balancer

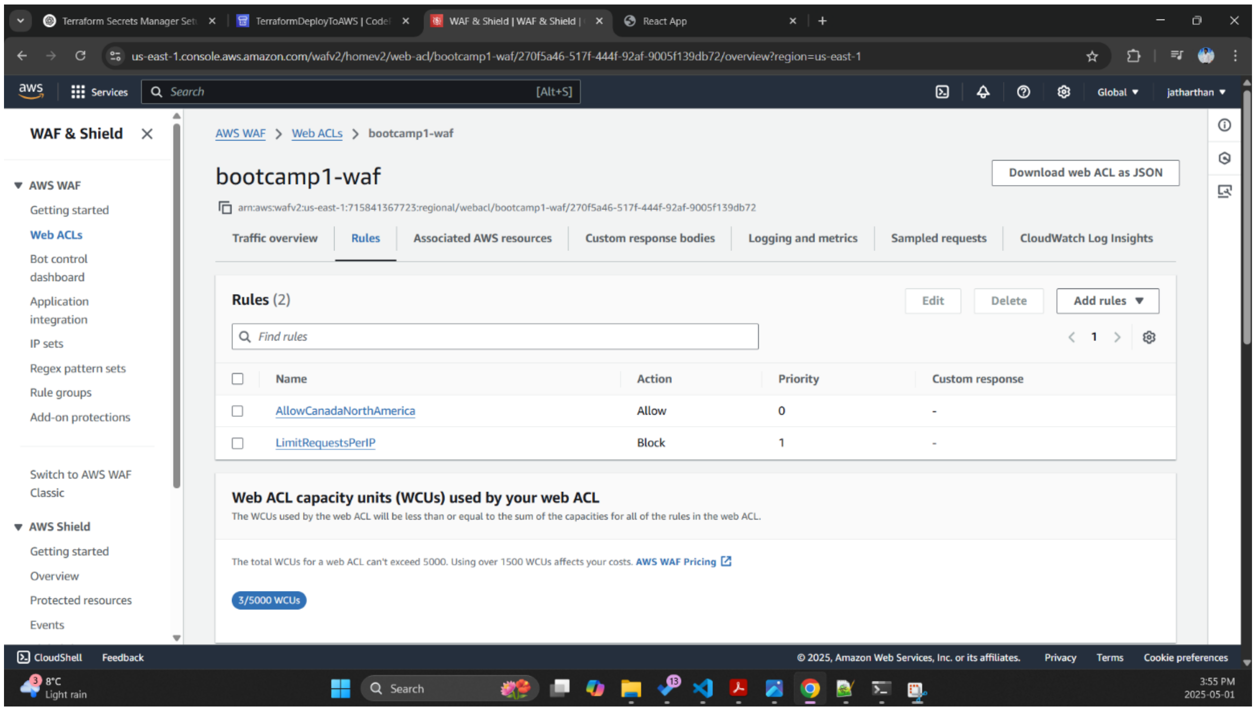

Not secure bootcamp1-external-web-lb-713184494.us-east-1.elb.amazonaws.com/#/db

AURORA DATABASE DEMO PAGE

ID	AMOUNT	DESC
ADD		
9	500	test
10	501	jjj
11	504	sad

Terraform Secrets Manager Set
TerraformDeployToAWS | Code
Load balancer details | EC2 | us
React App
HOME
DB DEMO
DEL
8°C Light rain
3:53 PM 2025-05-01

08.WAF rules that restricts number of request per IP and only allow request originated from the Canadian and US regions



AWS WAF > Web ACLs > bootcamp1-waf

bootcamp1-waf

arn:aws:wafv2:us-east-1:715841367723:regional/webacl/bootcamp1-waf/270f5a46-517f-444f-92af-9005f139db72

Rules (2)

Name	Action	Priority	Custom response
AllowCanadaNorthAmerica	Allow	0	-
LimitRequestsPerIP	Block	1	-

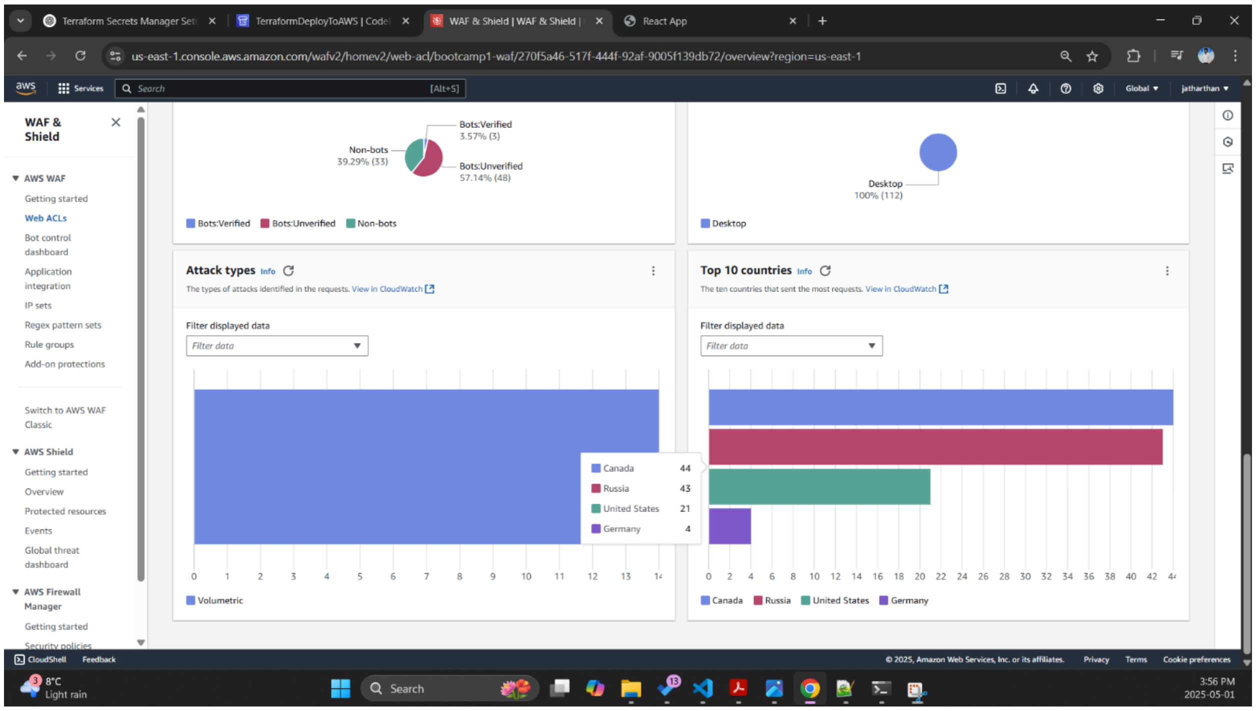
Web ACL capacity units (WCUs) used by your web ACL

