www.linkedin.com/in/prafulpatel16

https://github.com/prafulpatel16

PRAFUL PATEL

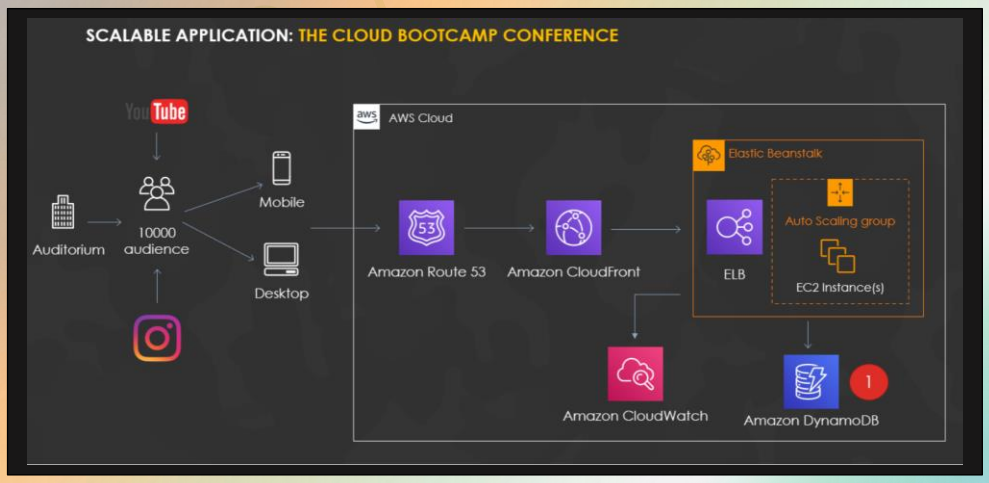
www.linkedin.com/in/prafulpatel16

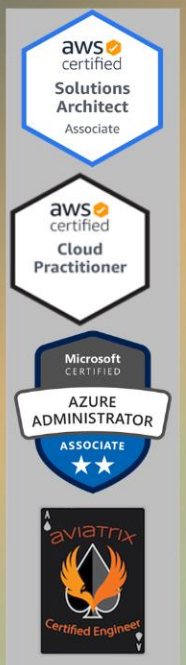
https://github.com/prafulpatel16

AWS BOOTCAMP PROJECT

PROJECT 3: DEPLOYMENT OF SCALABLE APPLICATION

IMPLEMENTED BY: PRAFUL PATEL





Date: Feb 15, 2022

Project Definition:

An IT services provider company called PRAfect Systems Inc. is engaged into providing software development solutions. Currently, they are going to launch a bootcamp where more than 10000 users may access their web application. So in that case due to heavy load on the web application the application has to auto scale without creating and managing any infrastructure components from scratch.

The plan is to deploy the web application itself without managing any infrastructure headache. The web servers should be able to handle load and database should also be scaled up.

Project Description:

Web Servers: Contains python application and packages.
Database Servers: Contains DynamoDB (NoSQL) database.

This project demonstrates an experience of web application deployment through Elastic Beanstalk and store the emails in DynamoDB which is NoSQL scalable database. Also in this project the web application is going to be cached the static and dynamic content using Cloudfront to an Edge locations close to the users.

Solution:

This project needs to be deployed on aws using AWS's platform as service called "Elastic Beanstalk" along with public server(EC2 machine) and one db server (dynamoDBdb instance). VPC services should consists more than public subnet and more than one private subnet so that web application machine should be places behind a public server, and behind private server the DB instance should be placed.

For Web server perspective, this server should be from a linux distribution systems. For networking and security perspective, web application server will have to be accessed and allowed publicly available so internet gateway needs to be provisioned and assigned to the public ip. For security concerns and avoiding cyber threats it has to configure some security group and assign and appropriate routes and rules based on requirement. Also web serve should be allowed to access from SSH at port 22 for system administration perspective, and it should allow to be accessible at custom TCP: Port 8080 from everywhere.

For DB server perspective, this needs to be behind a private subnet so it should not allow from external world so certain settings like publicly available should be disabled. Also it has to strictly define some rules from security groups so that no one can access and abuse the db server. DB credentials should be defined while provisioning a db server and pass through safe mechanism.

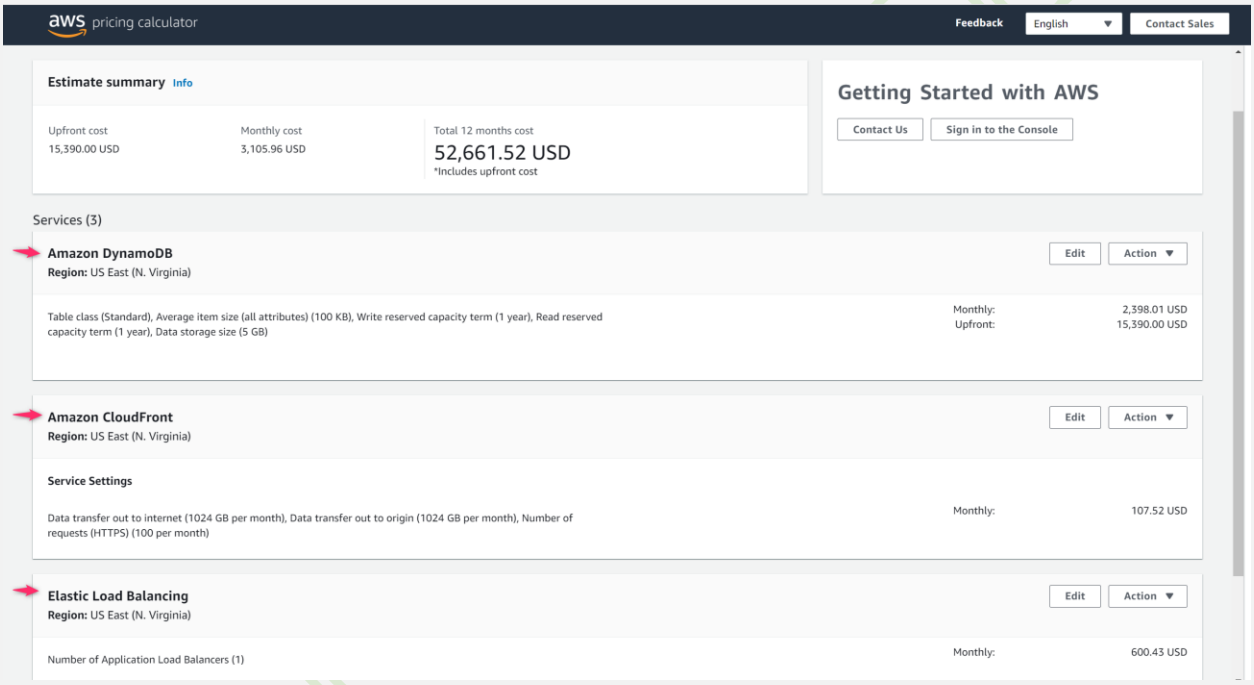
OS and application packages requirements are provided separately which needs to be provisioned and be ready before go on live session.

➢ Project Cost Estimation:

(Note: This cost is Not any actual cost, it's just an estimation based on high level requirement. Price may be vary based on adding and removing services based on requirement.)

Elastic Beanstalk

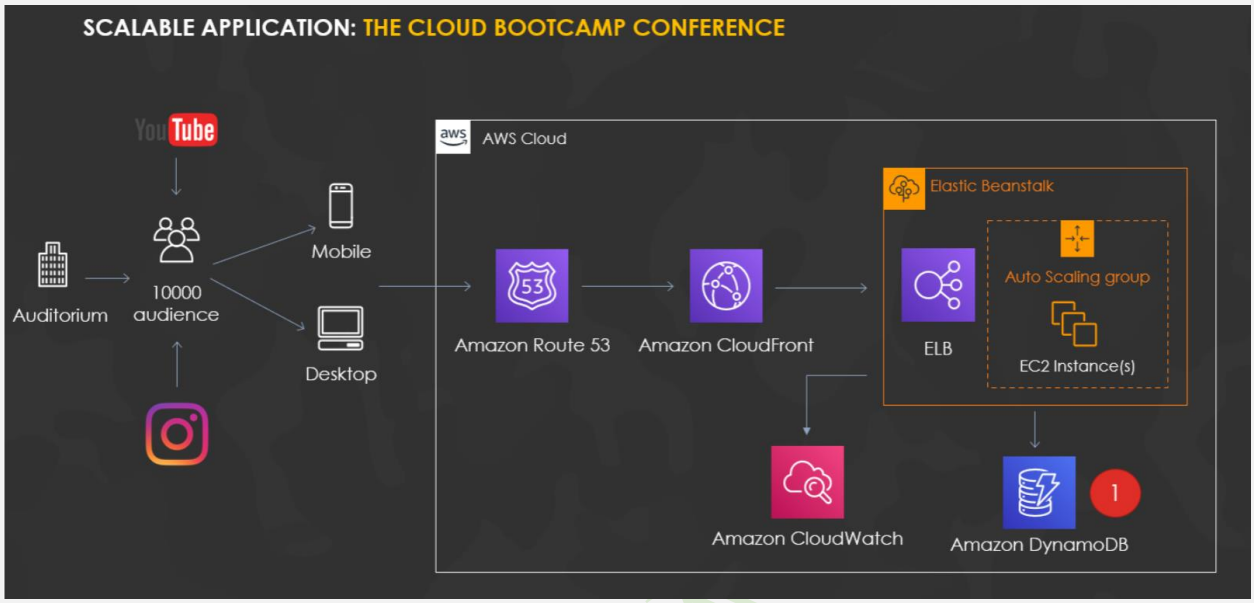
There is no additional charge for AWS Elastic Beanstalk–you pay only for the AWS resources actually used to store and run your application.



Tools & Technologies covered:

- AWS public cloud
- Elastic Beanstalk
- Cloudfront
- Auto Scaling
- Cloudwatch
- DynamoDB
- Route 53
- AWS Pricing Calculator
- Mobaxterm SSH client

➢ Architectural Diagram:



This migration project will be completed in 3 implementation phases.

- **Project implementation Phases:**

  - Implementation Phase 1: Create a DynamoDB table and configure Elastic Beanstalk to create an application.
  - Implementation Phase 2: Create Cloudfront distribution.
  - Implementation Phase 3: Perform Load testing and test application auto scale up

- **Implementation:**

  - **Implementation Phase 1: Create a DynamoDB table and configure Elastic Beanstalk to create an application**

  - Create DynamoDB table
    - Create a new table
    - Name: users
    - Primary key: email
  - Configure Elastic Beanstalk
  - Create application

  - **Implementation Phase 2: Create Cloudfront distribution**

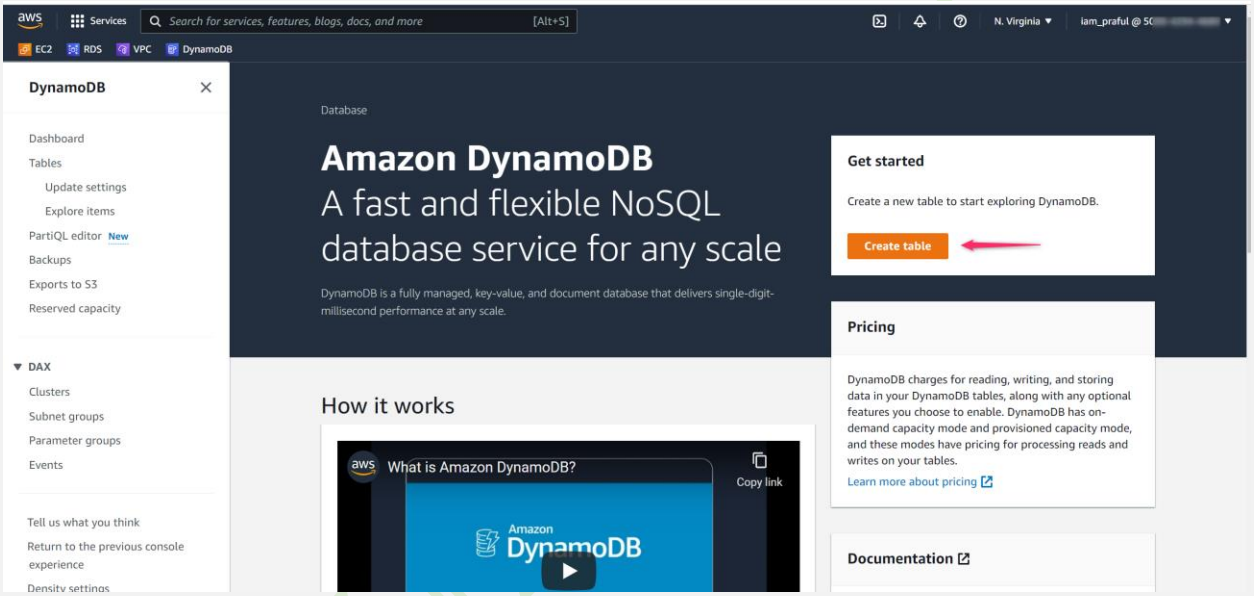
  - **Implementation Phase 3: Perform load testing and verify application auto scale up**

  - **Solution execution in Action:**

Implementation Phase 1 in Action : Create a DynamoDB table and configure Elastic Beanstalk to create an application

Create a DynamoDB table

1. Create a table: users , primary key: email



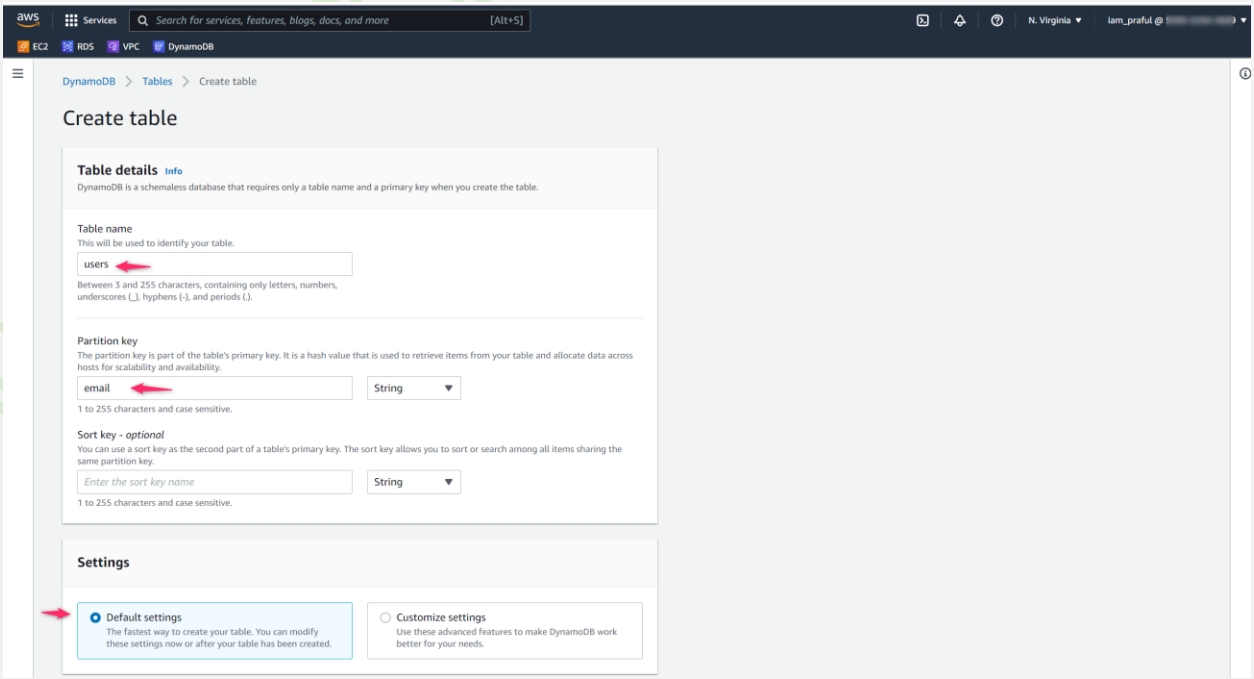
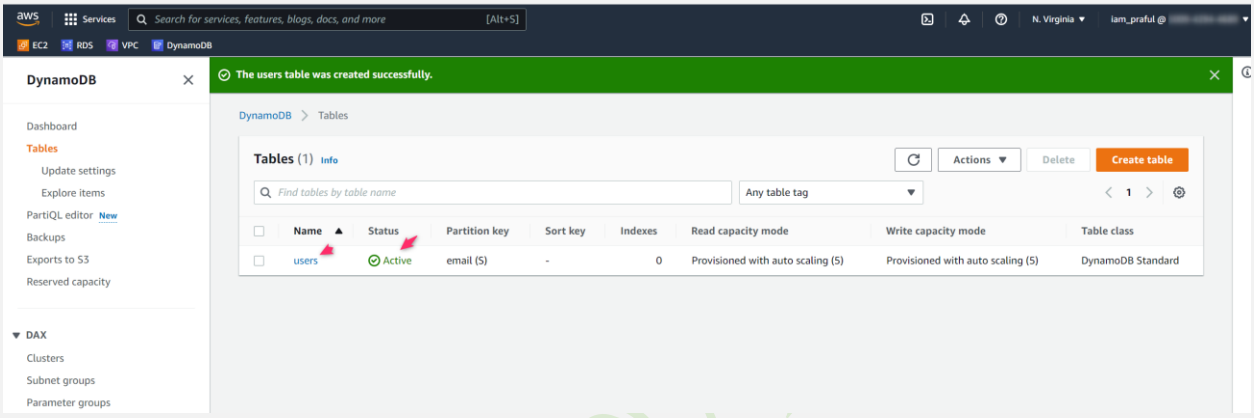
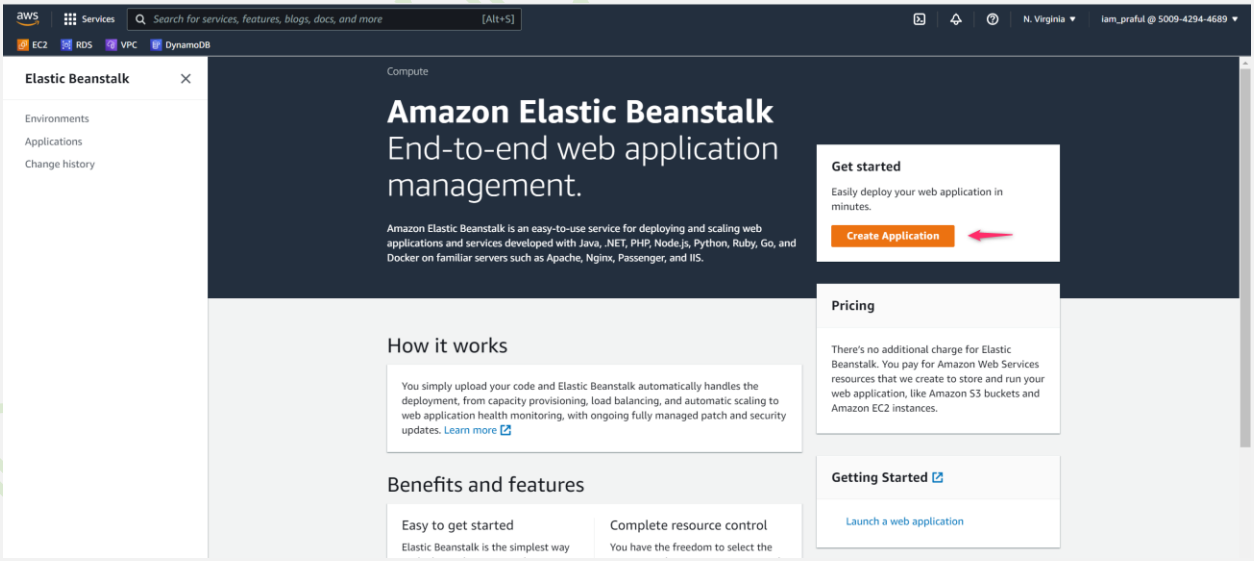