The WCUs used by the web ACL will be less than or equal to the sum of the capacities for all of the rules in the web ACL.

The total WCUs for a web ACL can't exceed 5000. Using over 1500 WCUs affects your costs. AWS WAF Pricing

3/5000 WCUs

09.Some insights from WAF



Non-bots 39.29% (33)
Bots:Verified 3.57% (3)
Bots:Unverified 57.14% (48)

Desktop 100% (112)

Attack types

The types of attacks identified in the requests. View in CloudWatch

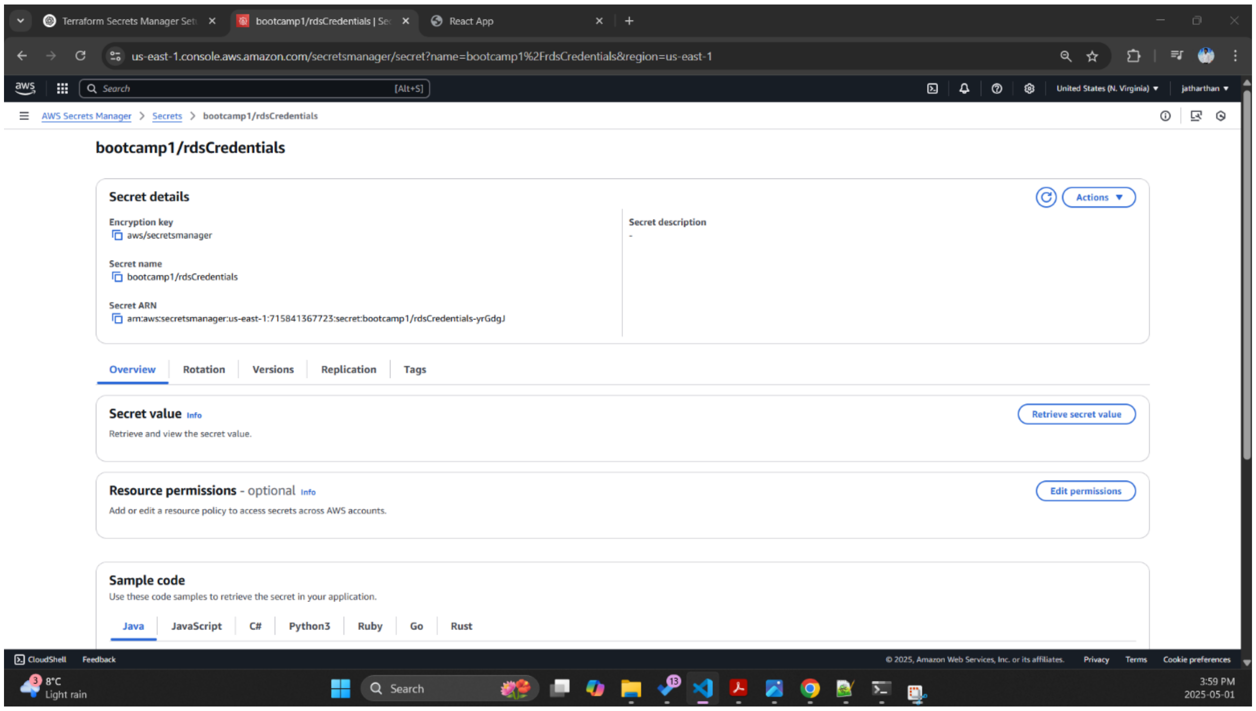
Volumetric

Top 10 countries

The ten countries that sent the most requests. View in CloudWatch

Canada	44
Russia	43
United States	21
Germany	4

10.secret manager



us-east-1.console.aws.amazon.com/secretsmanager/secret?name=bootcamp1%2FrdsCredentials&region=us-east-1

AWS Secrets Manager > Secrets > bootcamp1/rdsCredentials

bootcamp1/rdsCredentials

Secret details

Encryption key
aws/secretsmanager

Secret name
bootcamp1/rdsCredentials

Secret ARN
arn:aws:secretsmanager:us-east-1:715841367723:secret:bootcamp1/rdsCredentials-yrGdgJ

Secret description
-

Overview | Rotation | Versions | Replication | Tags

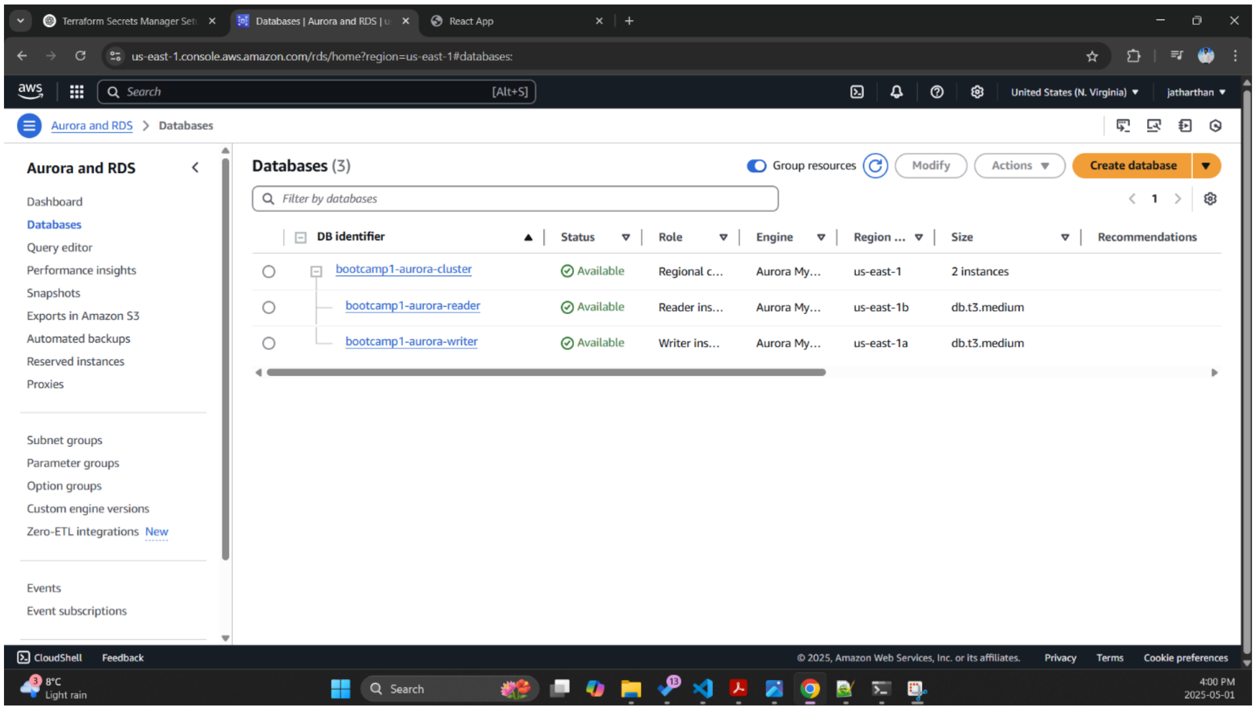
Secret value Info
Retrieve and view the secret value.

Resource permissions - optional Info
Add or edit a resource policy to access secrets across AWS accounts.

Sample code
Use these code samples to retrieve the secret in your application.

Java | JavaScript | C# | Python3 | Ruby | Go | Rust

11.Deployed Database



us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#databases:

Aurora and RDS > Databases

Databases (3)

DB identifier	Status	Role	Engine	Region ...	Size	Recommendations
bootcamp1-aurora-cluster	Available	Regional c...	Aurora My...	us-east-1	2 instances	
bootcamp1-aurora-reader	Available	Reader ins...	Aurora My...	us-east-1b	db.t3.medium	
bootcamp1-aurora-writer	Available	Writer ins...	Aurora My...	us-east-1a	db.t3.medium	

Setting up of CI/CD pipeline for the application deployment.

The two separate AWS code pipeline was setup to deploy Web tier application and App tier application.

When new changes are pushed to GitHub repo The application will be build by CodeBuild and deployed to respective Ec2 autoscaling groups by CodeDeploy. The deployment will happen in Blue Green deployment fashion. Code Deploy will create a new autoscaling group spin up new instances and deploy the application to that instance. When the health checks passed it will trigger traffic rerouting to the new autoscaling group by registering new instances to the respective target groups. Then after a wait time it will deregister the old instances and delete the old autoscaling group.

- GitHub Repo of web Tier that includes application code, buildspecs.yml, appspecs.yml

https://github.com/jatharthan/aws-codepipeline-web-tier-demo

https://github.com/jatharthan/aws-codepipeline-web-tier-demo/blob/master/buildspec.yml

https://github.com/jatharthan/aws-codepipeline-web-tier-demo/blob/master/appspec.yml