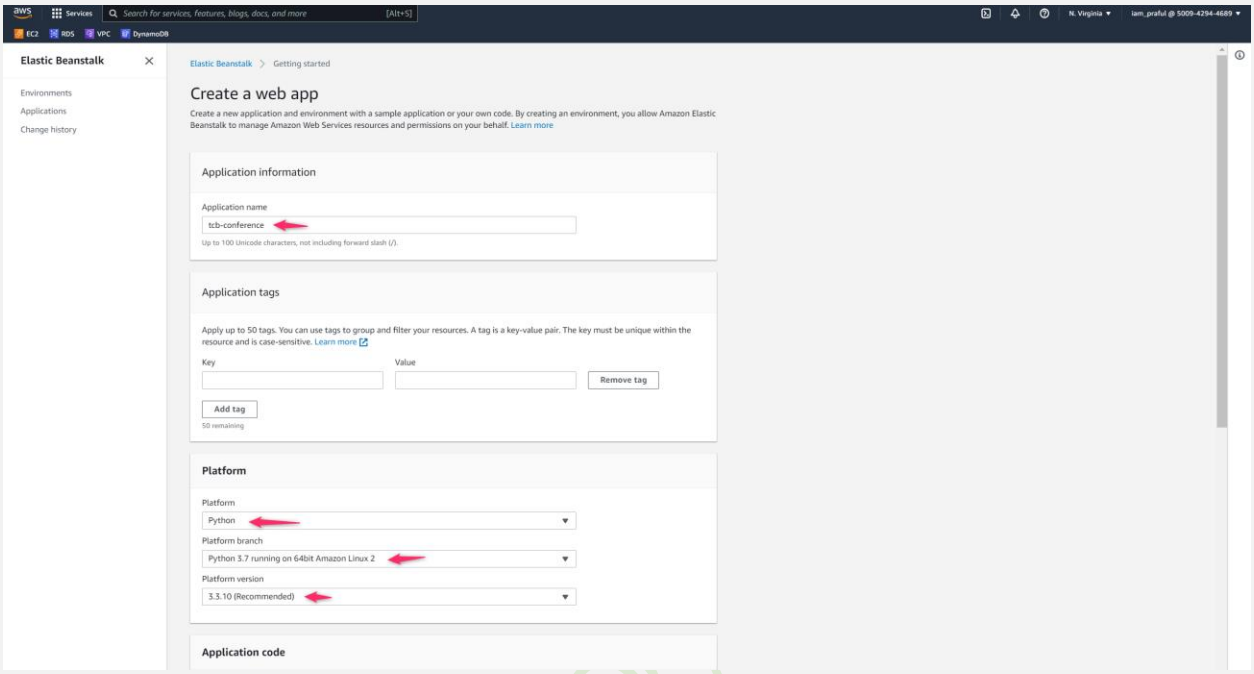
Table created successfully



Configure Elastic Beanstalk

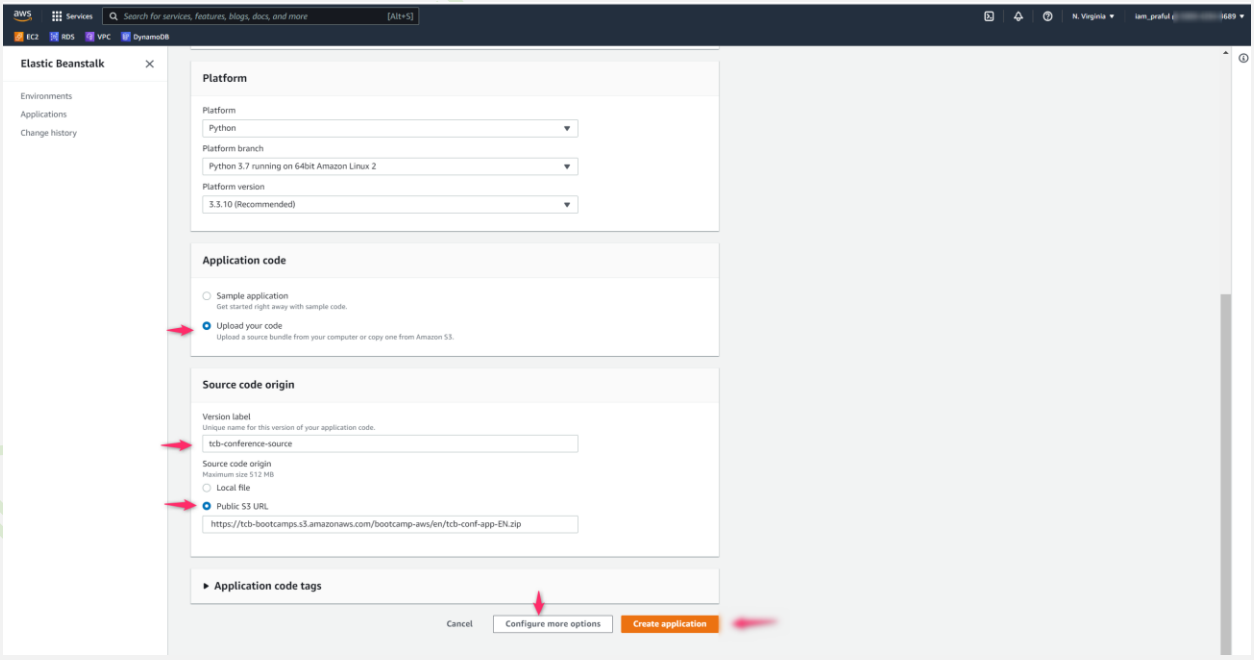


Create a WebApp



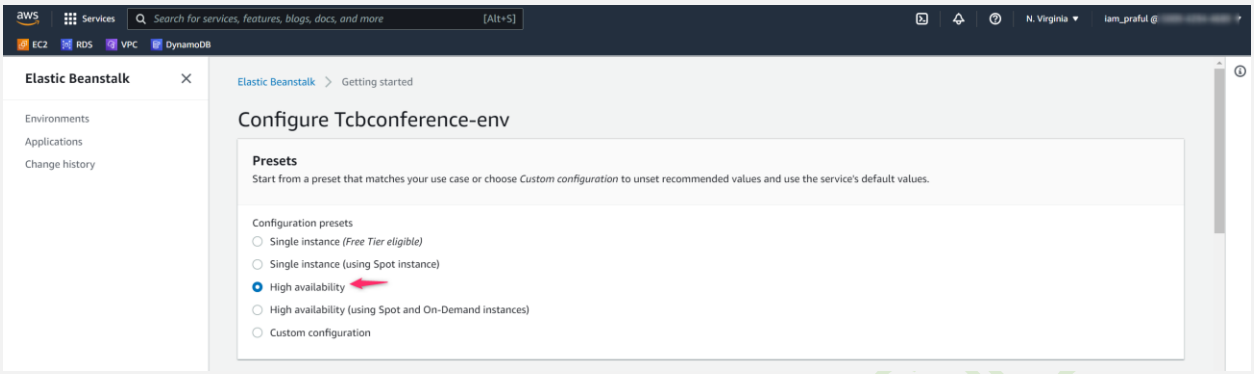
Web app source code url

https://tcb-bootcamps.s3.amazonaws.com/bootcamp-aws/en/tcb-conf-app-EN.zip

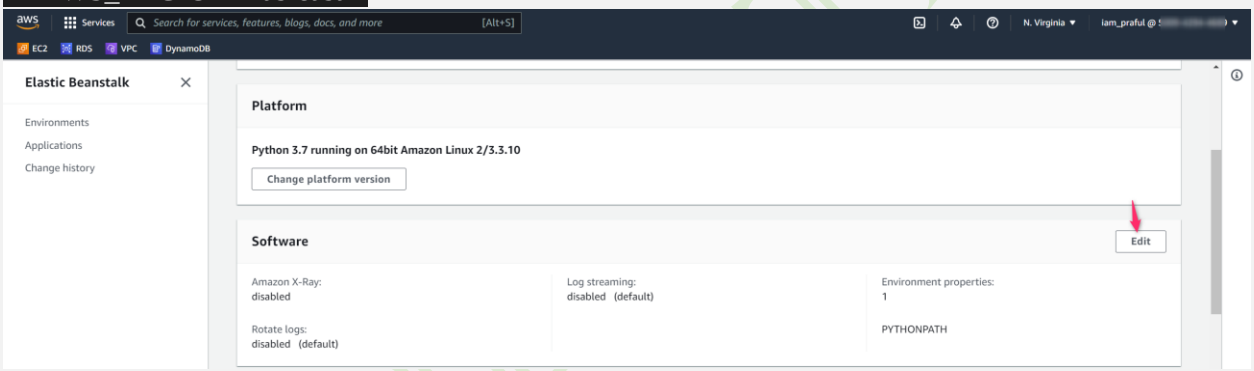


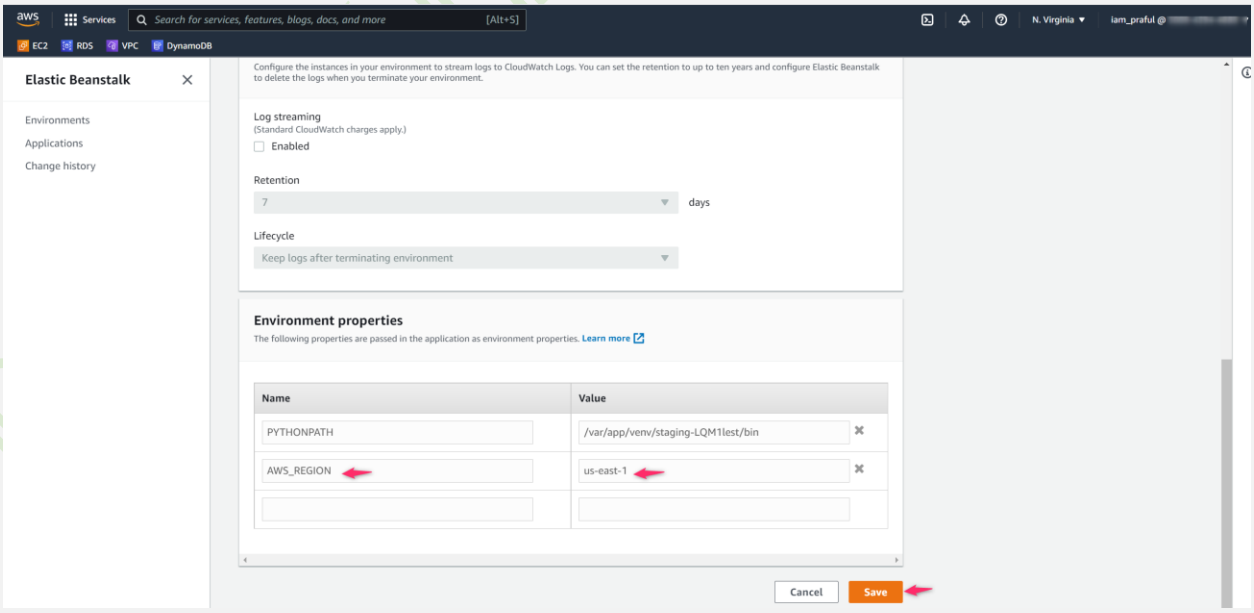
Click on Configure more options

- Presets:
-- Configuration presets: High availability



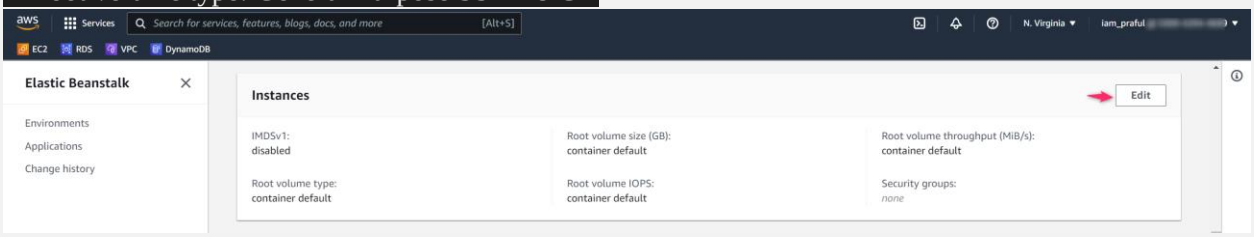
- Software:
-- Environment properties:
--- AWS_REGION = us-east-1

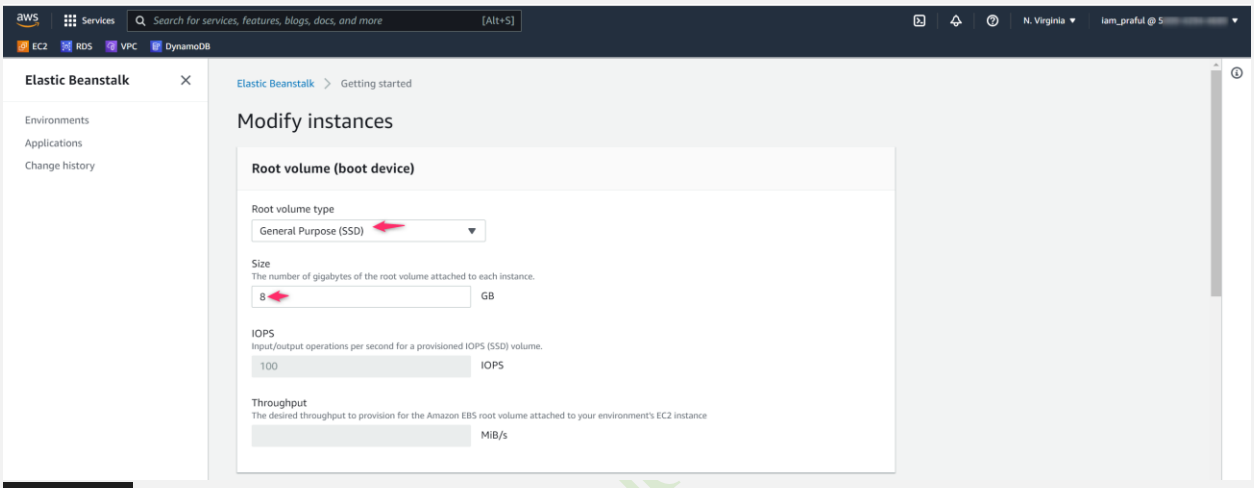




- Instances:

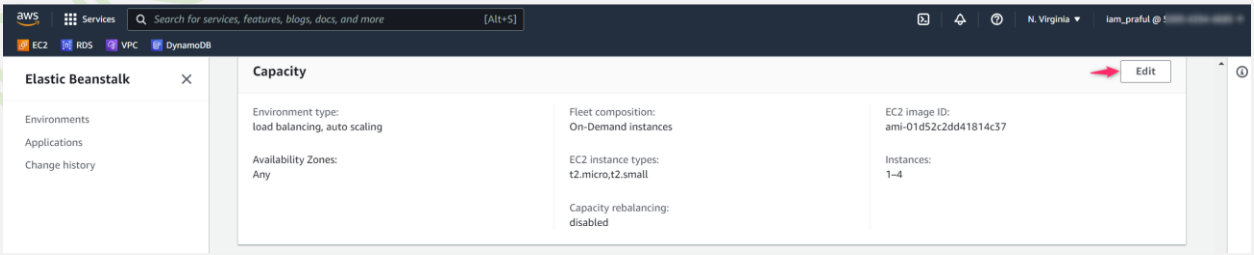
-- Root volume type: General Purpose SSD - 8 GB

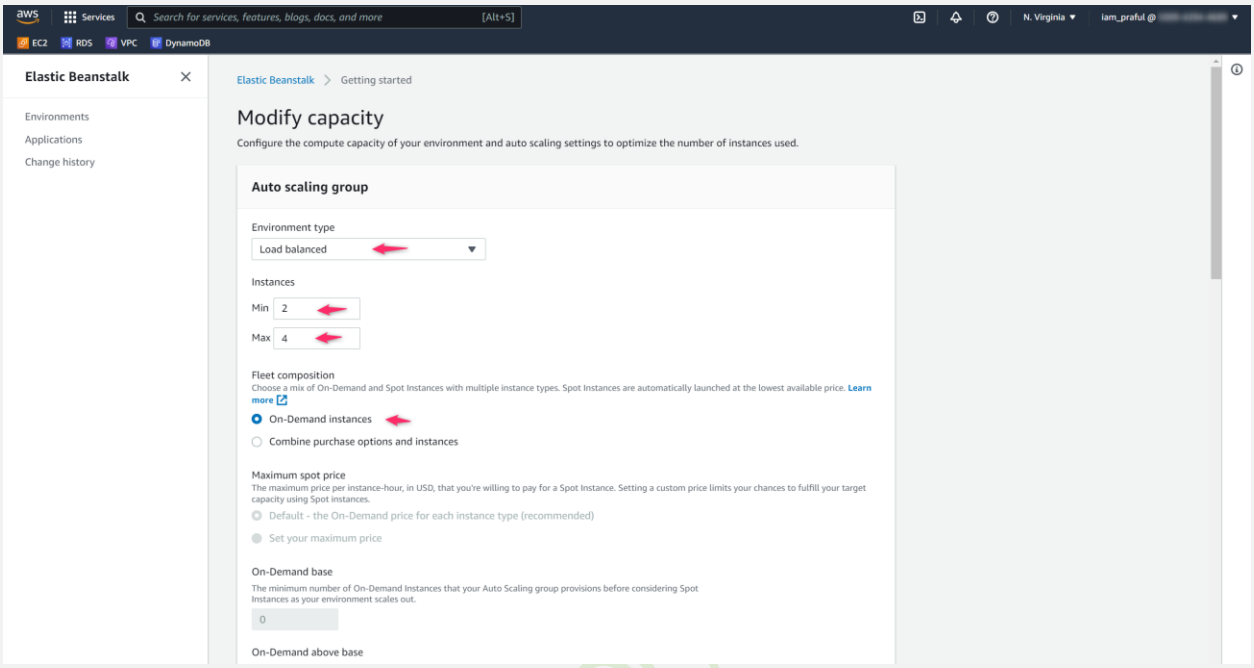


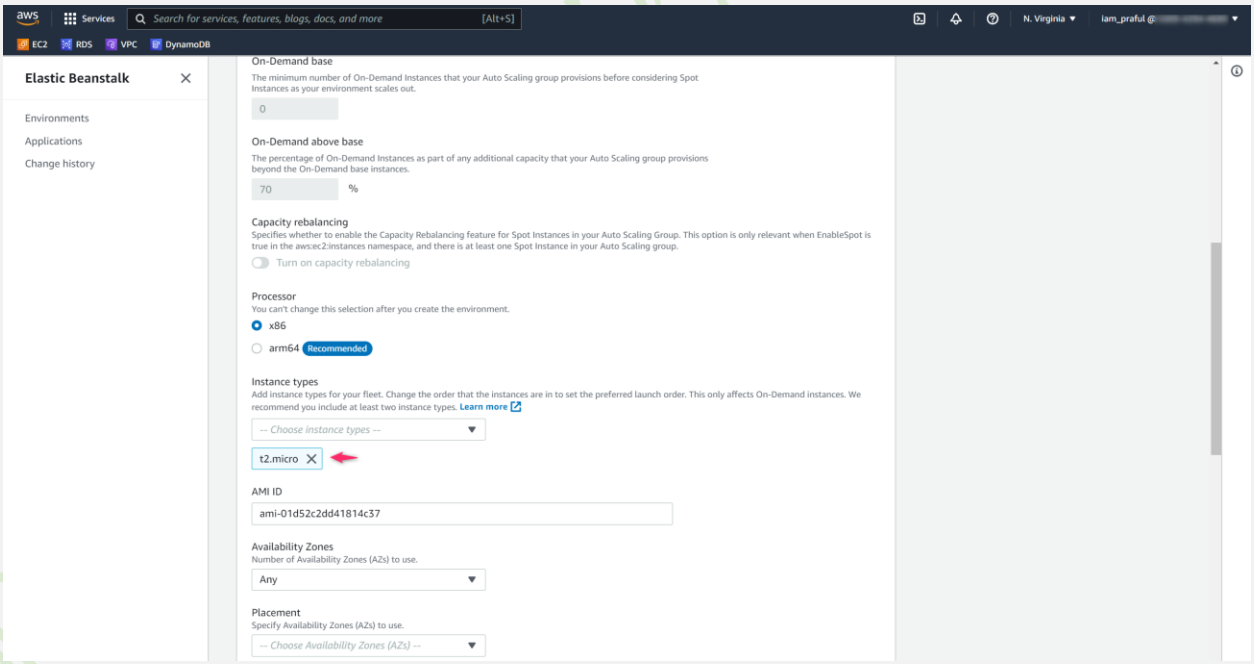


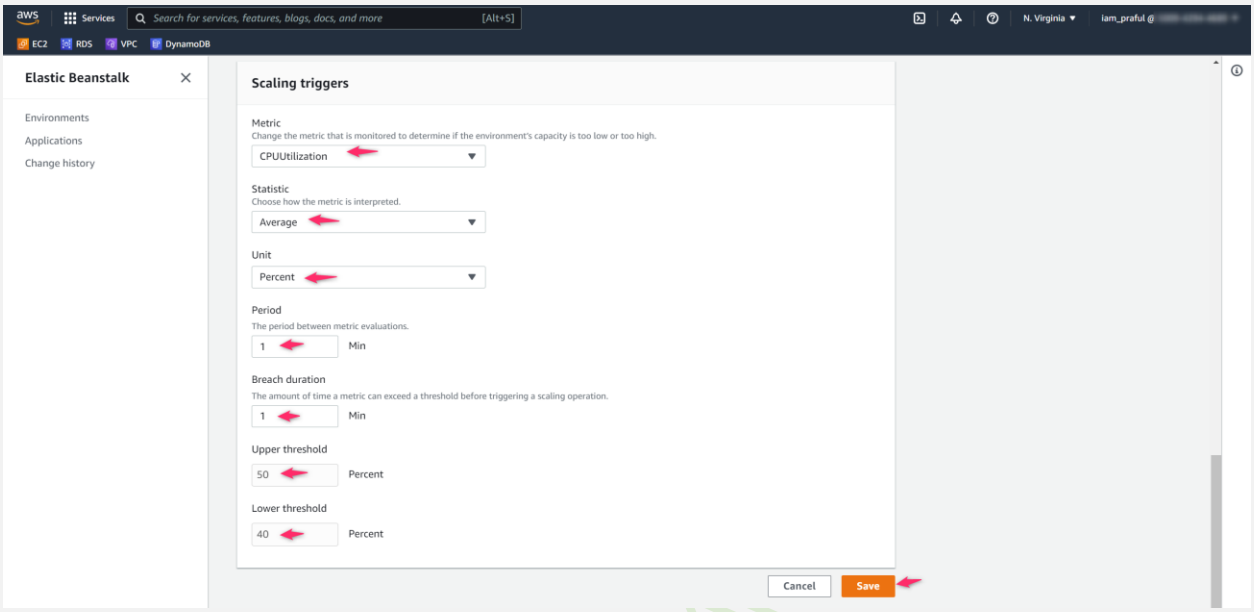
- Capacity:
-- Instances Min: 2
-- Instances Max: 4
-- Instance type: t2-micro

-- Scaling triggers:
--- CPUUtilization
--- Unit: Percent
--- Period: 1 Min
--- Breach durantion: 1 Min
--- Upper threshold: 50
--- Lower threshold: 40