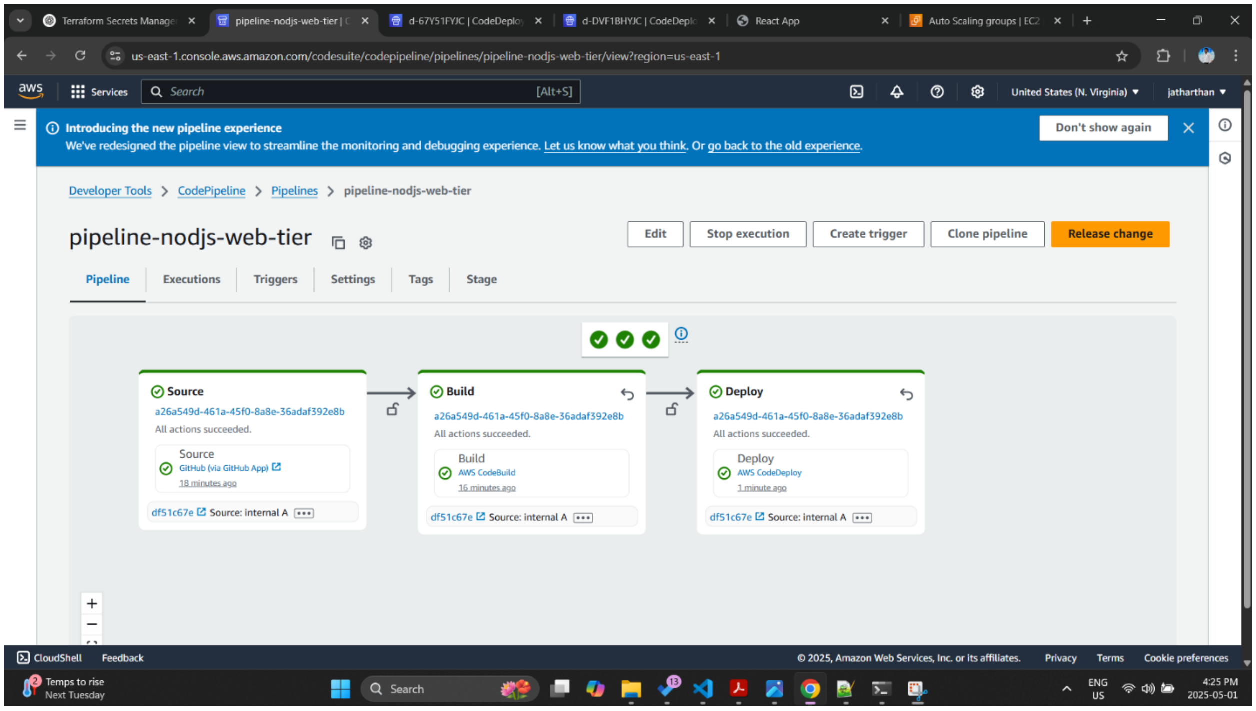
- GitHub Repo of app Tier that includes application code, buildspecs.yml, appspecs.yml

https://github.com/jatharthan/aws-codepipeline-app-tier-demo

https://github.com/jatharthan/aws-codepipeline-app-tier-demo/blob/master/buildspec.yml

https://github.com/jatharthan/aws-codepipeline-app-tier-demo/blob/master/appspec.yml

01.web tier deployment pipeline



us-east-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/pipeline-nodjs-web-tier/view?region=us-east-1

Introducing the new pipeline experience
We've redesigned the pipeline view to streamline the monitoring and debugging experience. Let us know what you think. Or go back to the old experience.

Developer Tools > CodePipeline > Pipelines > pipeline-nodjs-web-tier

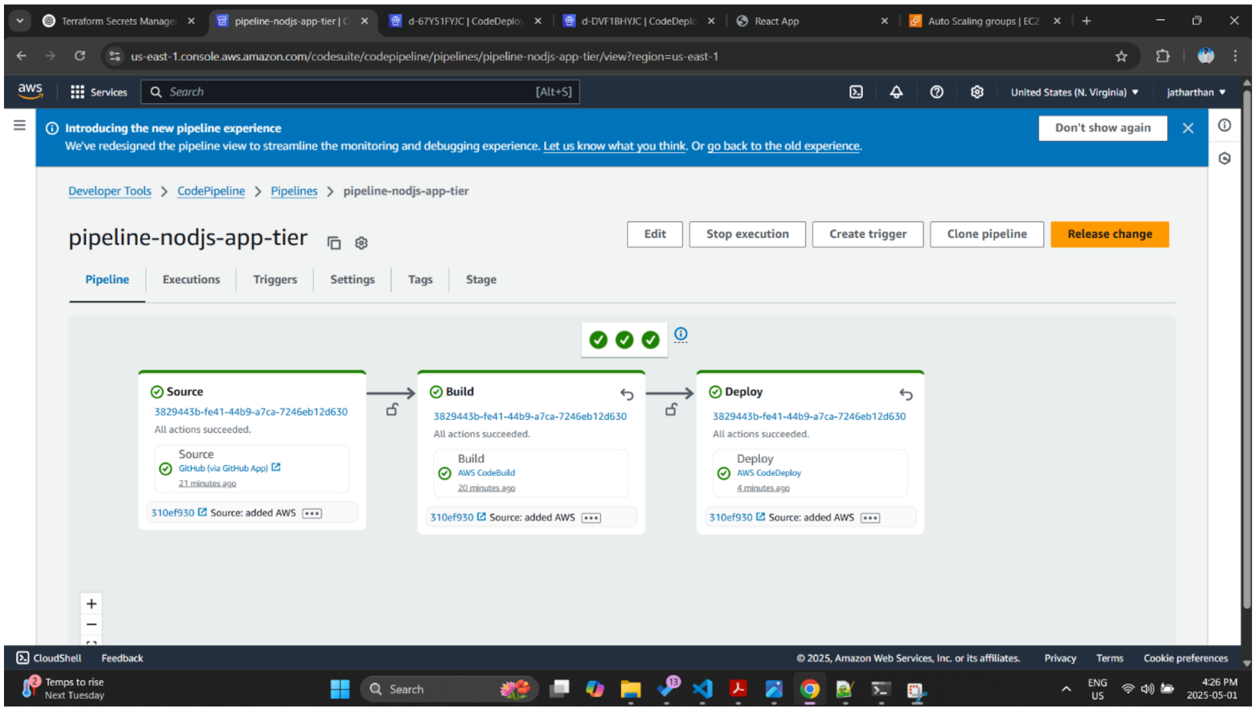
pipeline-nodjs-web-tier

Source
a26a549d-461a-45f0-8a8e-36adaf392e8b
All actions succeeded.
Source
GitHub (via GitHub App)
18 minutes ago
df51c67e Source: internal A

Build
a26a549d-461a-45f0-8a8e-36adaf392e8b
All actions succeeded.
Build
AWS CodeBuild
16 minutes ago
df51c67e Source: internal A

Deploy
a26a549d-461a-45f0-8a8e-36adaf392e8b
All actions succeeded.
Deploy
AWS CodeDeploy
1 minute ago
df51c67e Source: internal A

02.App tier deployment pipeline



us-east-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/pipeline-nodjs-app-tier/view?region=us-east-1

Introducing the new pipeline experience
We've redesigned the pipeline view to streamline the monitoring and debugging experience. Let us know what you think. Or go back to the old experience.

Developer Tools > CodePipeline > Pipelines > pipeline-nodjs-app-tier

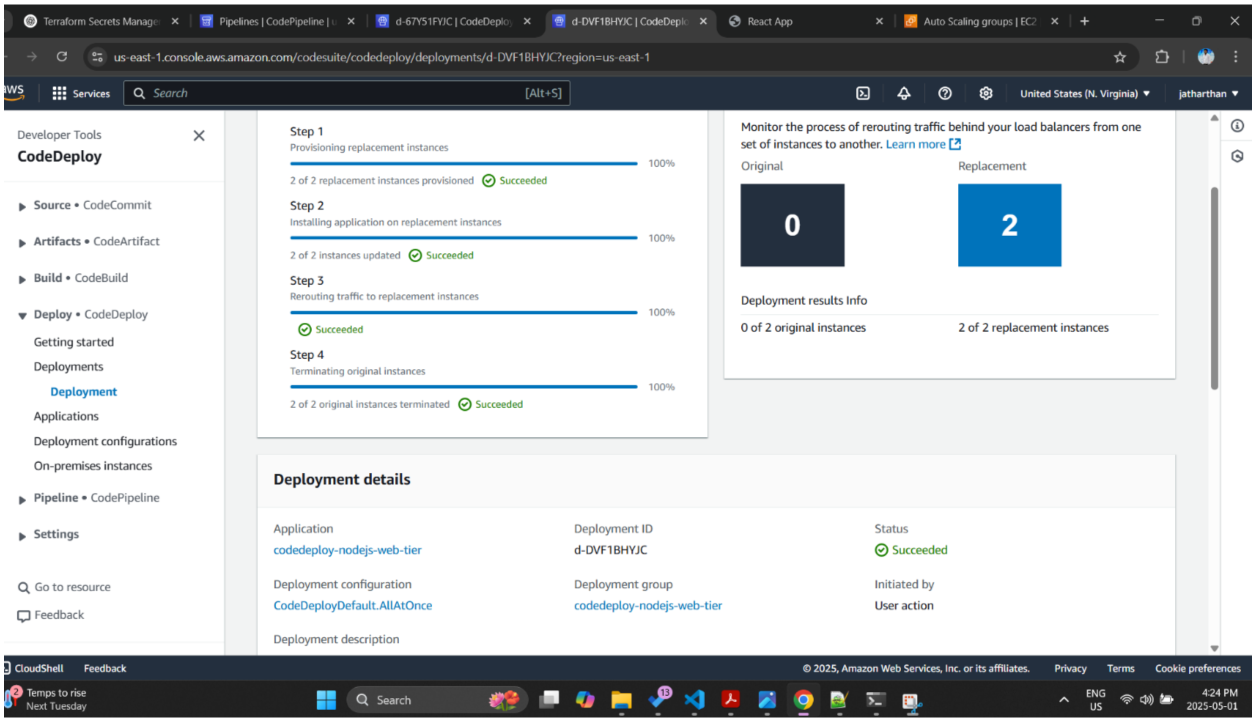
pipeline-nodjs-app-tier

Source
3829443b-fe41-44b9-a7ca-7246eb12d630
All actions succeeded.
Source
GitHub (via GitHub App)
21 minutes ago
310ef930 Source: added AWS

Build
3829443b-fe41-44b9-a7ca-7246eb12d630
All actions succeeded.
Build
AWS CodeBuild
20 minutes ago
310ef930 Source: added AWS

Deploy
3829443b-fe41-44b9-a7ca-7246eb12d630
All actions succeeded.
Deploy
AWS CodeDeploy
4 minutes ago
310ef930 Source: added AWS

03.deployment stages of web tier


us-east-1.console.aws.amazon.com/codesuite/codedeploy/deployments/d-DVF1BHYJC?region=us-east-1

Step 1
Provisioning replacement instances 100%
2 of 2 replacement instances provisioned ✓ Succeeded