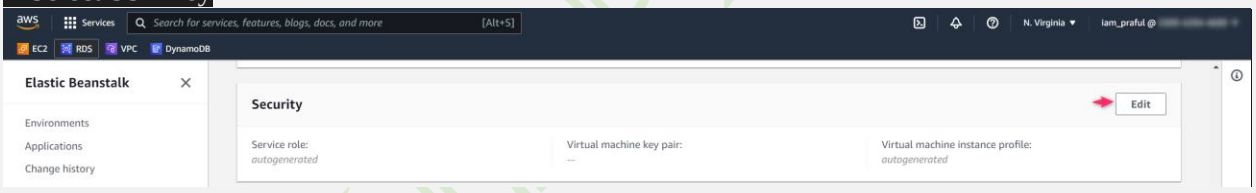


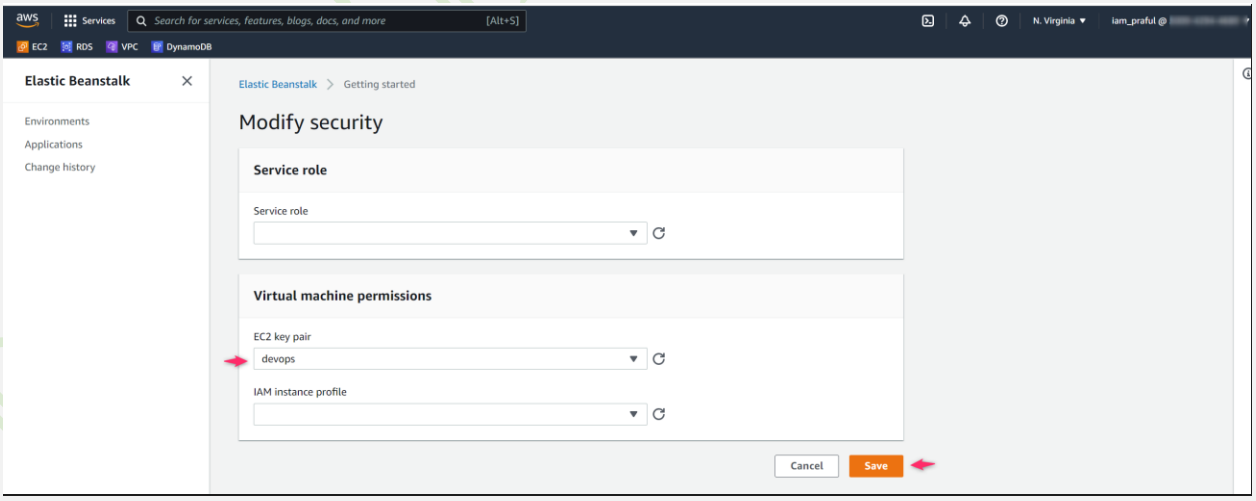

Services
Search for services, features, blogs, docs, and more
[Alt+S]
N. Virginia
EC2
RDS
VPC
DynamoDB
Elastic Beanstalk
Environments
Applications
Change history
Elastic Beanstalk > Getting started
Modify capacity
Configure the compute capacity of your environment and auto scaling settings to optimize the number of instances used.
Auto scaling group
Environment type
Load balanced
Instances
Min 2
Max 4
Fleet composition
Choose a mix of On-Demand and Spot Instances with multiple instance types. Spot Instances are automatically launched at the lowest available price. Learn more
On-Demand instances
Combine purchase options and instances
Maximum spot price
The maximum price per instance-hour, in USD, that you're willing to pay for a Spot Instance. Setting a custom price limits your chances to fulfill your target capacity using Spot instances.
Default - the On-Demand price for each instance type (recommended)
Set your maximum price
On-Demand base
The minimum number of On-Demand Instances that your Auto Scaling group provisions before considering Spot Instances as your environment scales out.
0
On-Demand above base


Services
Search for services, features, blogs, docs, and more
[Alt+S]
N. Virginia
EC2
RDS
VPC
DynamoDB
Elastic Beanstalk
Environments
Applications
Change history
On-Demand base
The minimum number of On-Demand Instances that your Auto Scaling group provisions before considering Spot Instances as your environment scales out.
0
On-Demand above base
The percentage of On-Demand Instances as part of any additional capacity that your Auto Scaling group provisions beyond the On-Demand base instances.
70
%
Capacity rebalancing
Specifies whether to enable the Capacity Rebalancing feature for Spot Instances in your Auto Scaling Group. This option is only relevant when EnableSpot is true in the aws:ec2:instances namespace, and there is at least one Spot Instance in your Auto Scaling group.
Turn on capacity rebalancing
Processor
You can't change this selection after you create the environment.
x86
arm64
Recommended
Instance types
Add instance types for your fleet. Change the order that the instances are in to set the preferred launch order. This only affects On-Demand instances. We recommend you include at least two instance types. Learn more
-- Choose instance types --
t2.micro
AMI ID
ami-01d52c2dd41814c37
Availability Zones
Number of Availability Zones (AZs) to use.
Any
Placement
Specify Availability Zones (AZs) to use.
-- Choose Availability Zones (AZs) --



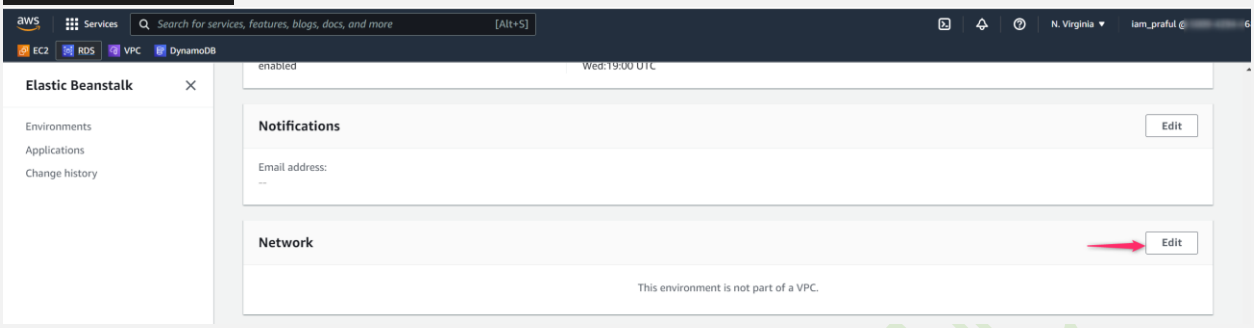
- Security:
-- Select SSH key

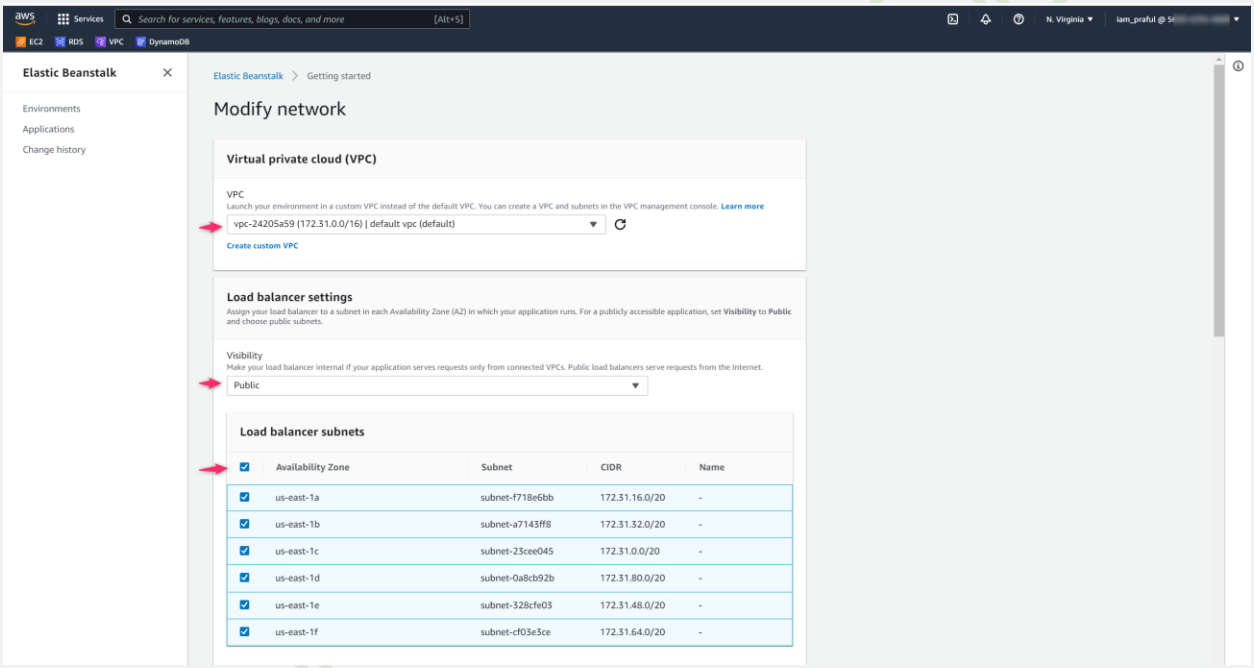


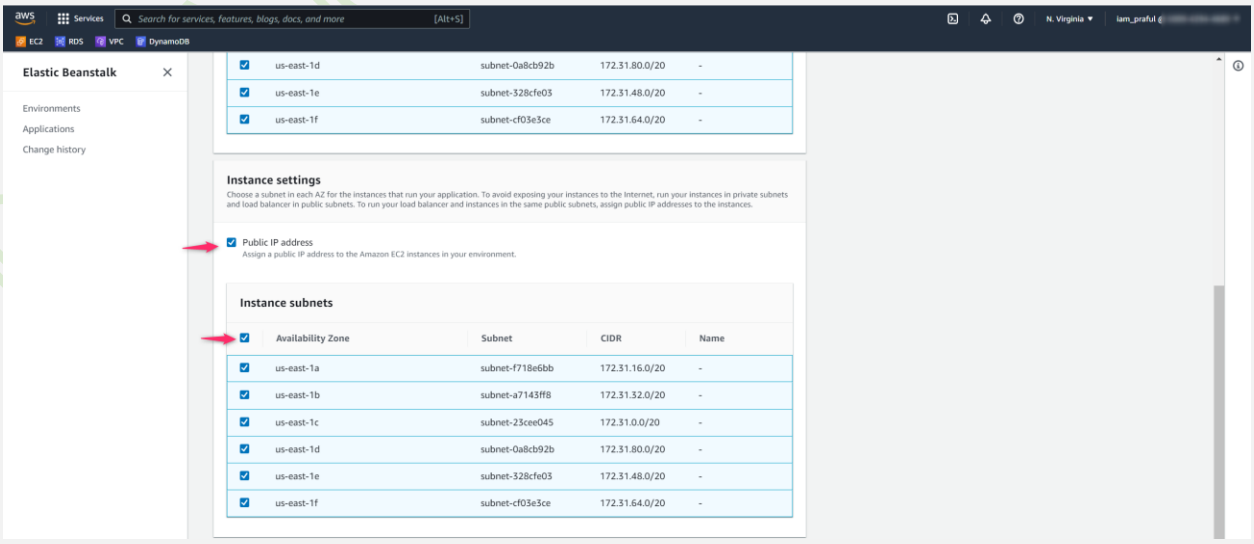


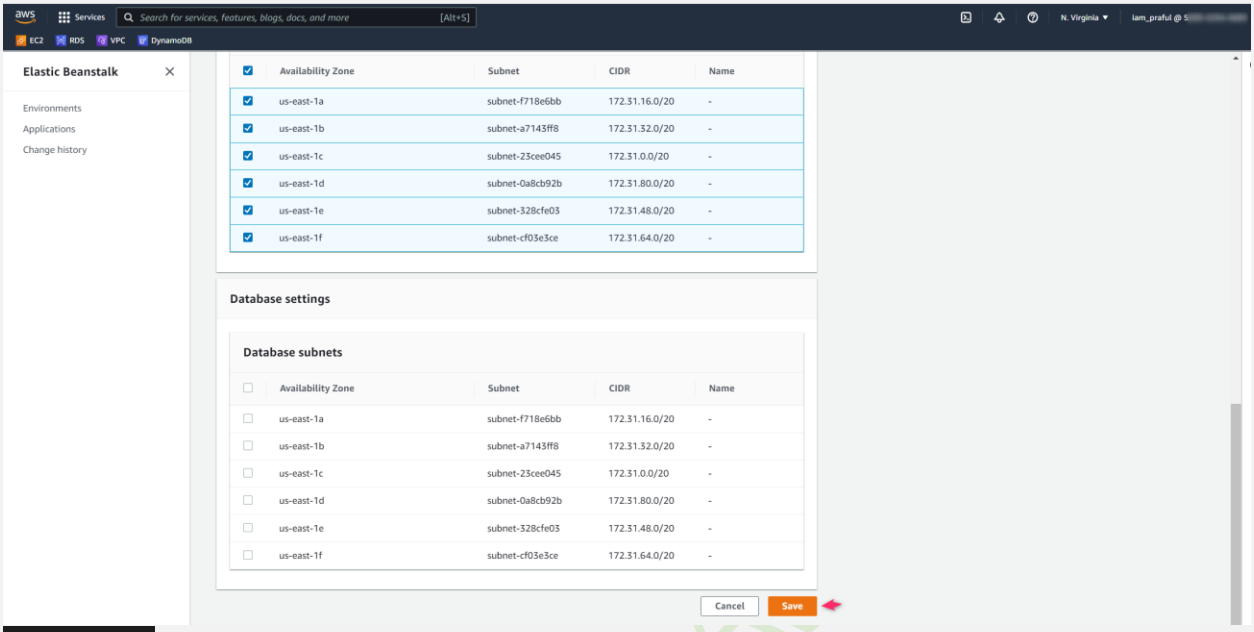
- Networking:
-- VPC: Default:
-- Add IP Public

-- Select all subnets

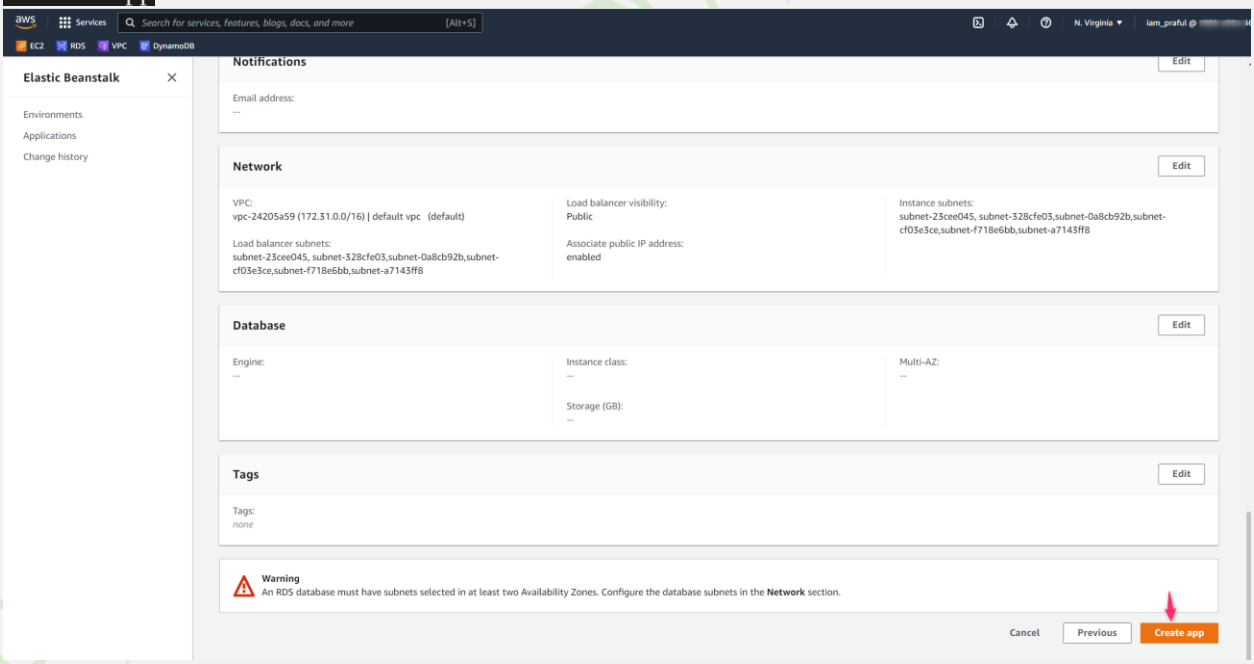


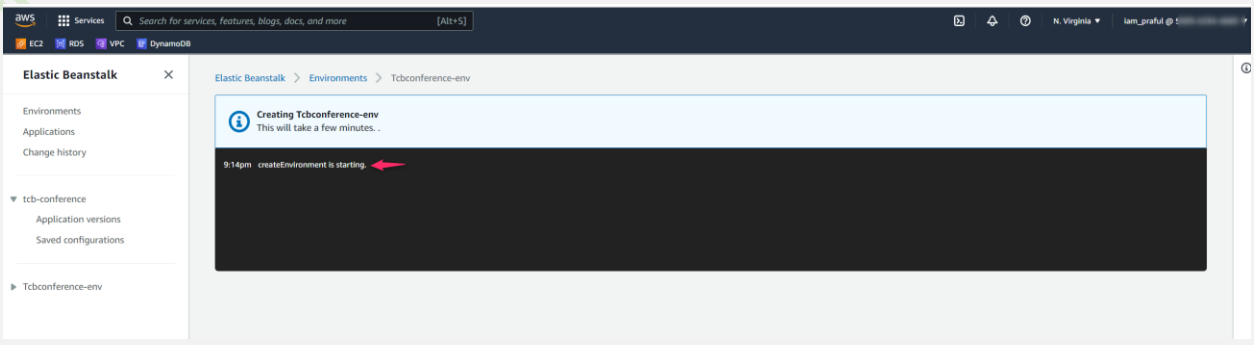


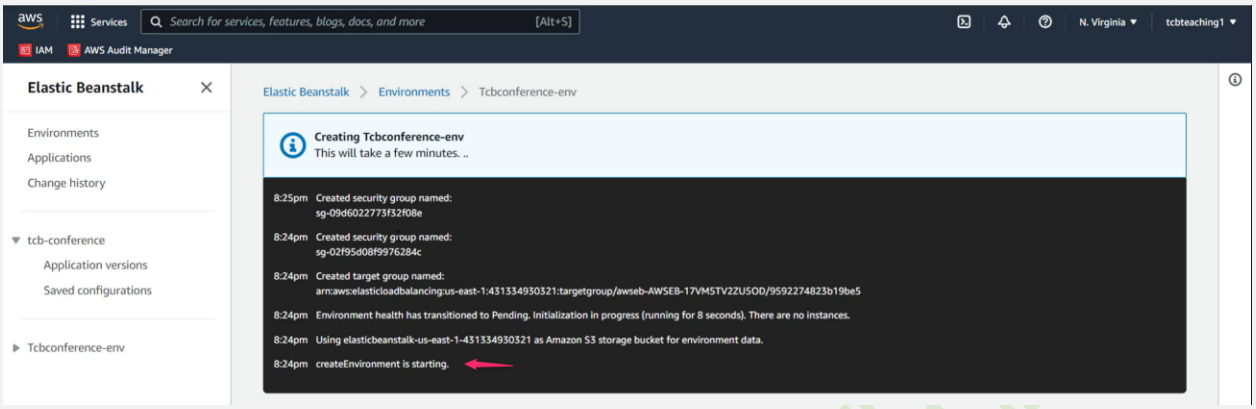


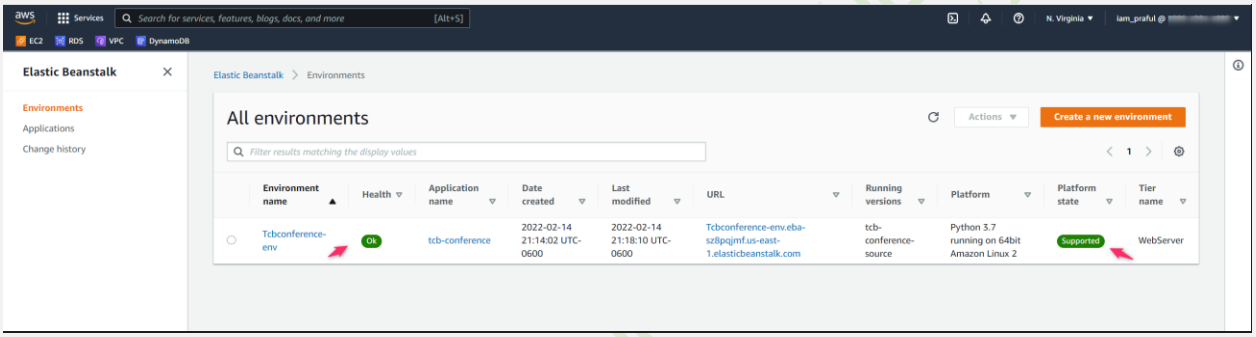

Elastic Beanstalk
Environments
Applications
Change history
Availability Zone
Subnet
CIDR
Name
us-east-1a subnet-f718e6bb 172.31.16.0/20
us-east-1b subnet-a7143ff8 172.31.32.0/20
us-east-1c subnet-23cee045 172.31.0.0/20
us-east-1d subnet-0a8cb92b 172.31.80.0/20
us-east-1e subnet-328cfe03 172.31.48.0/20
us-east-1f subnet-cf03e3ce 172.31.64.0/20
Database settings
Database subnets
Cancel
Save