Step 2
Installing application on replacement instances 100%
2 of 2 instances updated ✓ Succeeded

Step 3
Rerouting traffic to replacement instances 100%
✓ Succeeded

Step 4
Terminating original instances 100%
2 of 2 original instances terminated ✓ Succeeded

Monitor the process of rerouting traffic behind your load balancers from one set of instances to another. Learn more

Original	Replacement
0	2

Deployment results Info

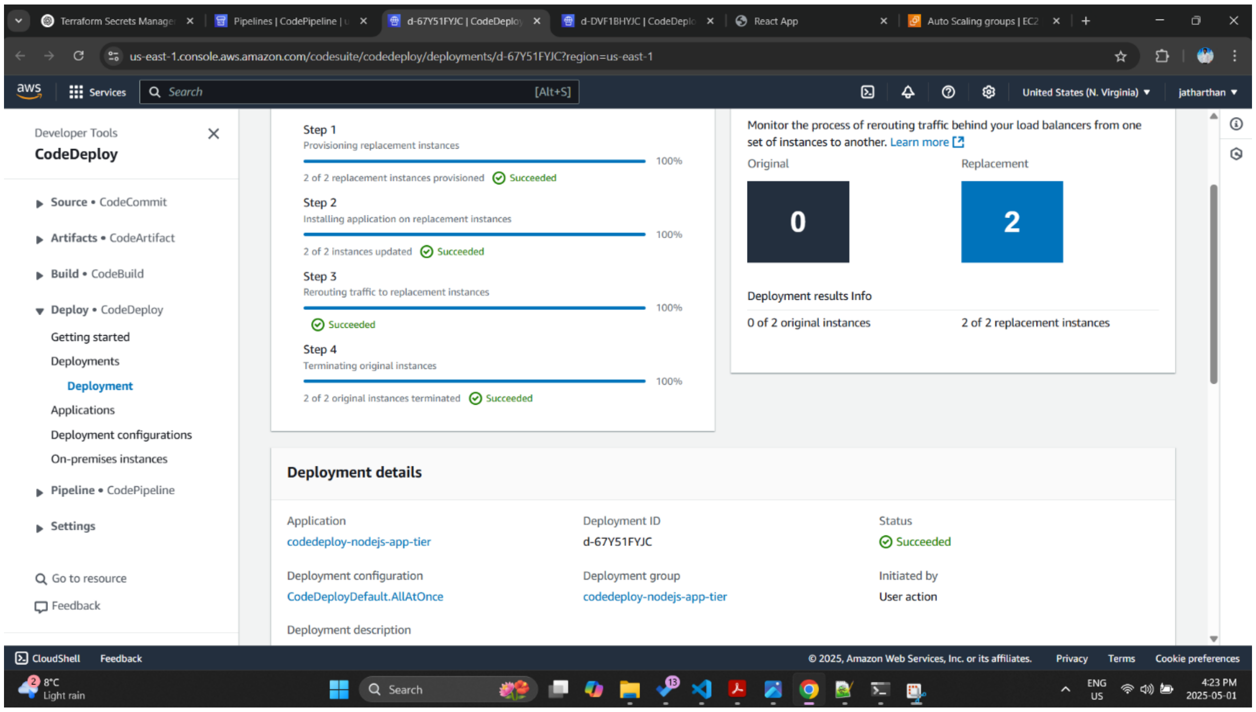
0 of 2 original instances	2 of 2 replacement instances

Deployment details

Application	Deployment ID	Status
codedeploy-nodejs-web-tier	d-DVF1BHYJC	✓ Succeeded
Deployment configuration	**Deployment group**	**Initiated by**
CodeDeployDefault.AllAtOnce	codedeploy-nodejs-web-tier	User action

Deployment description

04.deployment stages of app tier


us-east-1.console.aws.amazon.com/codesuite/codedeploy/deployments/d-67Y51FYJC?region=us-east-1

Step 1
Provisioning replacement instances 100%
2 of 2 replacement instances provisioned ✓ Succeeded

Step 2
Installing application on replacement instances 100%
2 of 2 instances updated ✓ Succeeded

Step 3
Rerouting traffic to replacement instances 100%
✓ Succeeded

Step 4
Terminating original instances 100%
2 of 2 original instances terminated ✓ Succeeded

Monitor the process of rerouting traffic behind your load balancers from one set of instances to another. Learn more