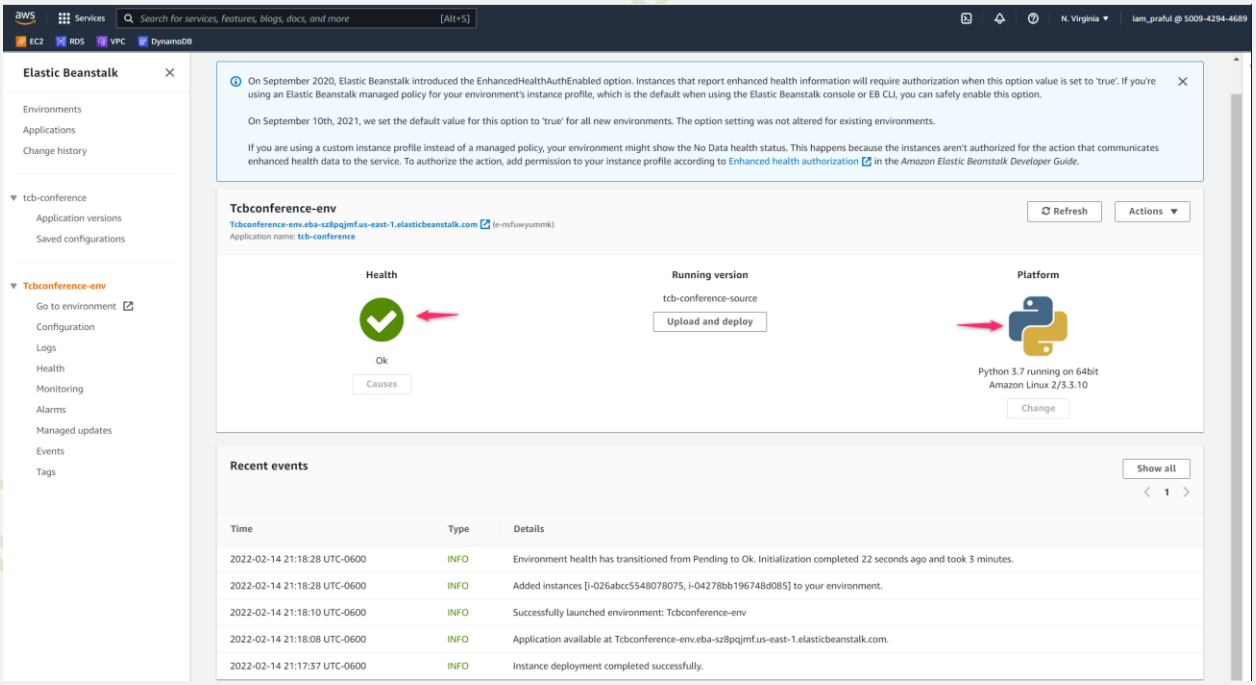
- Create App


Elastic Beanstalk
Notifications
Edit
Email address:
Network
VPC:
vpc-24205a59 (172.31.0.0/16) | default vpc (default)
Load balancer visibility:
Public
Instance subnets:
subnet-23cee045, subnet-328cfe03,subnet-0a8cb92b,subnet-cf03e3ce,subnet-f718e6bb,subnet-a7143ff8
Load balancer subnets:
subnet-23cee045, subnet-328cfe03,subnet-0a8cb92b,subnet-cf03e3ce,subnet-f718e6bb,subnet-a7143ff8
Associate public IP address:
enabled
Database
Engine:
Instance class:
Multi-AZ:
Storage (GB):
Tags
Tags:
none
Warning
An RDS database must have subnets selected in at least two Availability Zones. Configure the database subnets in the Network section.
Cancel
Previous
Create app

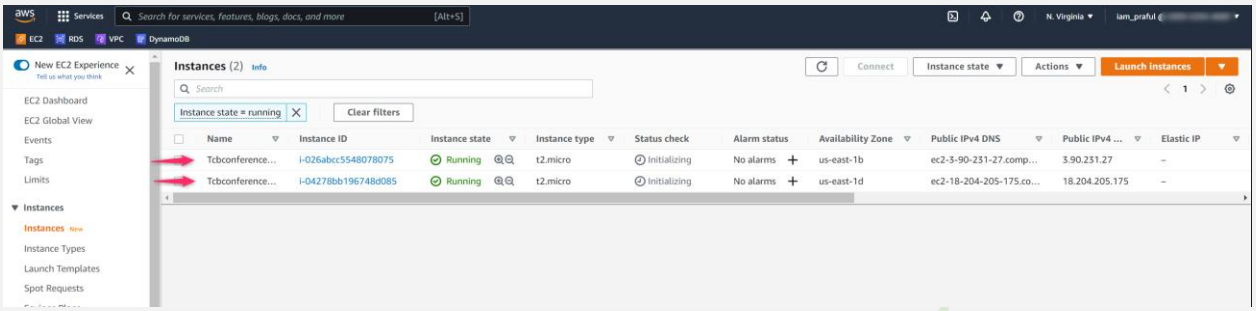

Elastic Beanstalk
Environments
Applications
Change history
tcb-conference
Application versions
Saved configurations
Tcbconference-env
Elastic Beanstalk > Environments > Tcbconference-env
Creating Tcbconference-env
This will take a few minutes. .
9:14pm createEnvironment is starting.



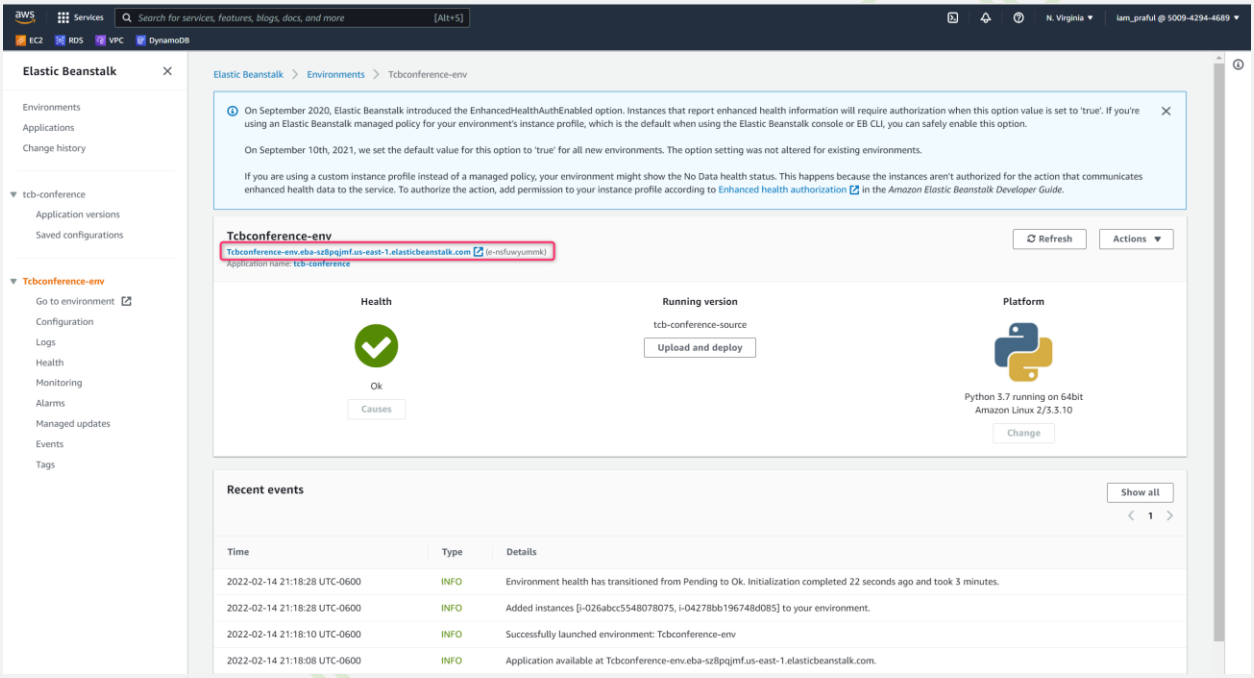


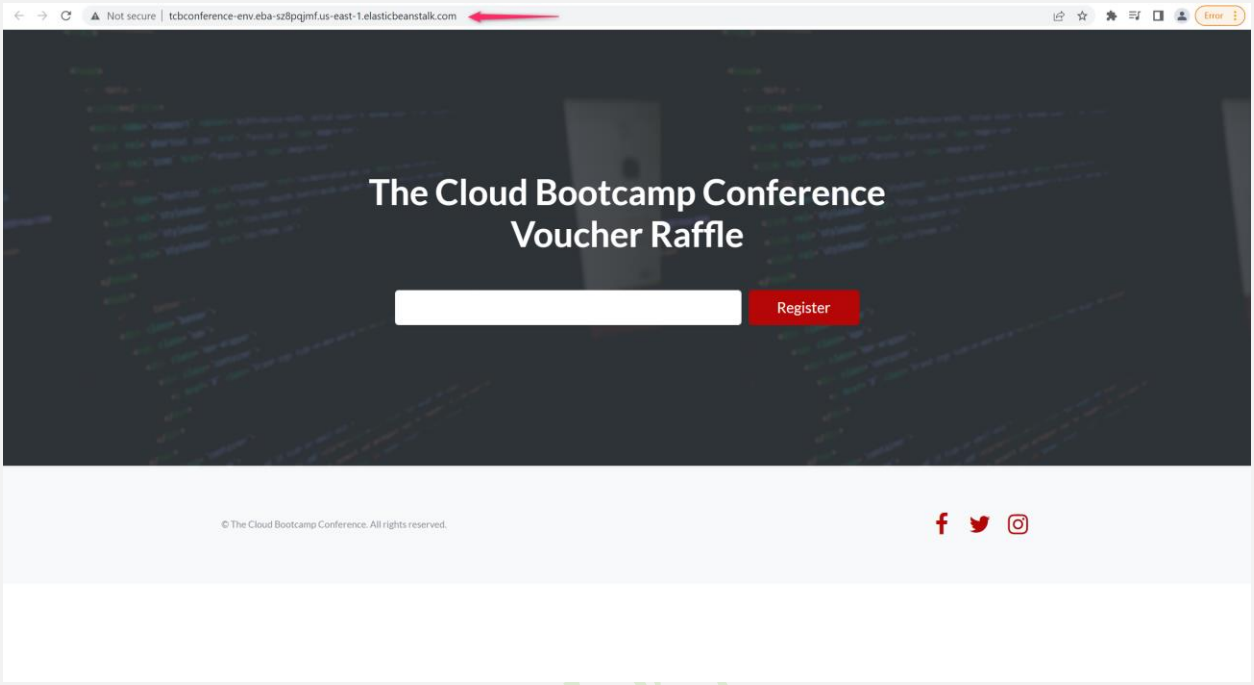


Verify that 2 EC2 instance has started

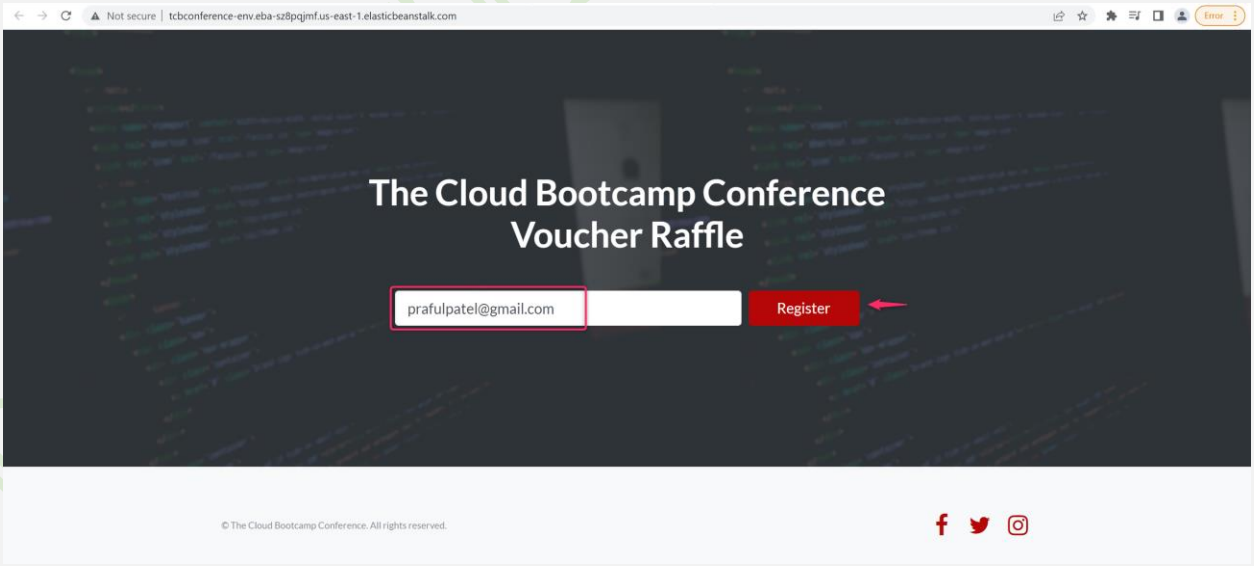


Open an application in web browser

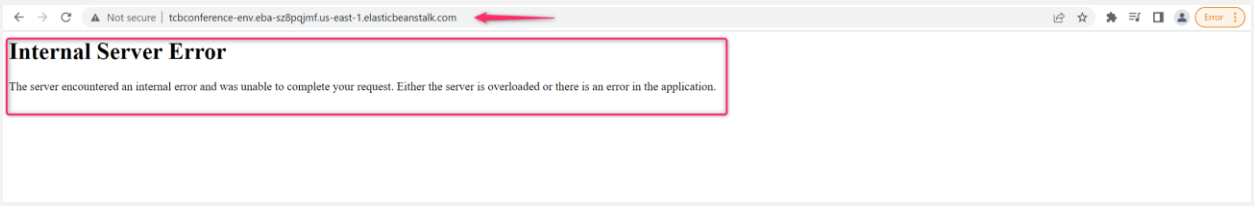




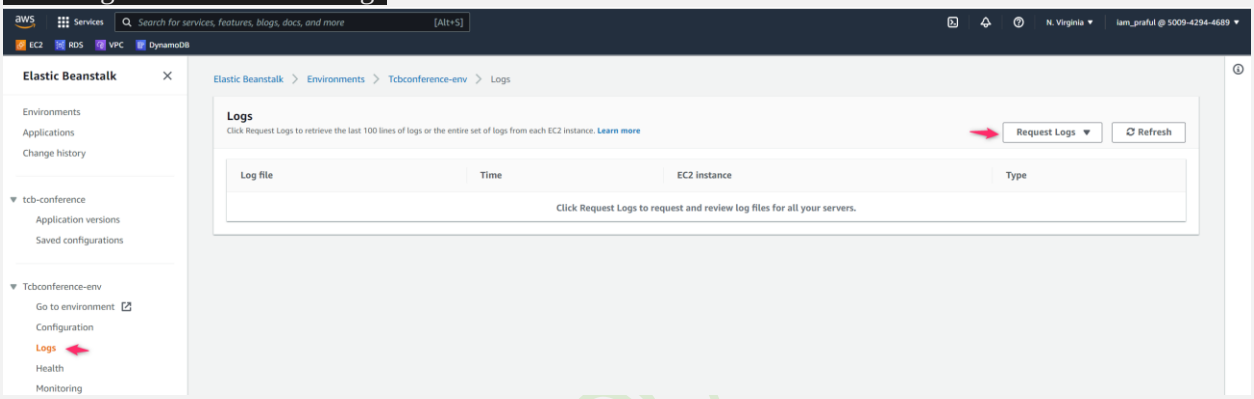
- Once the deploy completes, access the application and simulate a record insertion

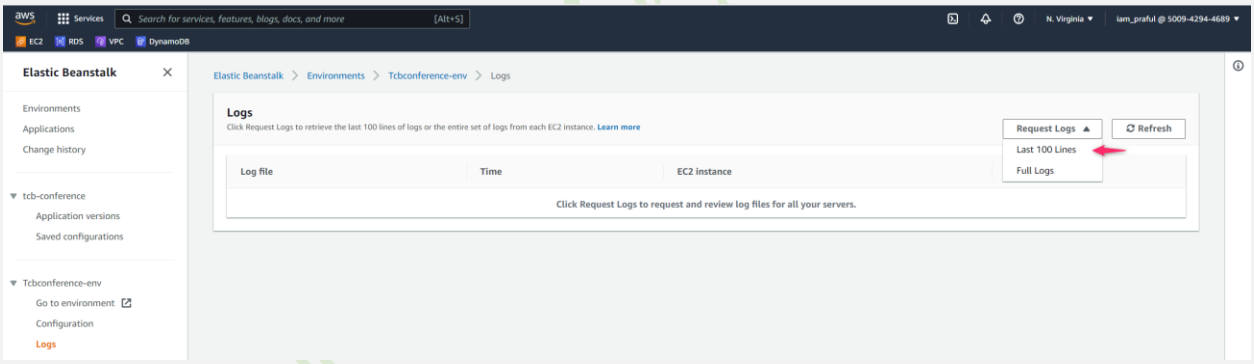


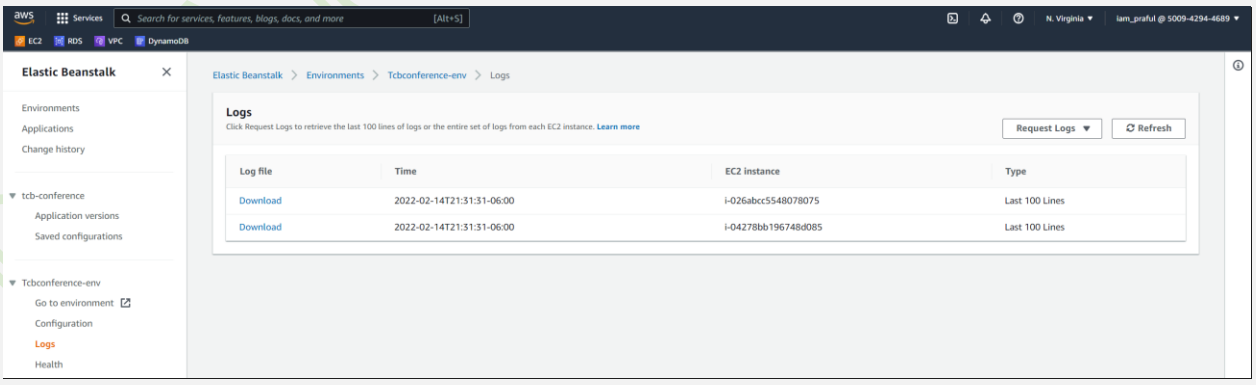
Upon inserting email address it givens an error



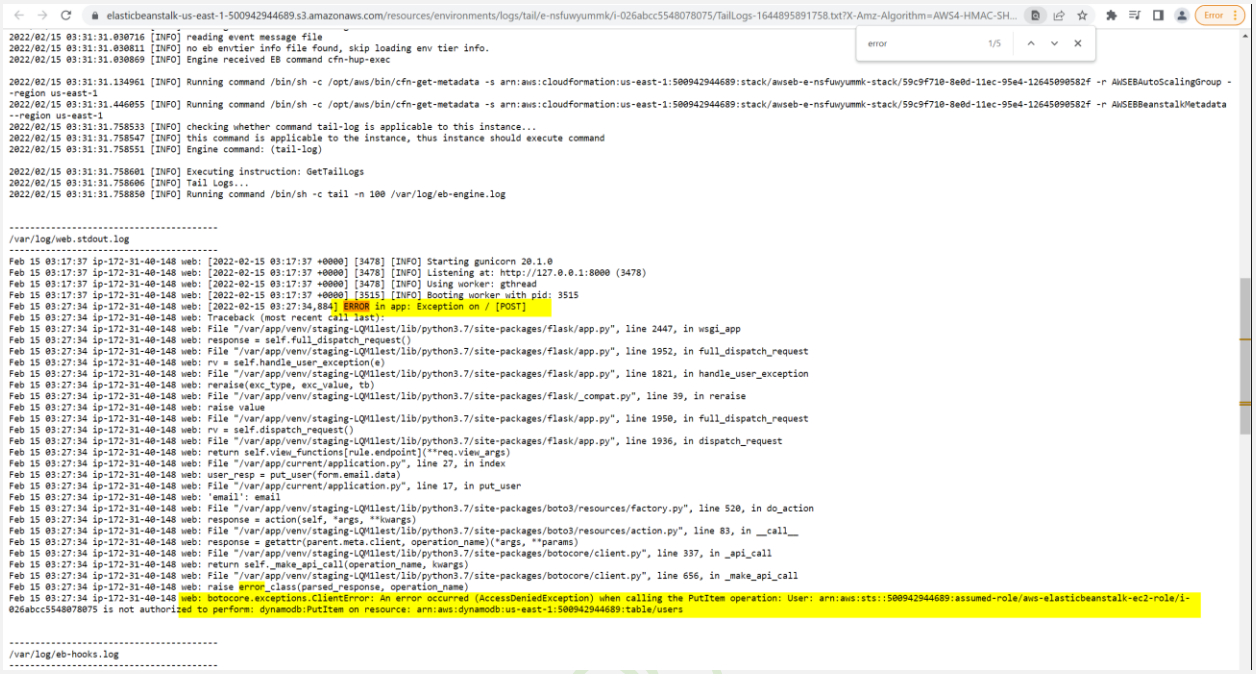
- Investigate the error in the logs





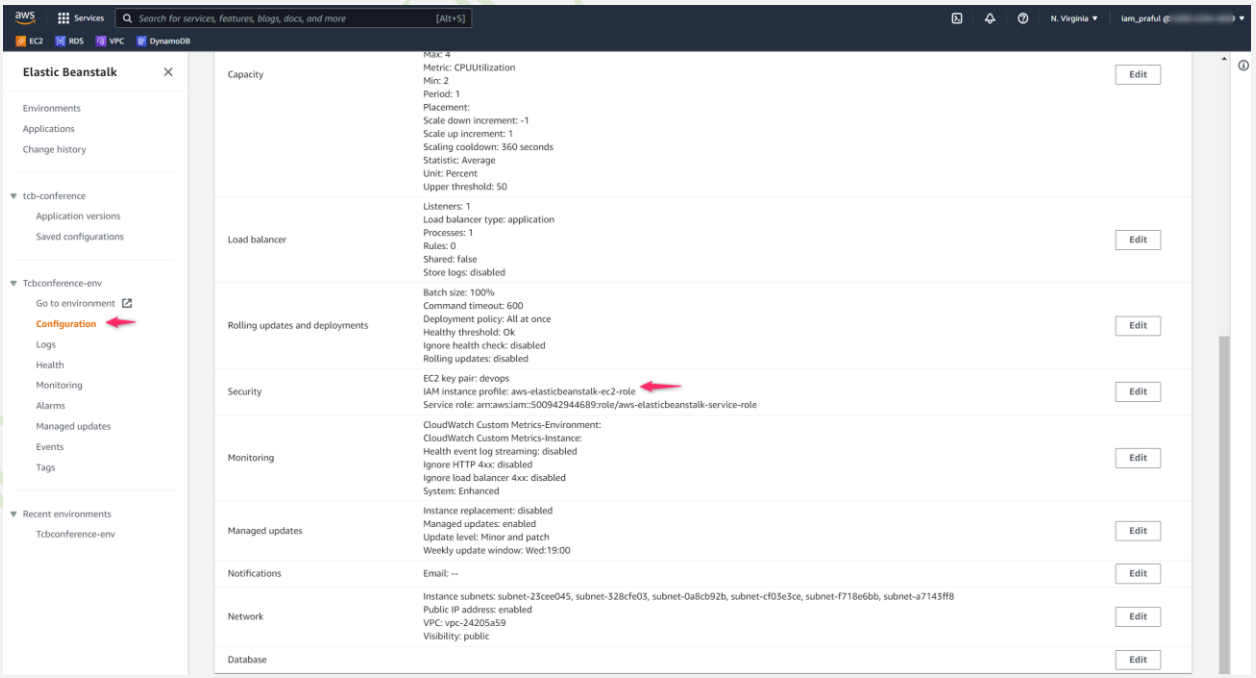


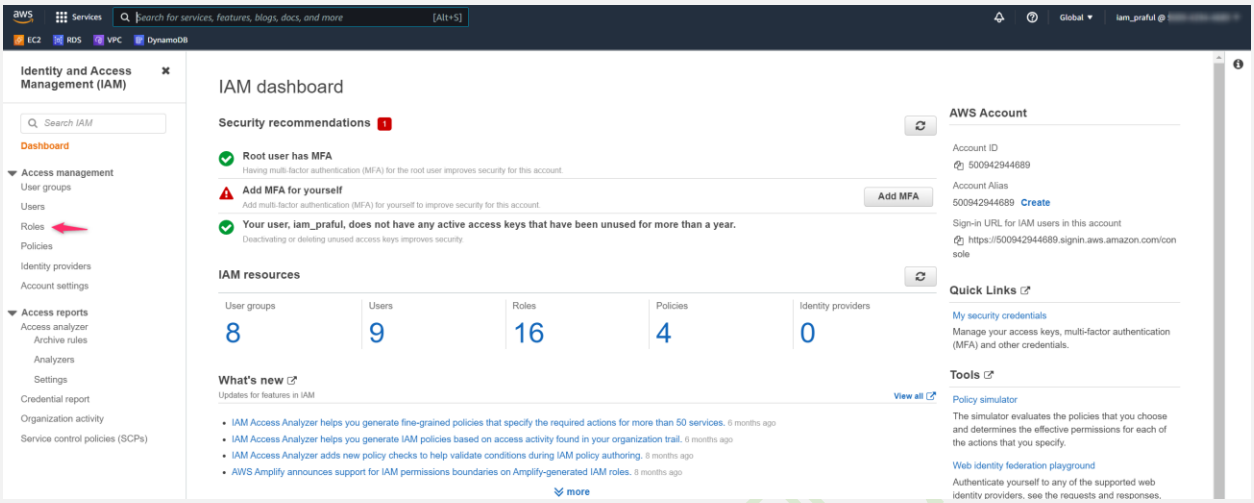
Click on download and observe the log for an error logs