Original	Replacement
0	2

Deployment results Info

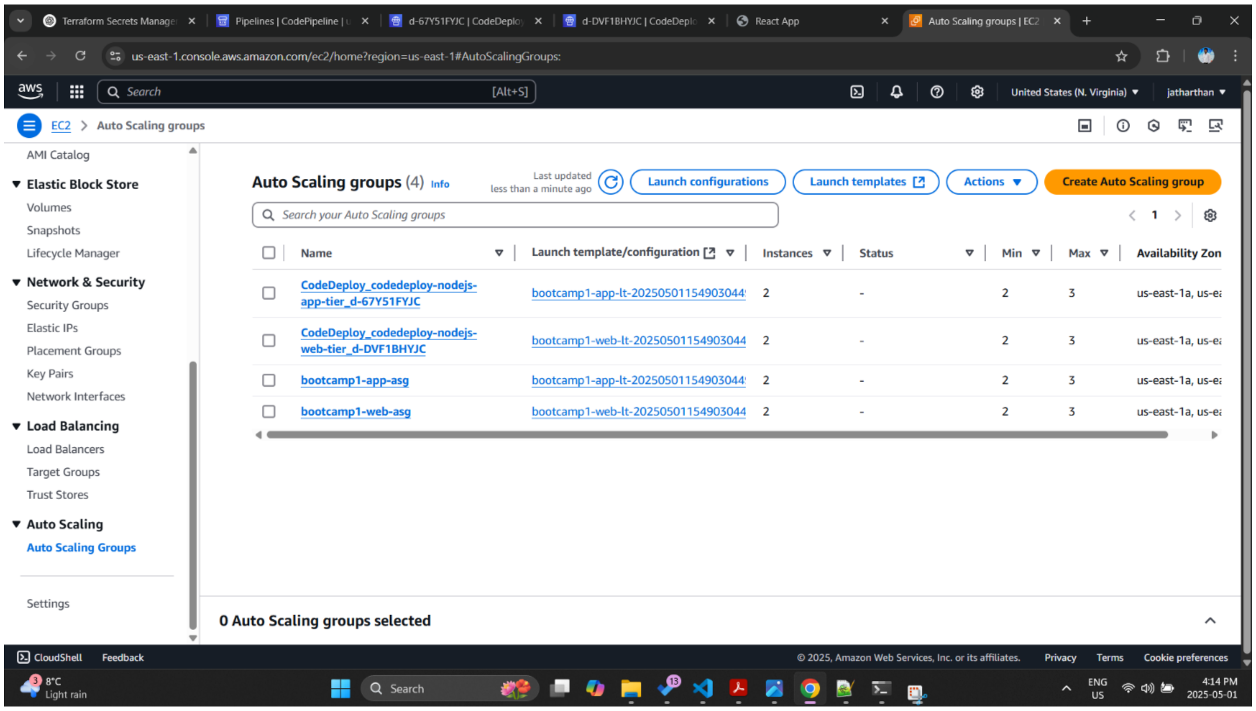
0 of 2 original instances	2 of 2 replacement instances

Deployment details

Application	Deployment ID	Status
codedeploy-nodejs-app-tier	d-67Y51FYJC	✓ Succeeded
Deployment configuration	**Deployment group**	**Initiated by**
CodeDeployDefault.AllAtOnce	codedeploy-nodejs-app-tier	User action

Deployment description

05. New ASG created for blue green deployment



EC2 > Auto Scaling groups

Auto Scaling groups (4) Info

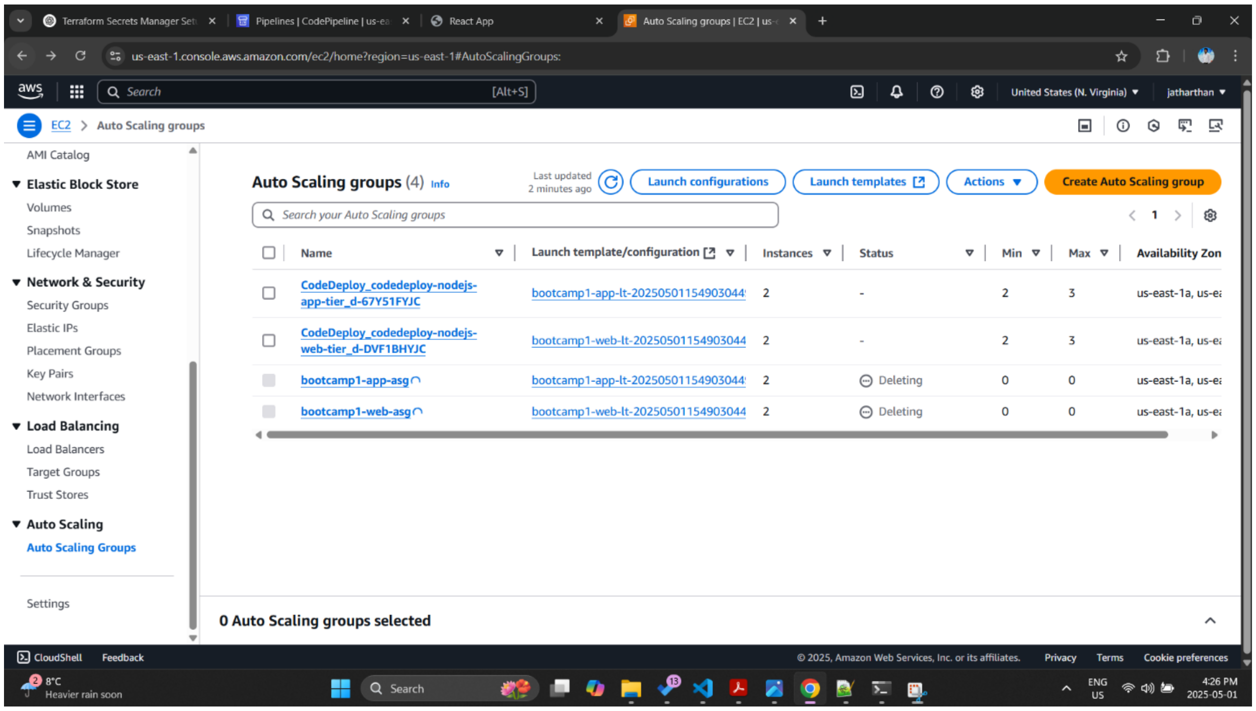
Last updated less than a minute ago

Launch configurations | Launch templates | Actions | Create Auto Scaling group

Name	Launch template/configuration	Instances	Status	Min	Max	Availability Zon
CodeDeploy_codedeploy-nodejs-app-tier_d-67Y51FYJC	bootcamp1-app-lt-20250501154903044	2	-	2	3	us-east-1a, us-ea
CodeDeploy_codedeploy-nodejs-web-tier_d-DVF1BHYJC	bootcamp1-web-lt-20250501154903044	2	-	2	3	us-east-1a, us-ea
bootcamp1-app-asg	bootcamp1-app-lt-20250501154903044	2	-	2	3	us-east-1a, us-ea
bootcamp1-web-asg	bootcamp1-web-lt-20250501154903044	2	-	2	3	us-east-1a, us-ea