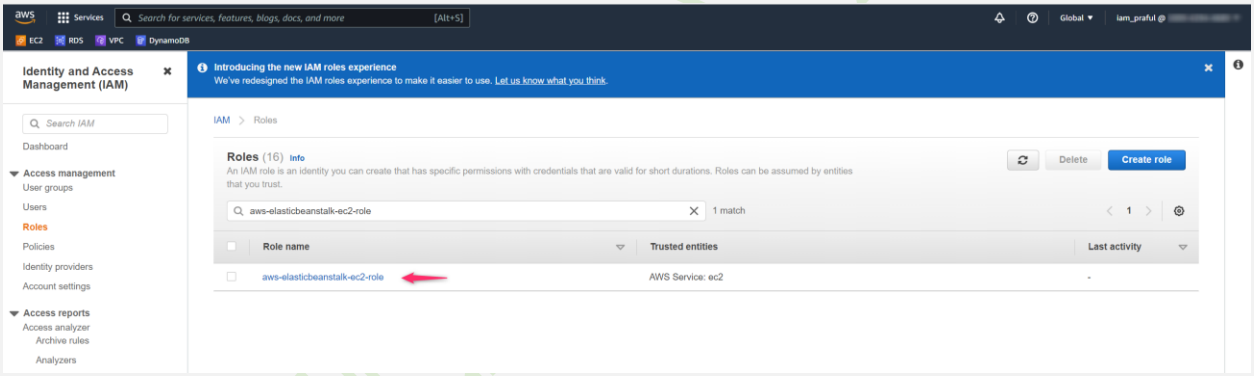
- Add the permission AmazonDynamoDBFullAccess on the EC2 associate role in IAM.

Create and IAM role

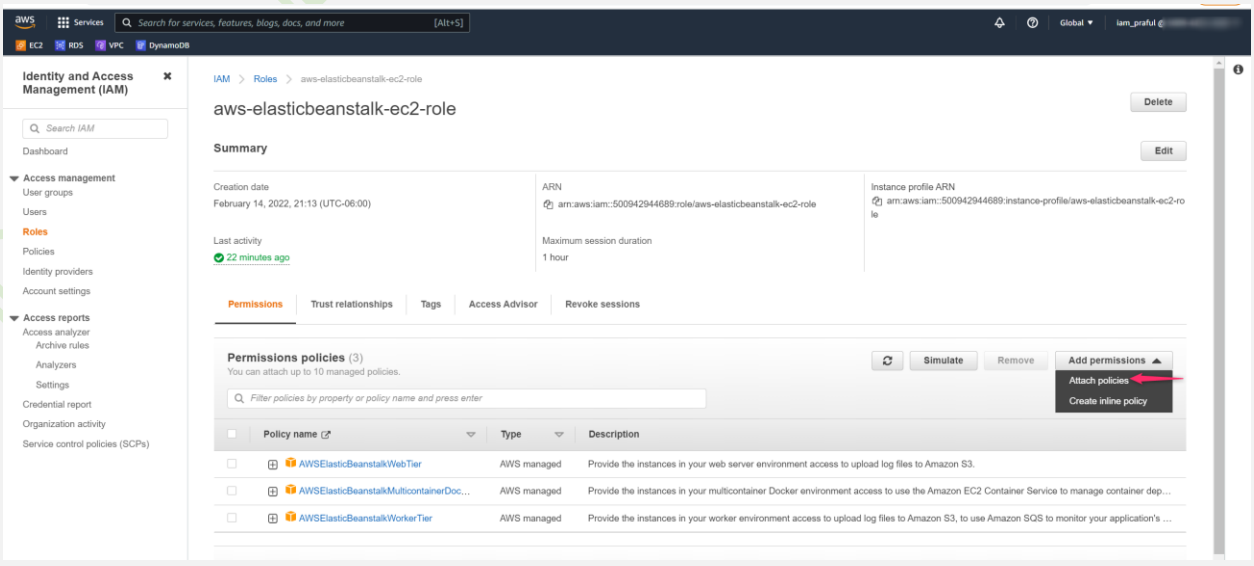


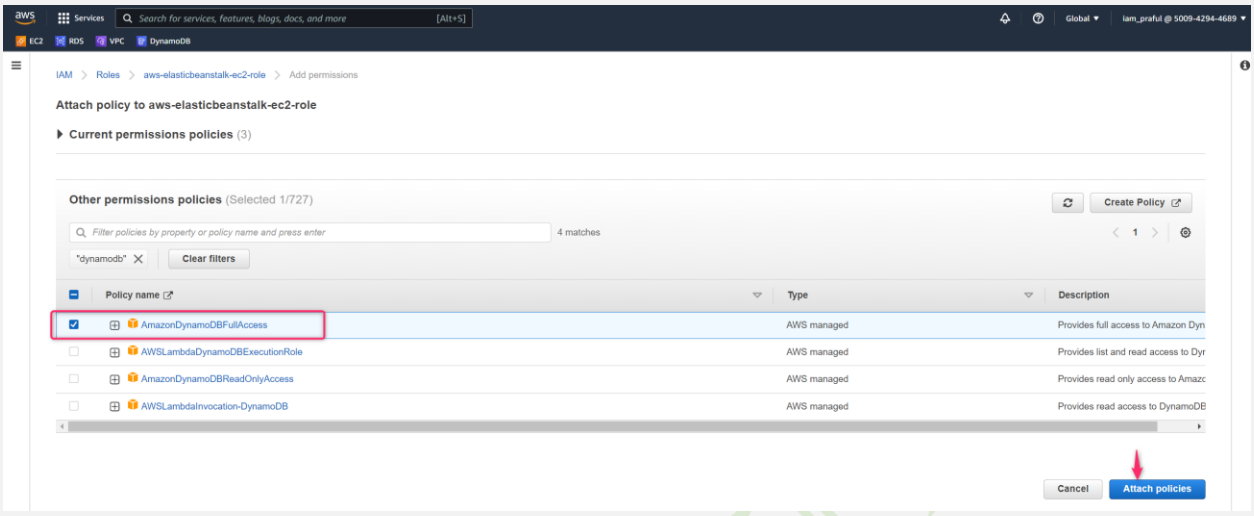


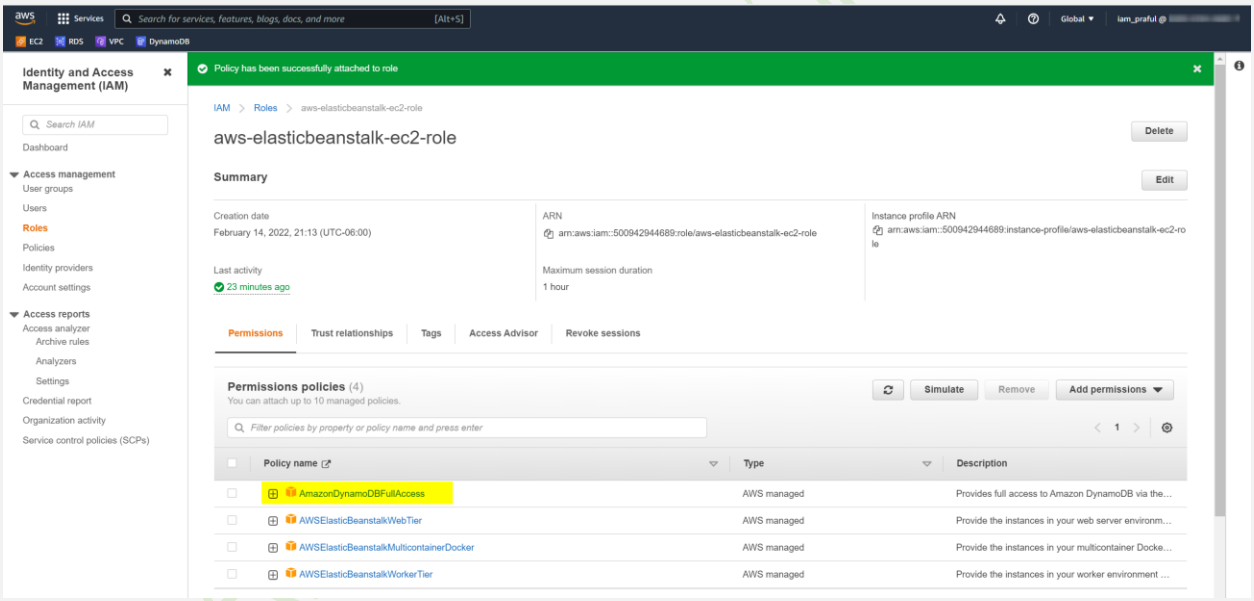
Search the role and click on it



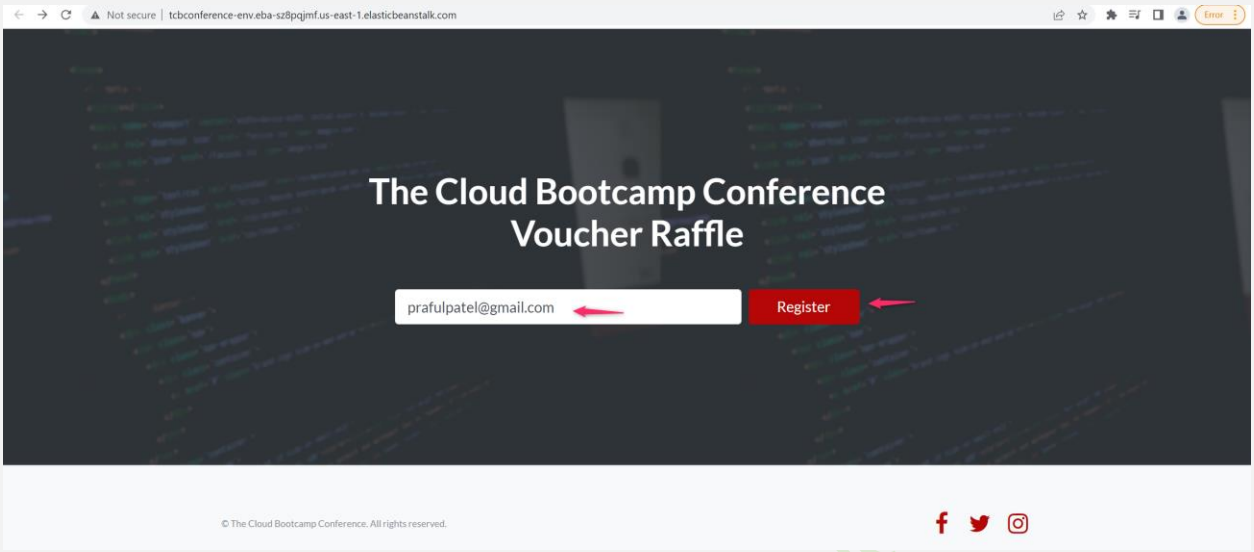
Click on attach policies – attach "DynamodbFullAccess" policy to EC2 role

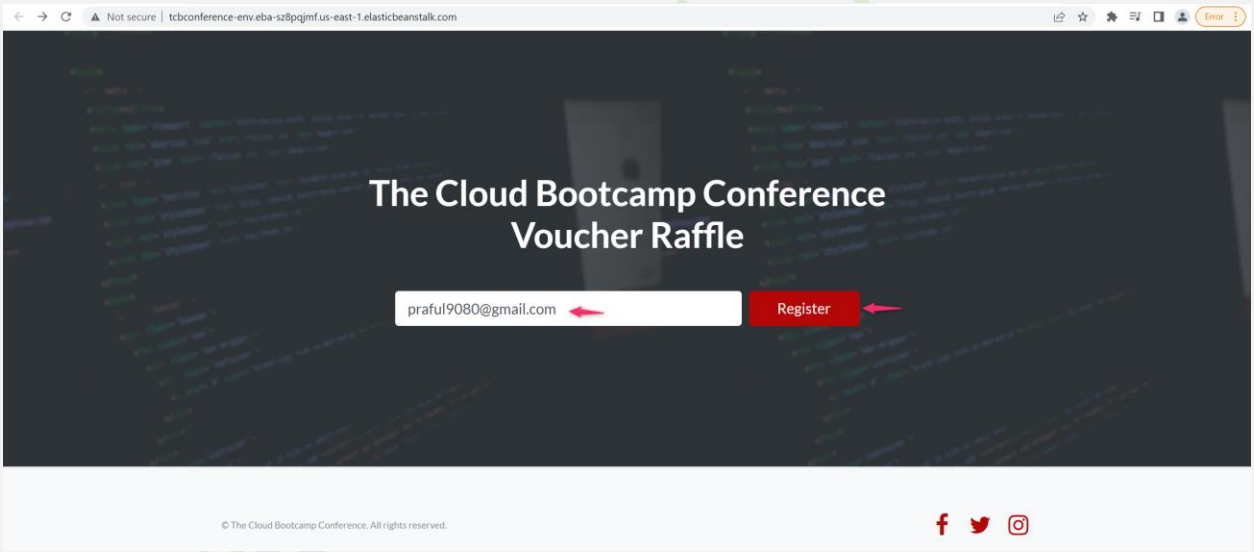




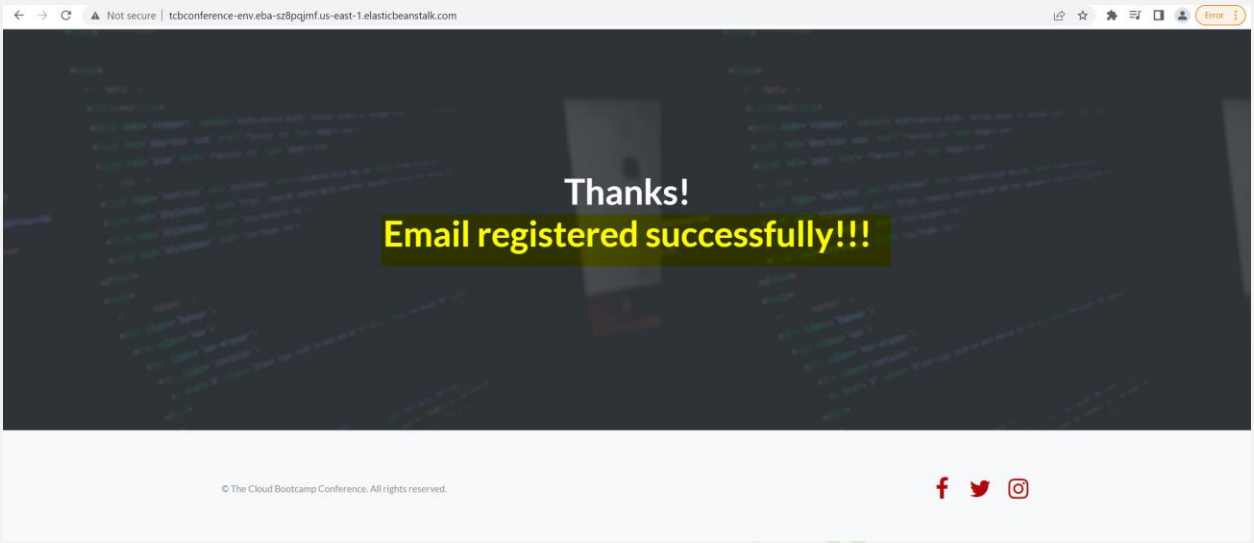


- Try to insert the registry once again. It should be sucessful.

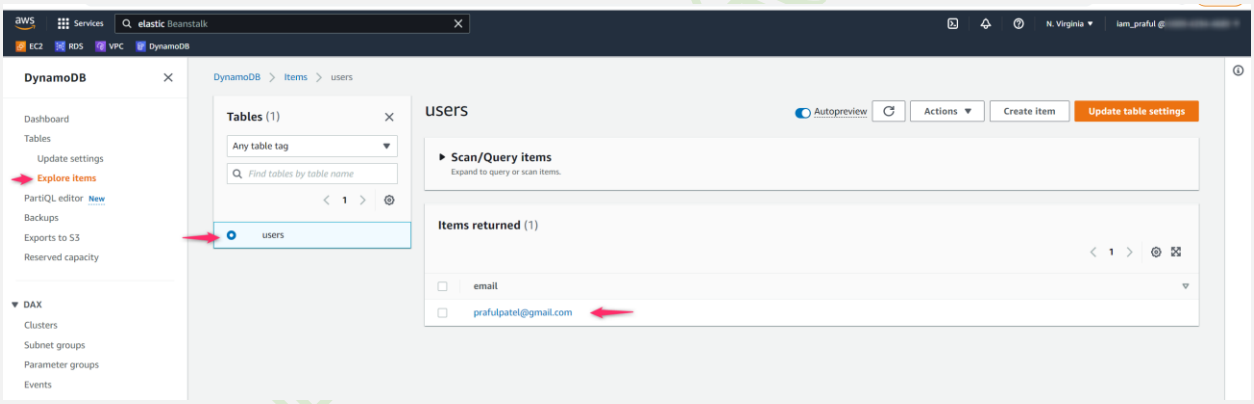


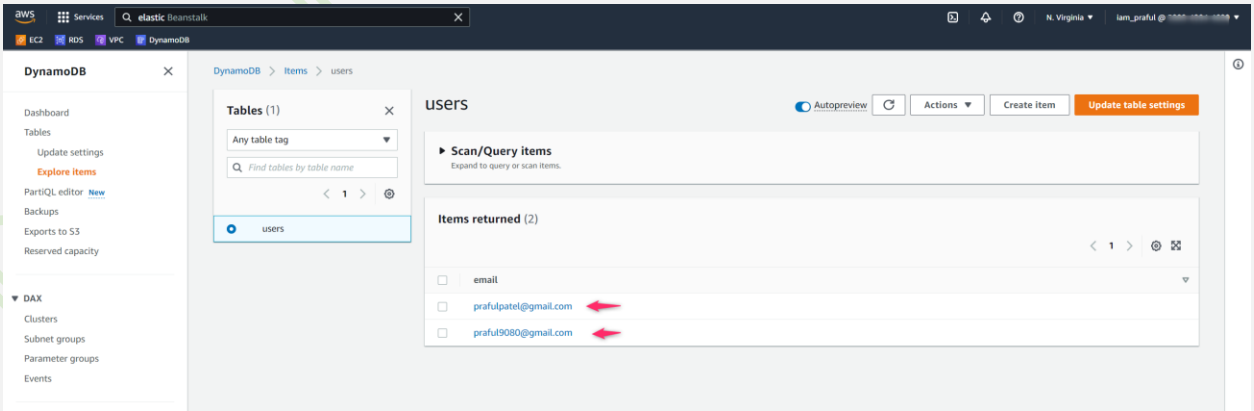


Data has added successfully from frontend web application

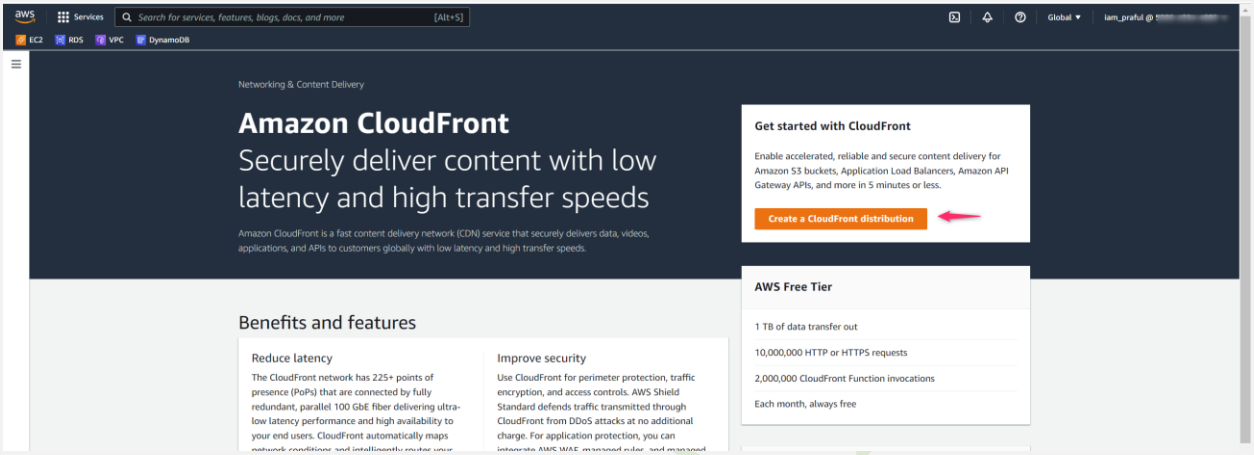


Verify that data has been added successfully to the backend database

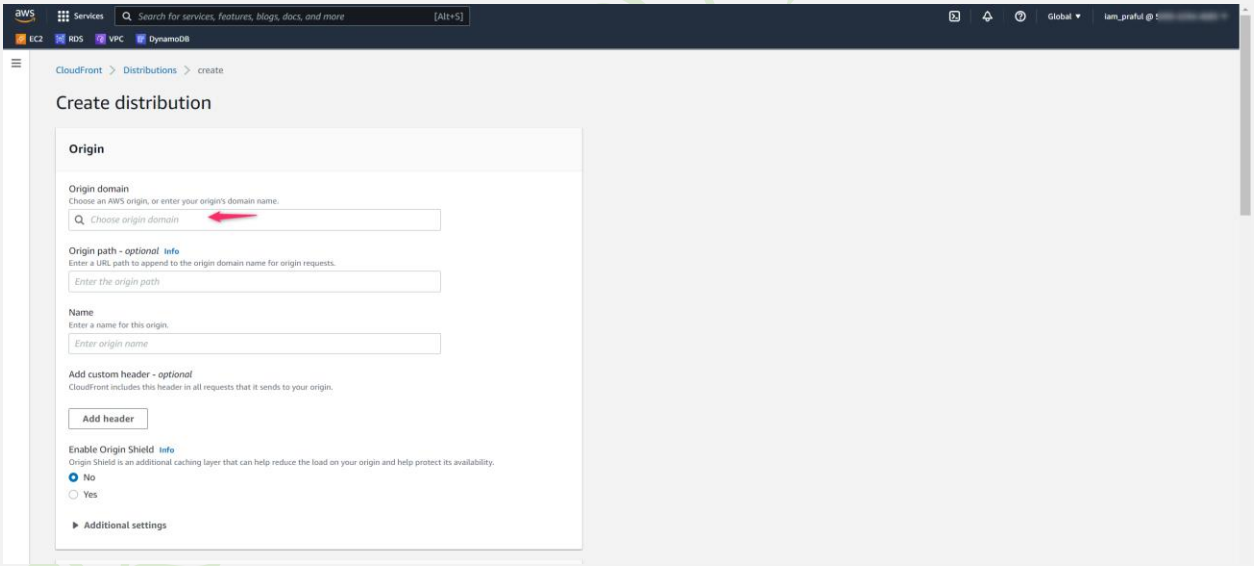


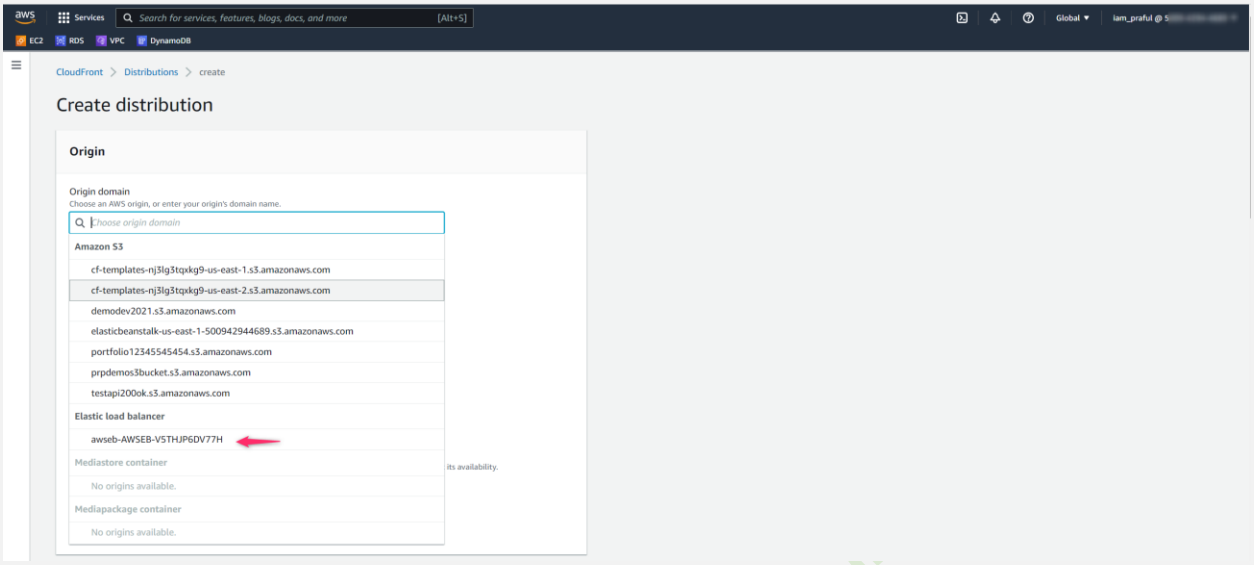


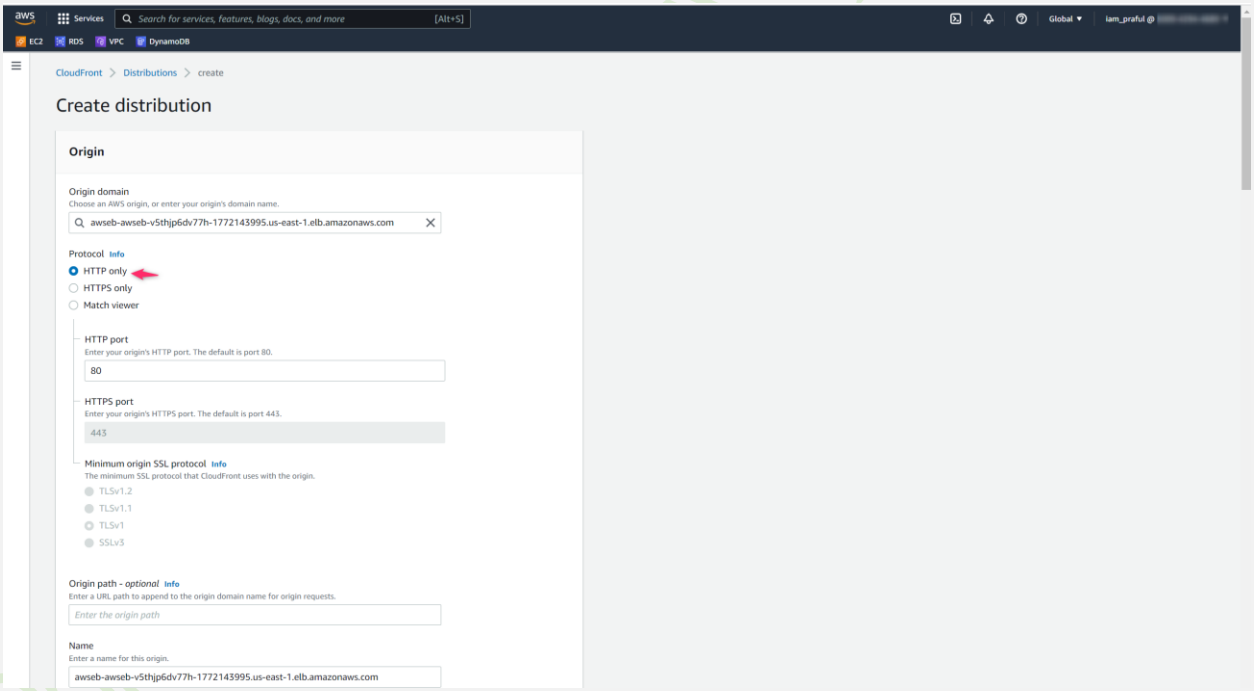
Implementation Phase 2 in Action: Create Cloudfront distribution

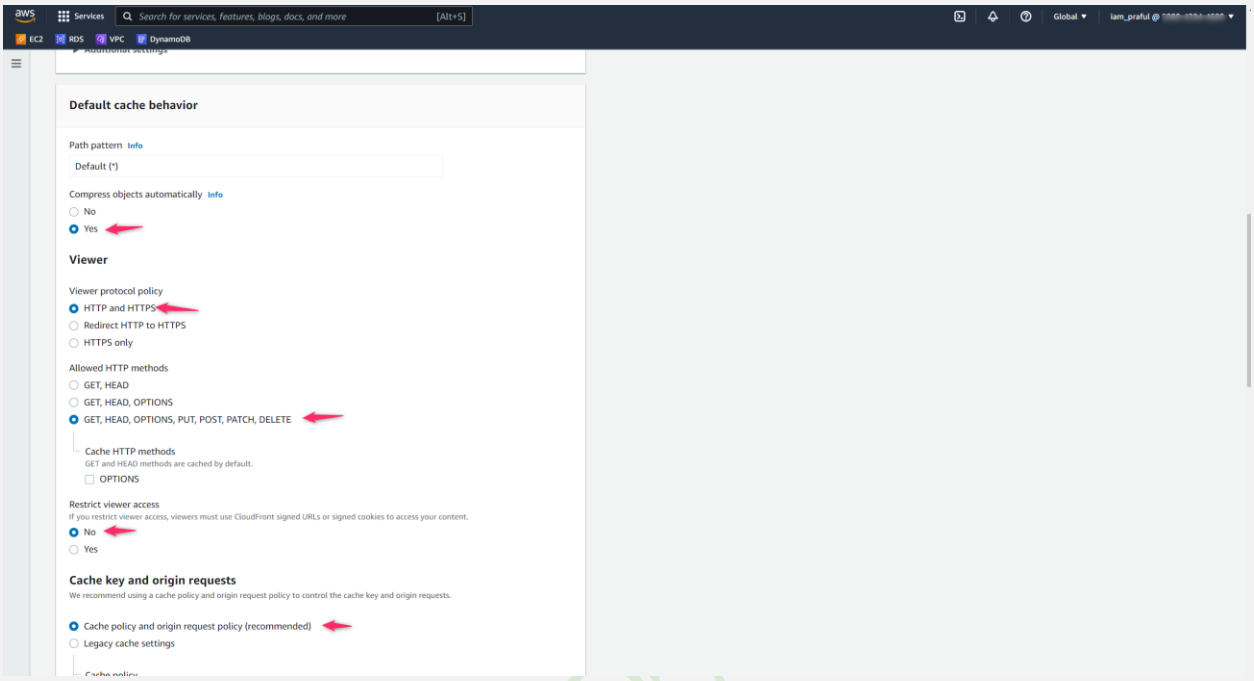


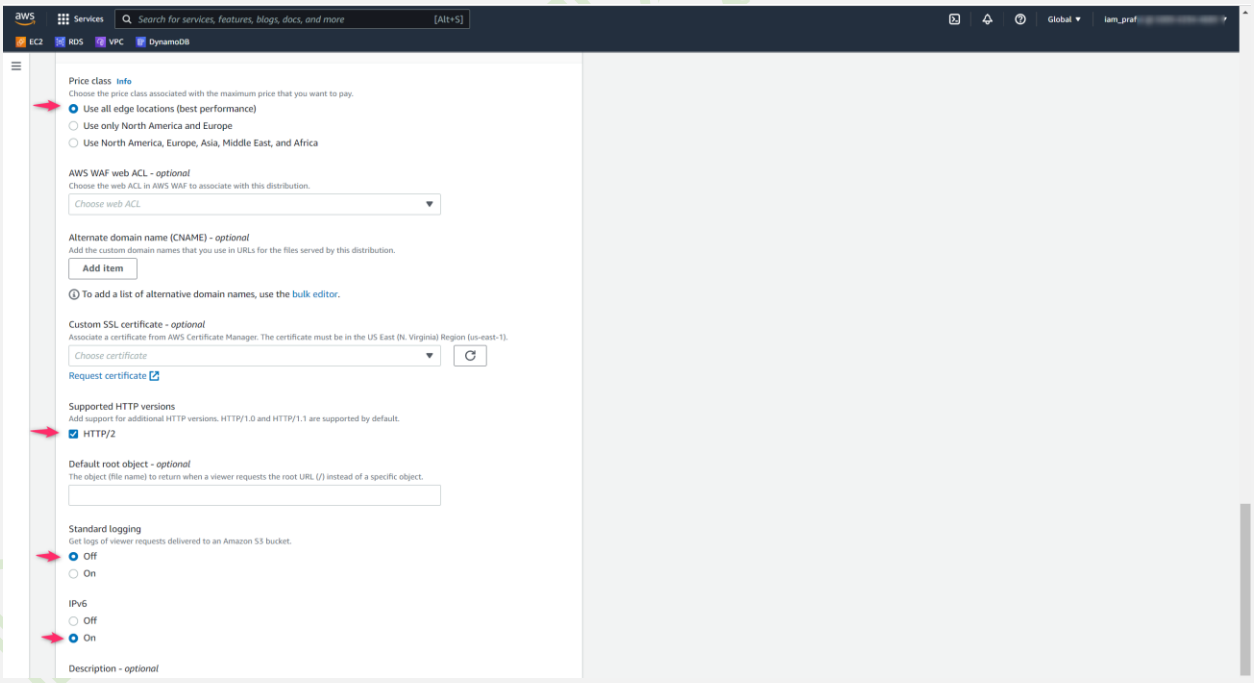
Search elastic load balancer

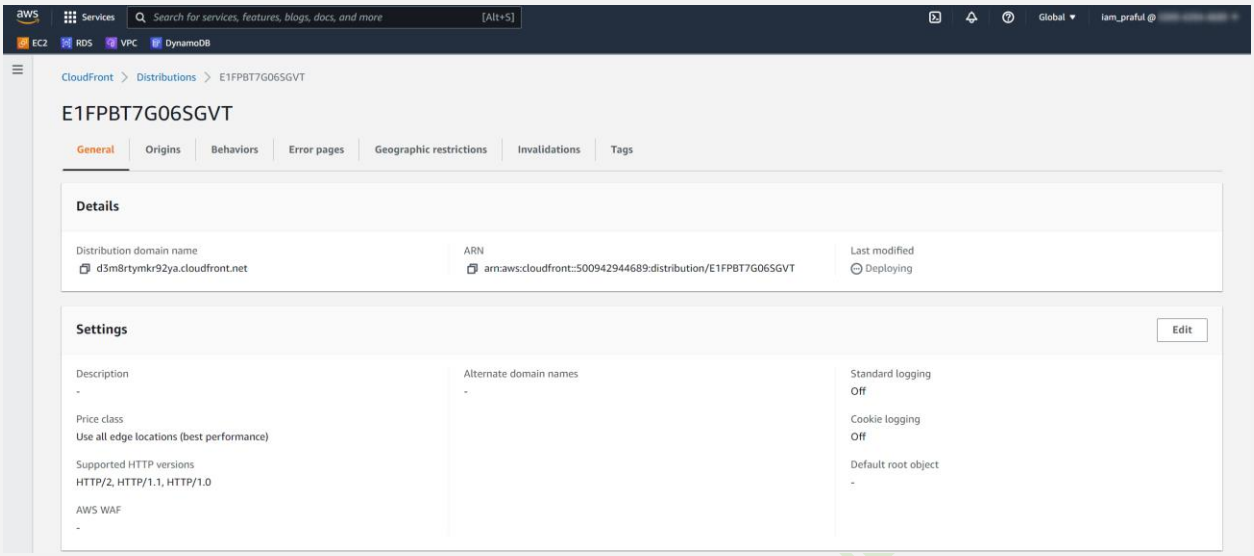



Services
Search for services, features, blogs, docs, and more
[Alt+S]
EC2
RDS
VPC
DynamoDB
Global
CloudFront > Distributions > create
Create distribution
Origin
Origin domain
Choose an AWS origin, or enter your origin's domain name.
Choose origin domain
Amazon S3
cf-templates-nj3lg3tqxkg9-us-east-1.s3.amazonaws.com
cf-templates-nj3lg3tqxkg9-us-east-2.s3.amazonaws.com
demodev2021.s3.amazonaws.com
elasticbeanstalk-us-east-1-500942944689.s3.amazonaws.com
portfolio12345545454.s3.amazonaws.com
prpdemos3bucket.s3.amazonaws.com
testapi200ok.s3.amazonaws.com
Elastic load balancer
awseb-AWSEB-V5THJP6DV77H
Mediastore container
No origins available.
Mediapackage container
No origins available.
its availability.

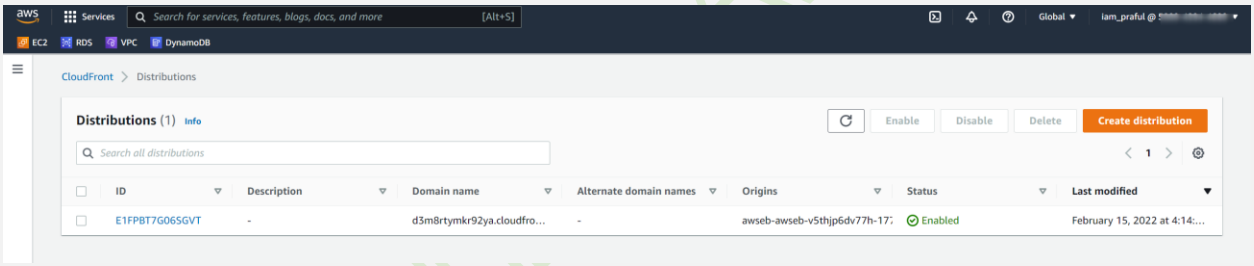

Services
Search for services, features, blogs, docs, and more
[Alt+S]
EC2
RDS
VPC
DynamoDB
Global
CloudFront > Distributions > create
Create distribution
Origin
Origin domain
Choose an AWS origin, or enter your origin's domain name.
awseb-awseb-v5thjp6dv77h-1772143995.us-east-1.elb.amazonaws.com
Protocol Info
HTTP only
HTTPS only
Match viewer
HTTP port
Enter your origin's HTTP port. The default is port 80.
80
HTTPS port
Enter your origin's HTTPS port. The default is port 443.
443
Minimum origin SSL protocol Info
The minimum SSL protocol that CloudFront uses with the origin.
TLSv1.2
TLSv1.1
TLSv1
SSLv3
Origin path - optional Info
Enter a URL path to append to the origin domain name for origin requests.
Enter the origin path
Name
Enter a name for this origin.
awseb-awseb-v5thjp6dv77h-1772143995.us-east-1.elb.amazonaws.com


Default cache behavior
Path pattern Info
Default (*)
Compress objects automatically Info
No
Yes
Viewer
Viewer protocol policy
HTTP and HTTPS
Redirect HTTP to HTTPS
HTTPS only
Allowed HTTP methods
GET, HEAD
GET, HEAD, OPTIONS
GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE
Cache HTTP methods
GET and HEAD methods are cached by default.
OPTIONS
Restrict viewer access
If you restrict viewer access, viewers must use CloudFront signed URLs or signed cookies to access your content.
No
Yes
Cache key and origin requests
We recommend using a cache policy and origin request policy to control the cache key and origin requests.
Cache policy and origin request policy (recommended)
Legacy cache settings

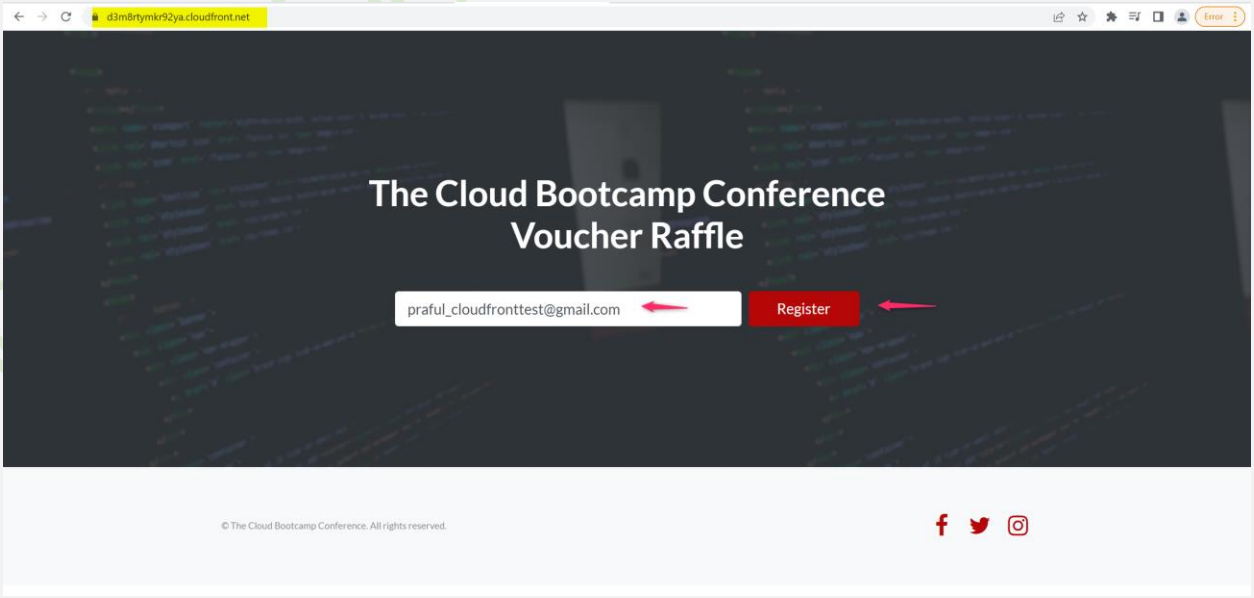

Price class Info
Choose the price class associated with the maximum price that you want to pay.
Use all edge locations (best performance)
Use only North America and Europe
Use North America, Europe, Asia, Middle East, and Africa
AWS WAF web ACL - optional
Choose the web ACL in AWS WAF to associate with this distribution.
Choose web ACL
Alternate domain name (CNAME) - optional
Add the custom domain names that you use in URLs for the files served by this distribution.
Add item
To add a list of alternative domain names, use the bulk editor.
Custom SSL certificate - optional
Associate a certificate from AWS Certificate Manager. The certificate must be in the US East (N. Virginia) Region (us-east-1).
Choose certificate
Request certificate
Supported HTTP versions
Add support for additional HTTP versions. HTTP/1.0 and HTTP/1.1 are supported by default.
HTTP/2
Default root object - optional
The object (file name) to return when a viewer requests the root URL (/) instead of a specific object.
Standard logging
Get logs of viewer requests delivered to an Amazon S3 bucket.
Off
On
IPv6
Off
On
Description - optional

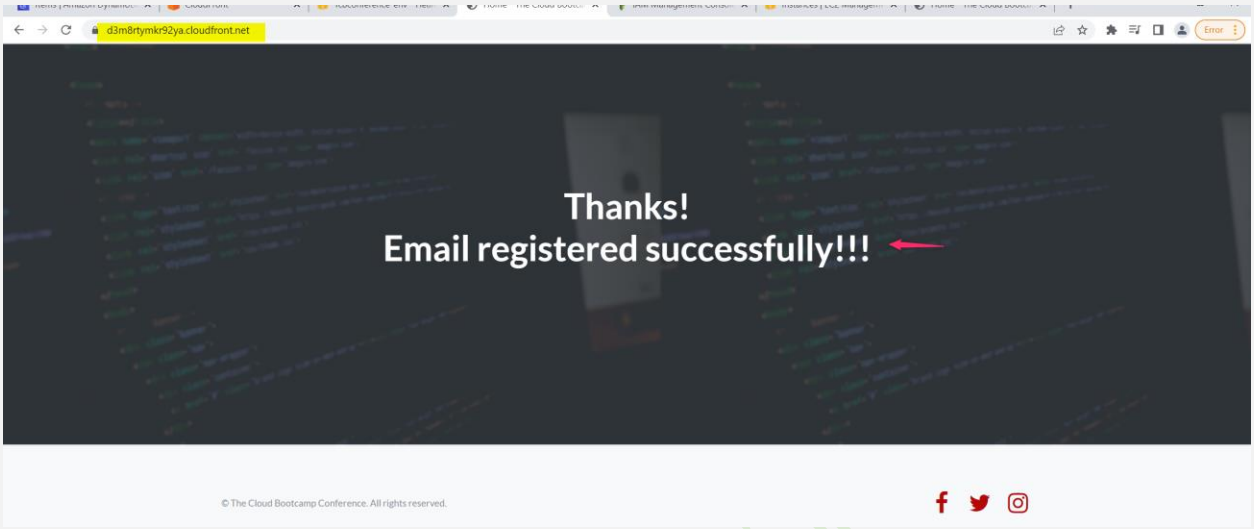


Cloudfront deployed



Verify that cloudfront URL is able to access and enter some data





- **Implementation Phase 3 in Action : Perform load test and verify application auto scale up**

Steps to run during the "dry-run & cutover":

Perform load testing.