0 Auto Scaling groups selected

06.previous ASG deleting



EC2 > Auto Scaling groups

Auto Scaling groups (4) Info

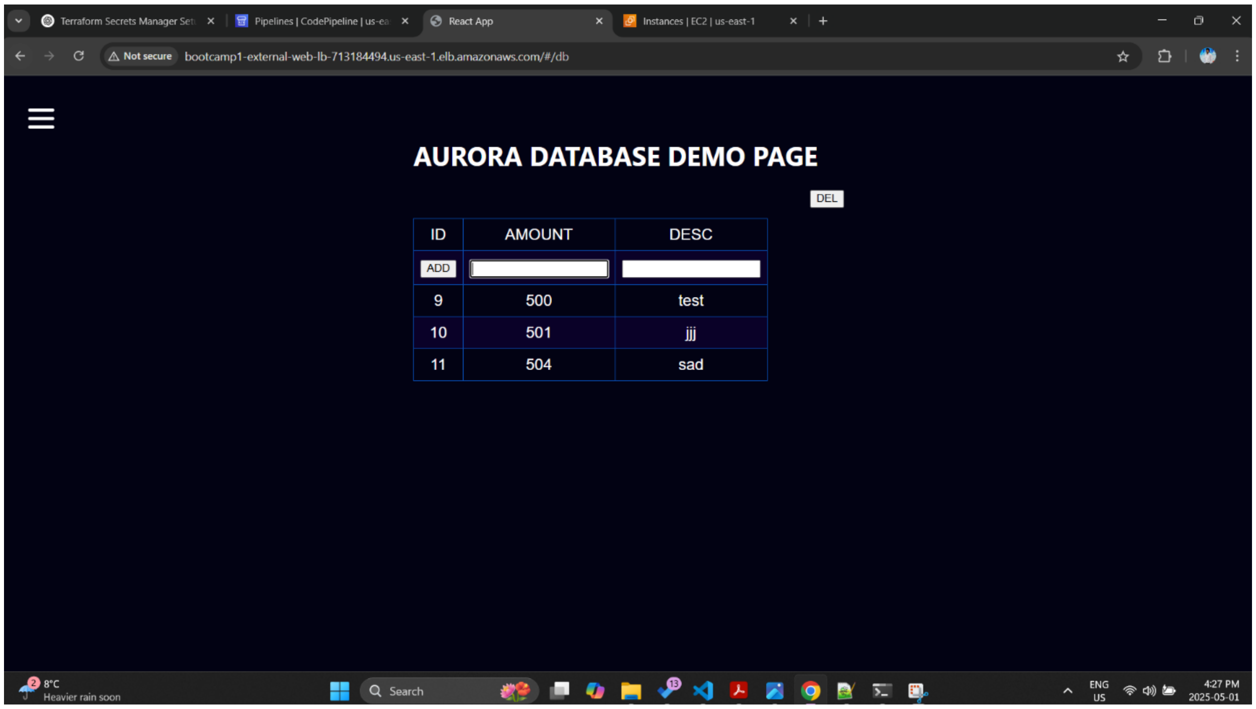
Last updated 2 minutes ago

Launch configurations | Launch templates | Actions | Create Auto Scaling group

Name	Launch template/configuration	Instances	Status	Min	Max	Availability Zon
CodeDeploy_codedeploy-nodejs-app-tier_d-67Y51FYJC	bootcamp1-app-lt-20250501154903044	2	-	2	3	us-east-1a, us-ea
CodeDeploy_codedeploy-nodejs-web-tier_d-DVF1BHYJC	bootcamp1-web-lt-20250501154903044	2	-	2	3	us-east-1a, us-ea
bootcamp1-app-asg	bootcamp1-app-lt-20250501154903044	2	Deleting	0	0	us-east-1a, us-ea
bootcamp1-web-asg	bootcamp1-web-lt-20250501154903044	2	Deleting	0	0	us-east-1a, us-ea

0 Auto Scaling groups selected

07.app is accessible after deployment


Terraform Secrets Manager Set
Pipelines | CodePipeline | us-ea
React App
Instances | EC2 | us-east-1
Not secure
bootcamp1-external-web-lb-713184494.us-east-1.elb.amazonaws.com/#/db
DEL
ADD
8°C
Heavier rain soon
Search
ENG
US
4:27 PM
2025-05-01

AURORA DATABASE DEMO PAGE

ID	AMOUNT	DESC
ADD		
9	500	test
10	501	jjj
11	504	sad

Summary

- ➢ A three-tier application was deployed using IaC using terraform and that deployment was automated using AWS codepipline, AWS codeDeploy.
- ➢ CI/CD was achieved by automating the application deployment to both tiers using CodePipline, CodeBuild and CodeDeploy. And deployment to autoscaling groups with blue green deployment.