- Install the stress tool to perform the load testing

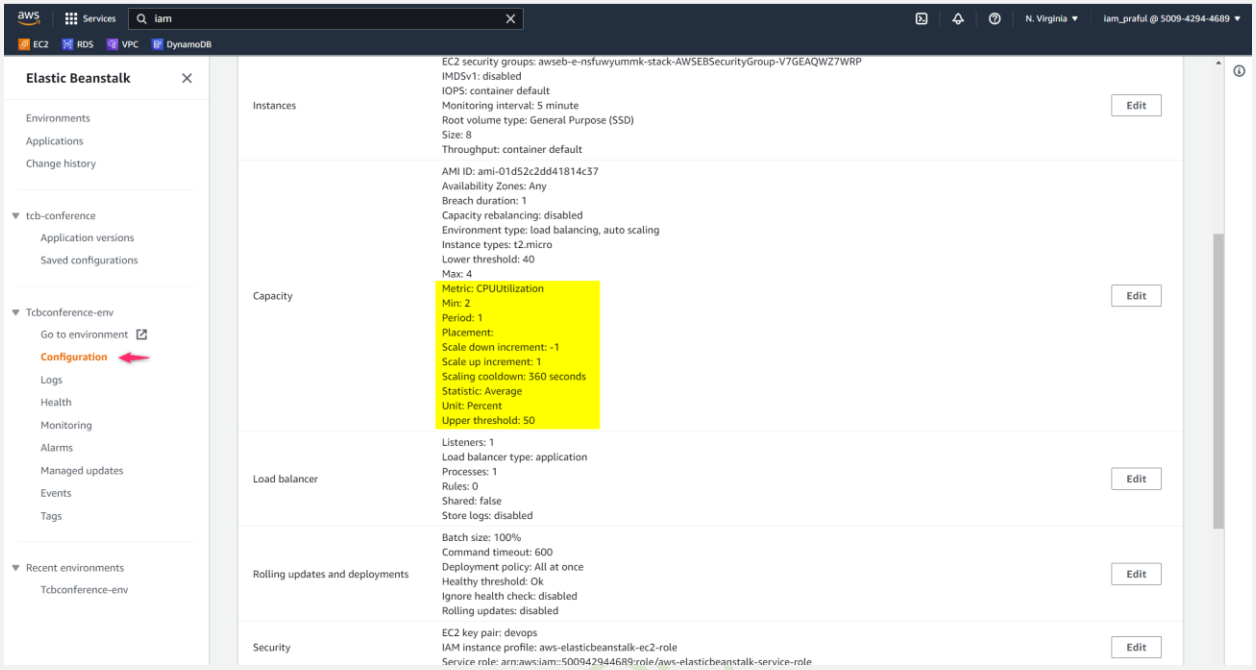
sudo amazon-linux-extras install epel -y
sudo yum install stress -y

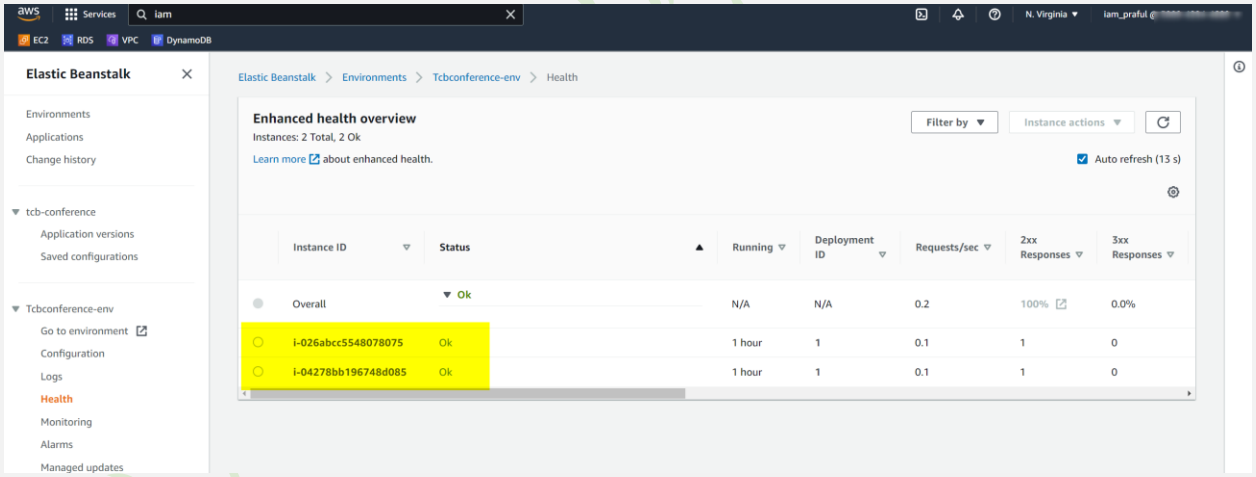
stress -c 4

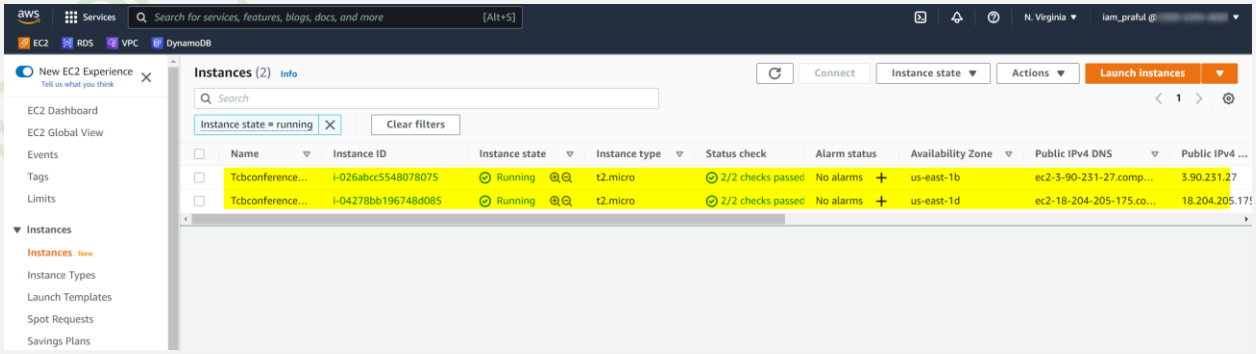
- Explore the resources created by the AWS Elastic Beanstalk (EC2, ELB, Auto Scaling Group, Cloud Watch resources) and also the auto scaling process

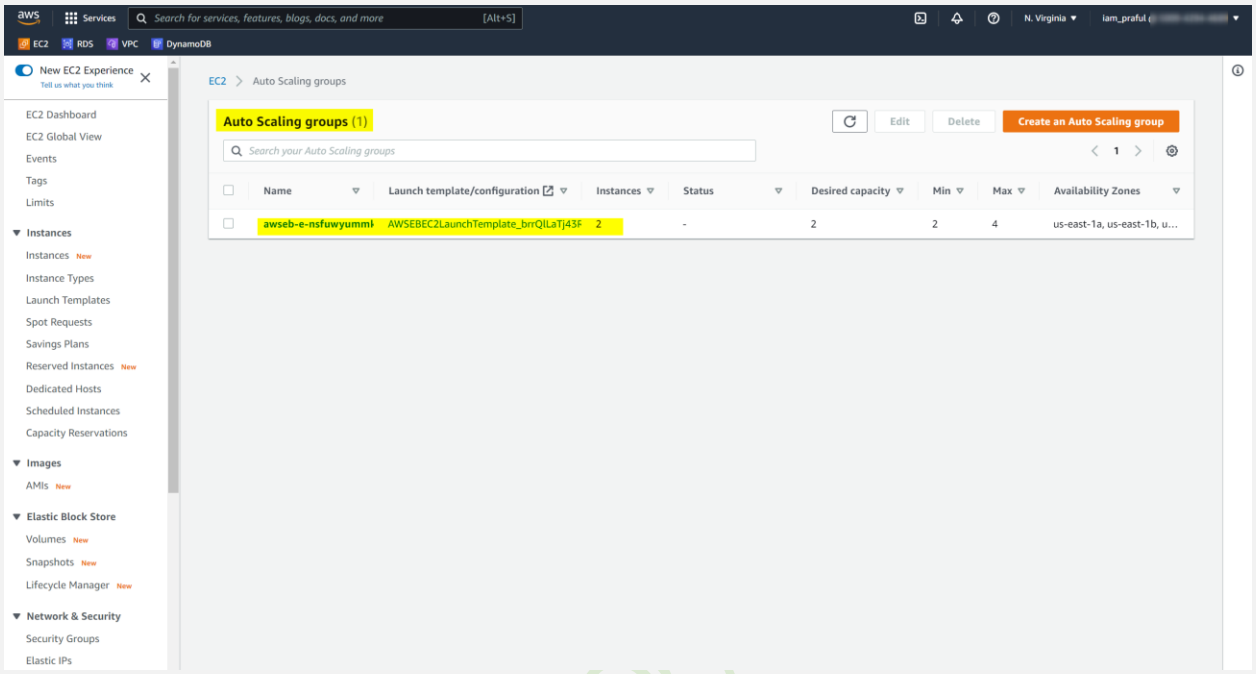
- Once you finish exploring it, please remove the Elastic Beanstalk application, Elastic Beanstalk environment , disable and delete the CloudFront distribution and finally delete the DynamoDB users table

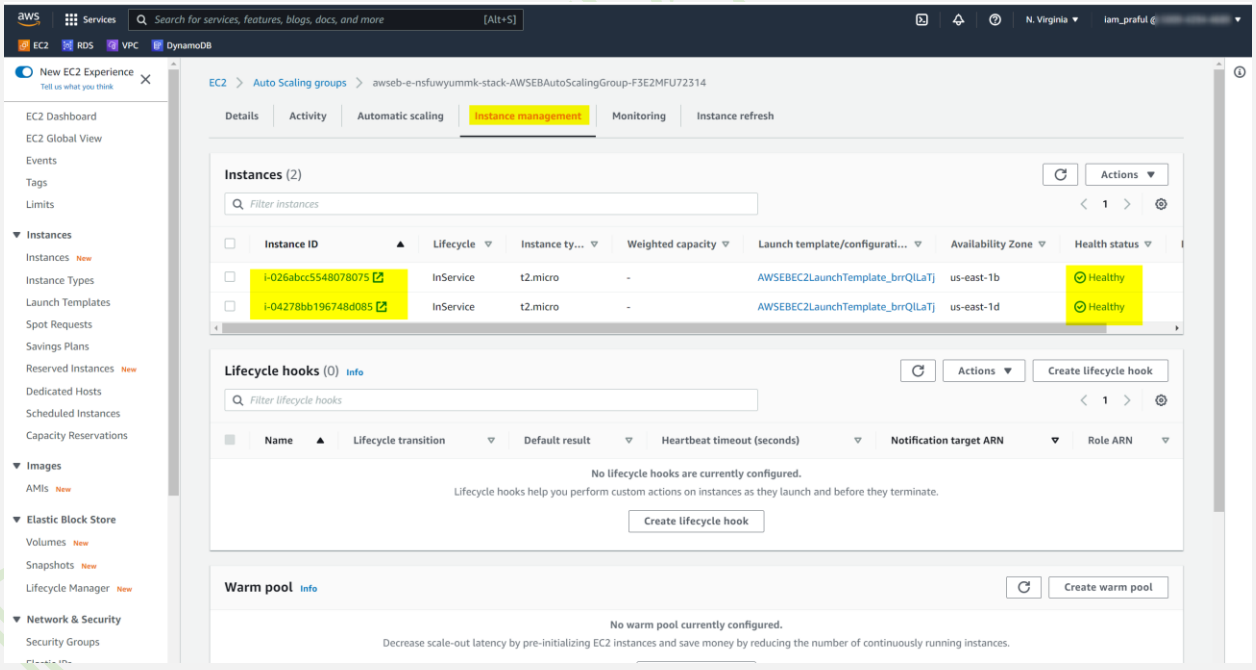
Situation before scaling and load test


Elastic Beanstalk
Environments
Applications
Change history
tcb-conference
Application versions
Saved configurations
Tcbconference-env
Go to environment
Configuration
Logs
Health
Monitoring
Alarms
Managed updates
Events
Tags
Recent environments
Tcbconference-env
Instances
EC2 security groups: awseb-e-nsfuwyummk-stack-AWSEBSecurityGroup-V7GEAQWZ7WRP
IMDSv1: disabled
IOPS: container default
Monitoring interval: 5 minute
Root volume type: General Purpose (SSD)
Size: 8
Throughput: container default
Edit
Capacity
AMI ID: ami-01d52c2dd41814c37
Availability Zones: Any
Breach duration: 1
Capacity rebalancing: disabled
Environment type: load balancing, auto scaling
Instance types: t2.micro
Lower threshold: 40
Max: 4
Metric: CPUUtilization
Min: 2
Period: 1
Placement:
Scale down increment: -1
Scale up increment: 1
Scaling cooldown: 360 seconds
Statistic: Average
Unit: Percent
Upper threshold: 50
Edit
Load balancer
Listeners: 1
Load balancer type: application
Processes: 1
Rules: 0
Shared: false
Store logs: disabled
Edit
Rolling updates and deployments
Batch size: 100%
Command timeout: 600
Deployment policy: All at once
Healthy threshold: Ok
Ignore health check: disabled
Rolling updates: disabled
Edit
Security
EC2 key pair: devops
IAM instance profile: aws-elasticbeanstalk-ec2-role
Edit

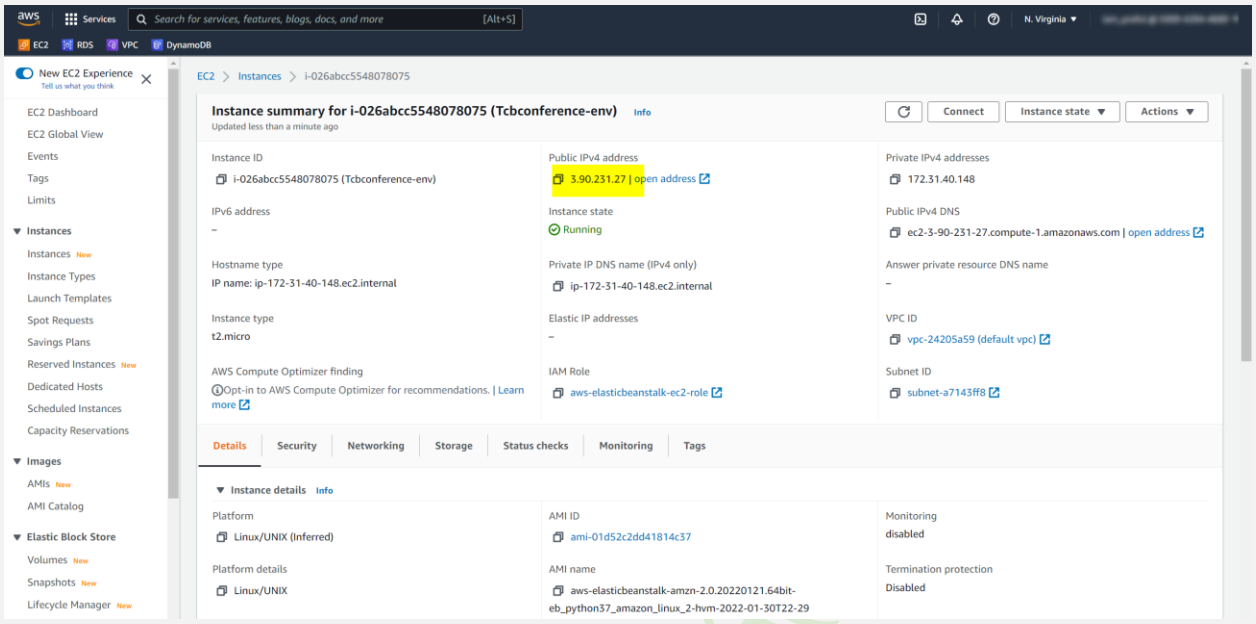

Elastic Beanstalk
Environments
Applications
Change history
tcb-conference
Application versions
Saved configurations
Tcbconference-env
Go to environment
Configuration
Logs
Health
Monitoring
Alarms
Managed updates
Elastic Beanstalk > Environments > Tcbconference-env > Health
Enhanced health overview
Instances: 2 Total, 2 Ok
Learn more about enhanced health.
Filter by
Instance actions
Auto refresh (13 s)
Instance ID
Status
Running
Deployment ID
Requests/sec
2xx Responses
3xx Responses
Overall
Ok
N/A
N/A
0.2
100%
0.0%
i-026abcc5548078075
Ok
1 hour
1
0.1
1
0
i-04278bb196748d085
Ok
1 hour
1
0.1
1
0


Search for services, features, blogs, docs, and more [Alt+S]
New EC2 Experience
Tell us what you think
EC2 Dashboard
EC2 Global View
Events
Tags
Limits
Instances
Instances New
Instance Types
Launch Templates
Spot Requests
Savings Plans
Instances (2) Info
Connect
Instance state
Actions
Launch instances
Search
Instance state = running
Clear filters
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
Public IPv4 ...
Tcbconference...
i-026abcc5548078075
Running
t2.micro
2/2 checks passed
No alarms
us-east-1b
ec2-3-90-231-27.comp...
3.90.231.27
Tcbconference...
i-04278bb196748d085
Running
t2.micro
2/2 checks passed
No alarms
us-east-1d
ec2-18-204-205-175.co...
18.204.205.175

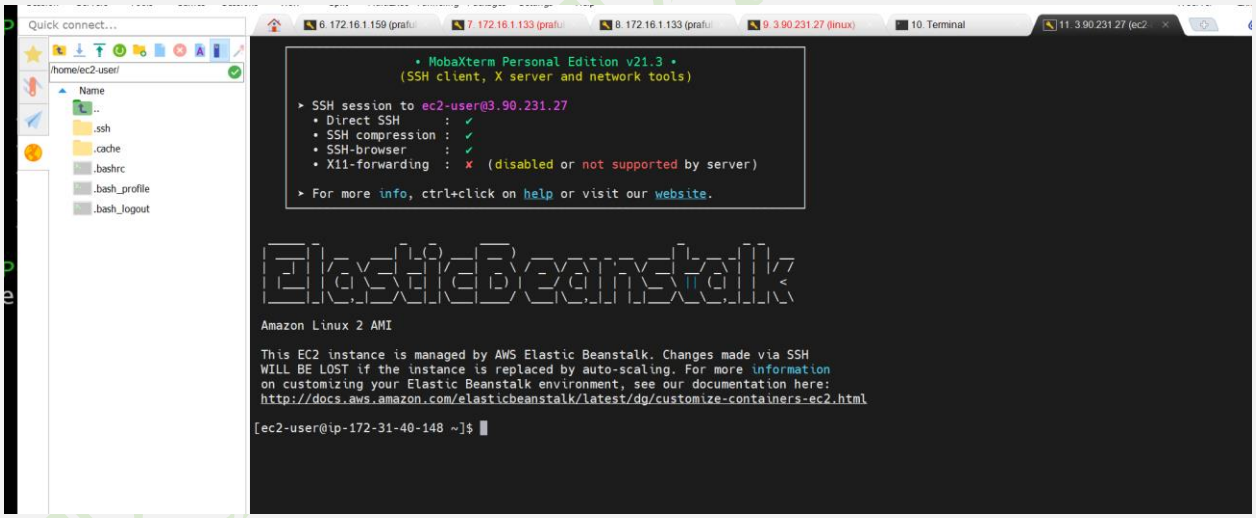




Go to one of EC2 instance and access one of public ip



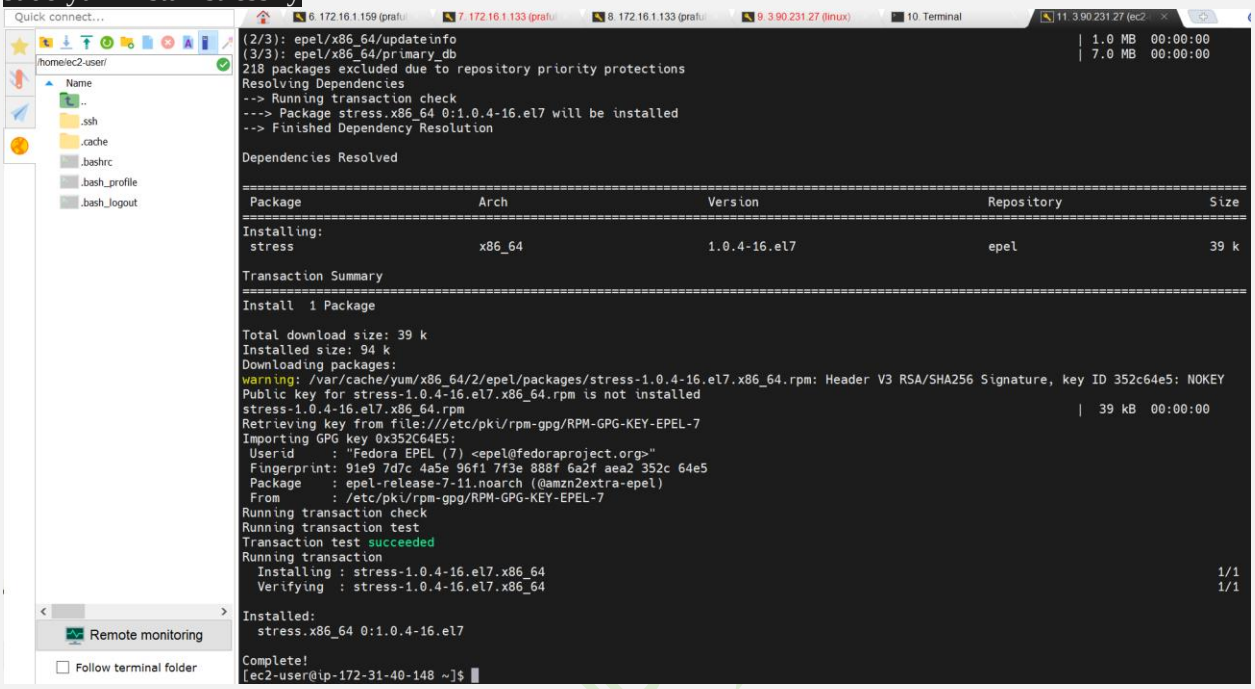
Login to EC2 server



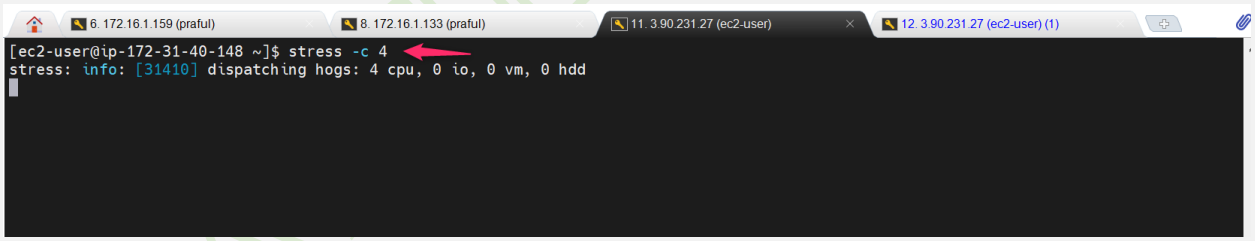
- Install the stress tool to perform the load testing

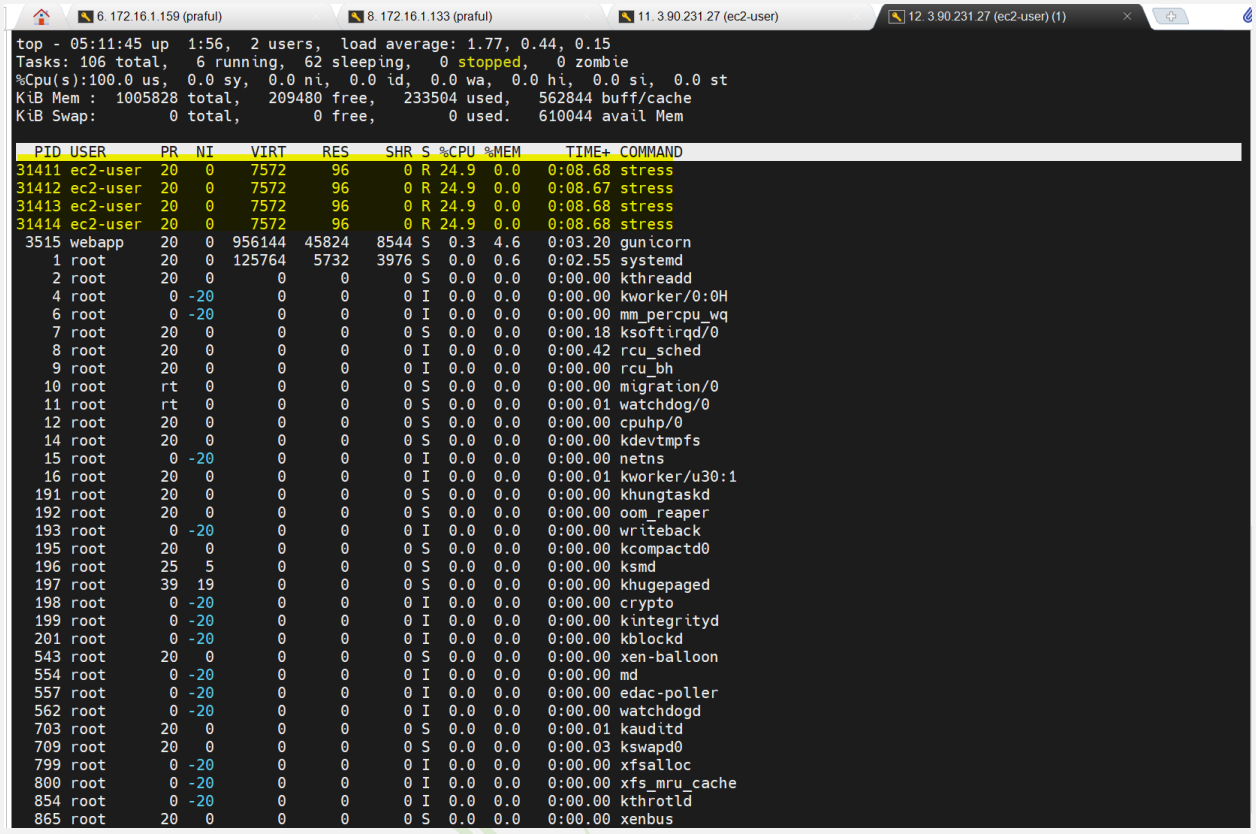
sudo amazon-linux-extras install epel -y

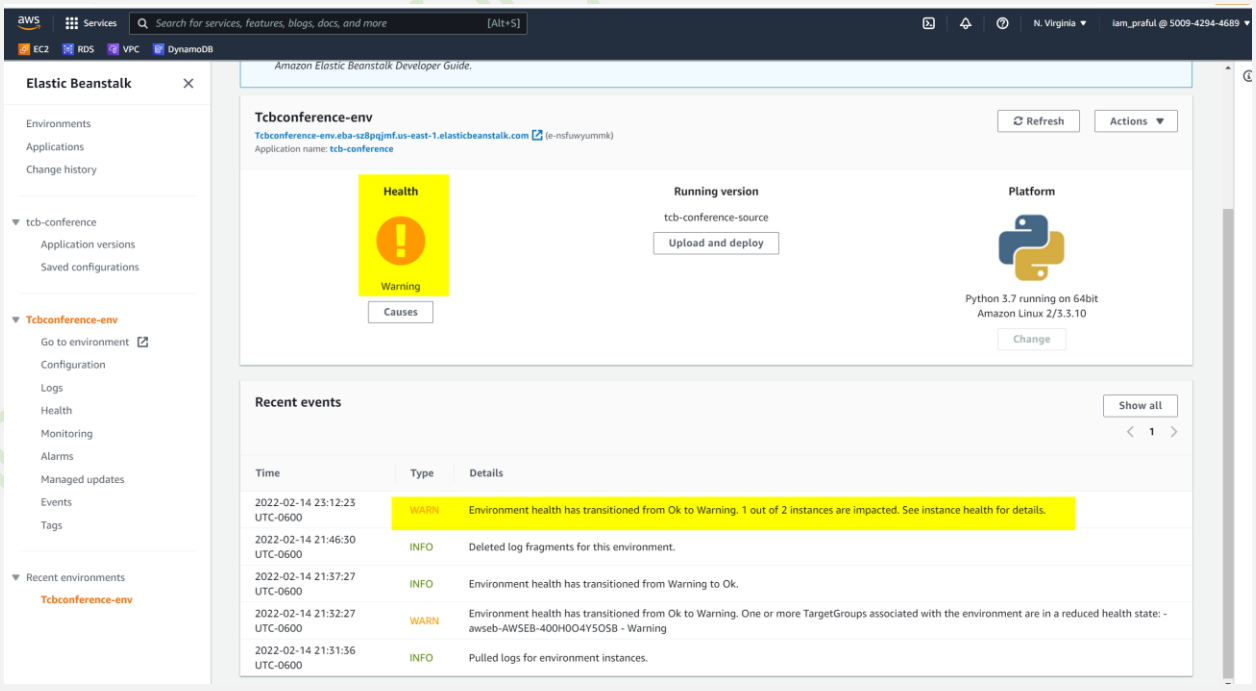
sudo yum install stress -y



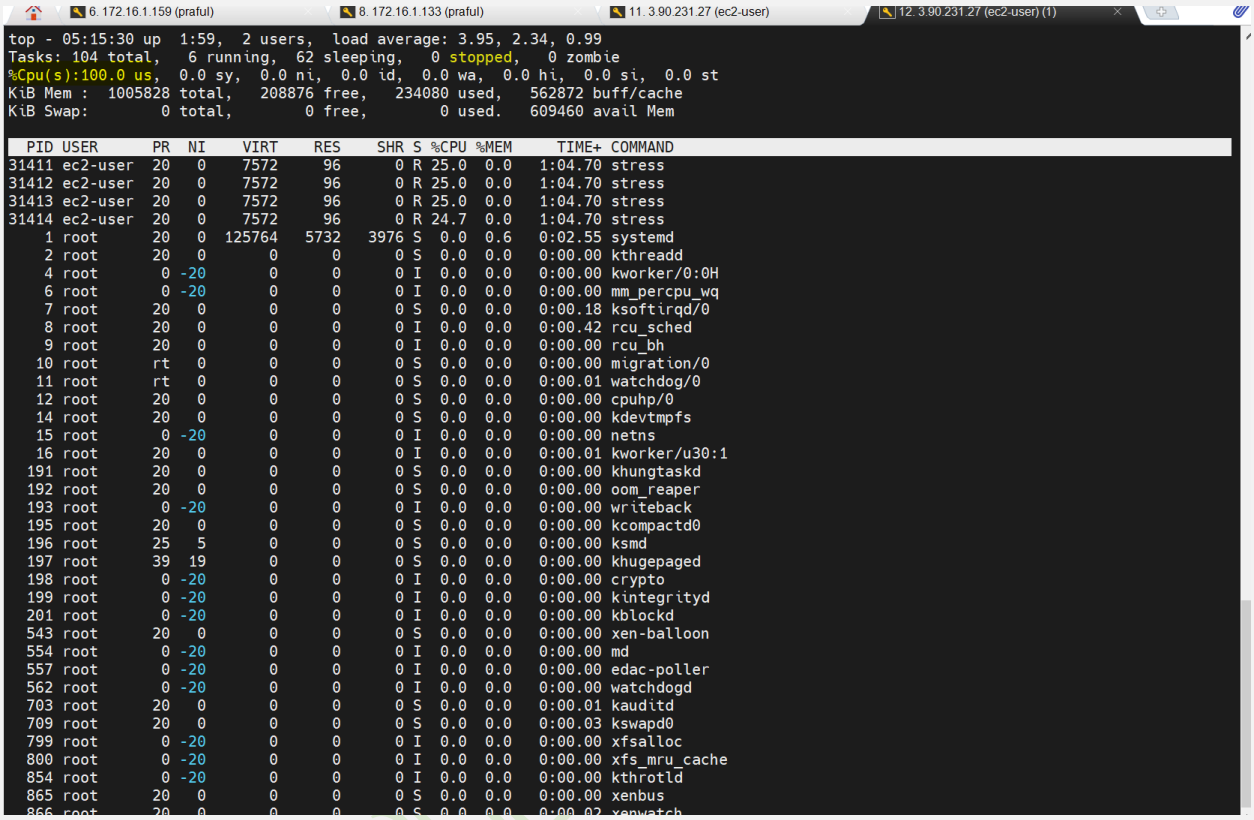
Increase load on one of the EC2 server using stress command
stress -c 4



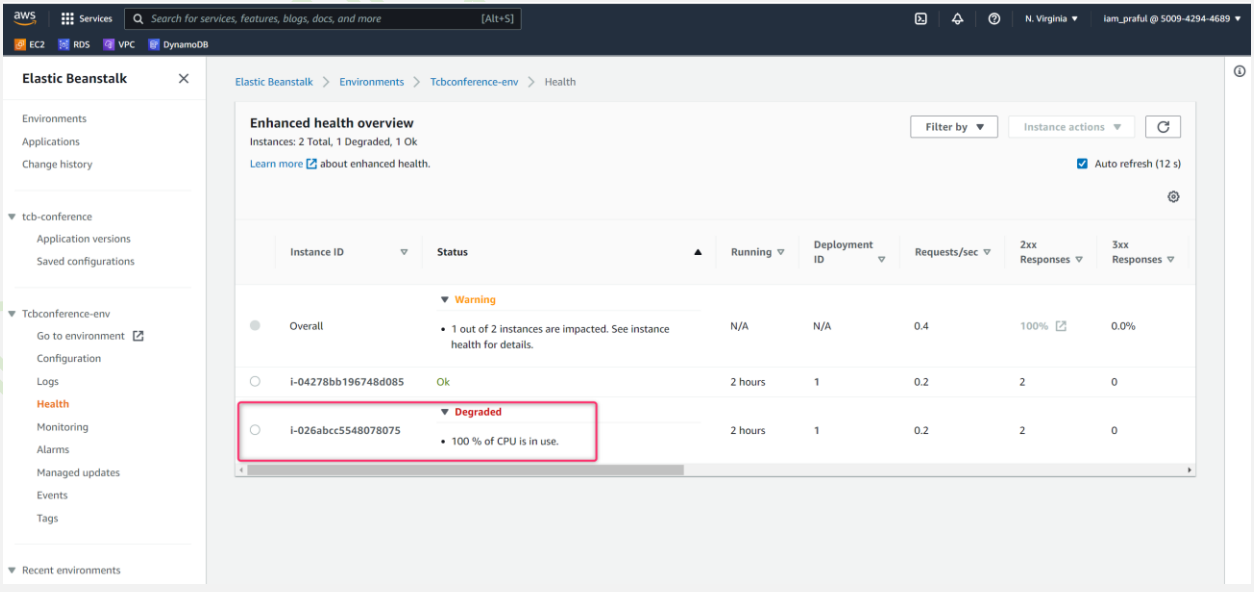

6. 172.16.1.159 (praful)
8. 172.16.1.133 (praful)
11. 3.90.231.27 (ec2-user)
12. 3.90.231.27 (ec2-user) (1)
top - 05:11:45 up 1:56, 2 users, load average: 1.77, 0.44, 0.15
Tasks: 106 total, 6 running, 62 sleeping, 0 stopped, 0 zombie
%Cpu(s):100.0 us, 0.0 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 1005828 total, 209480 free, 233504 used, 562844 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 610044 avail Mem
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
31411 ec2-user 20 0 7572 96 0 R 24.9 0.0 0:08.68 stress
31412 ec2-user 20 0 7572 96 0 R 24.9 0.0 0:08.67 stress
31413 ec2-user 20 0 7572 96 0 R 24.9 0.0 0:08.68 stress
31414 ec2-user 20 0 7572 96 0 R 24.9 0.0 0:08.68 stress
3515 webapp 20 0 956144 45824 8544 S 0.3 4.6 0:03.20 gunicorn
1 root 20 0 125764 5732 3976 S 0.0 0.6 0:02.55 systemd
2 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kthreadd
4 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/0:0H
6 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 mm_percpu_wq
7 root 20 0 0 0 0 S 0.0 0.0 0:00.18 ksoftirqd/0
8 root 20 0 0 0 0 I 0.0 0.0 0:00.42 rcu_sched
9 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_bh
10 root rt 0 0 0 0 S 0.0 0.0 0:00.00 migration/0
11 root rt 0 0 0 0 S 0.0 0.0 0:00.01 watchdog/0
12 root 20 0 0 0 0 S 0.0 0.0 0:00.00 cpuhp/0
14 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kdevtmpfs
15 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 netns
16 root 20 0 0 0 0 I 0.0 0.0 0:00.01 kworker/u30:1
191 root 20 0 0 0 0 S 0.0 0.0 0:00.00 khungtaskd
192 root 20 0 0 0 0 S 0.0 0.0 0:00.00 oom_reaper
193 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 writeback
195 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kcompactd0
196 root 25 5 0 0 0 S 0.0 0.0 0:00.00 ksmd
197 root 39 19 0 0 0 S 0.0 0.0 0:00.00 khugepaged
198 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 crypto
199 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kintegrityd
201 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kblockd
543 root 20 0 0 0 0 S 0.0 0.0 0:00.00 xen-balloon
554 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 md
557 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 edac-poller
562 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 watchdogd
703 root 20 0 0 0 0 S 0.0 0.0 0:00.01 kauditd
709 root 20 0 0 0 0 S 0.0 0.0 0:00.03 kswapd0
799 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 xfsalloc
800 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 xfs_mru_cache
854 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kthrotld
865 root 20 0 0 0 0 S 0.0 0.0 0:00.00 xenbus

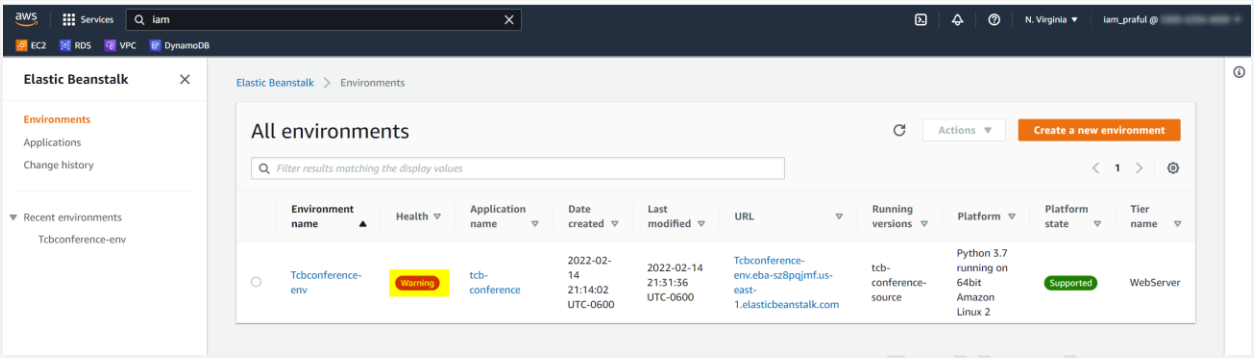

Services
Search for services, features, blogs, docs, and more
[Alt+S]
N. Virginia
iam_praful @ 5009-4294-4689
EC2
RDS
VPC
DynamoDB
Elastic Beanstalk
Amazon Elastic Beanstalk Developer Guide.
Environments
Applications
Change history
tcb-conference
Application versions
Saved configurations
Tcbconference-env
Go to environment
Configuration
Logs
Health
Monitoring
Alarms
Managed updates
Events
Tags
Recent environments
Tcbconference-env
Tcbconference-env
Tcbconference-env.eba-sz8pqjmf.us-east-1.elasticbeanstalk.com (e-nsfuwyummk)
Application name: tcb-conference
Refresh
Actions
Health
Warning
Causes
Running version
tcb-conference-source
Upload and deploy
Platform
Python 3.7 running on 64bit
Amazon Linux 2/3.3.10
Change
Recent events
Show all
Time
Type
Details
2022-02-14 23:12:23 UTC-0600
WARN
Environment health has transitioned from Ok to Warning. 1 out of 2 instances are impacted. See instance health for details.
2022-02-14 21:46:30 UTC-0600
INFO
Deleted log fragments for this environment.
2022-02-14 21:37:27 UTC-0600
INFO
Environment health has transitioned from Warning to Ok.
2022-02-14 21:32:27 UTC-0600
WARN
Environment health has transitioned from Ok to Warning. One or more TargetGroups associated with the environment are in a reduced health state: - awseb-AWSEB-400H0O4Y5OSB - Warning
2022-02-14 21:31:36 UTC-0600
INFO
Pulled logs for environment instances.

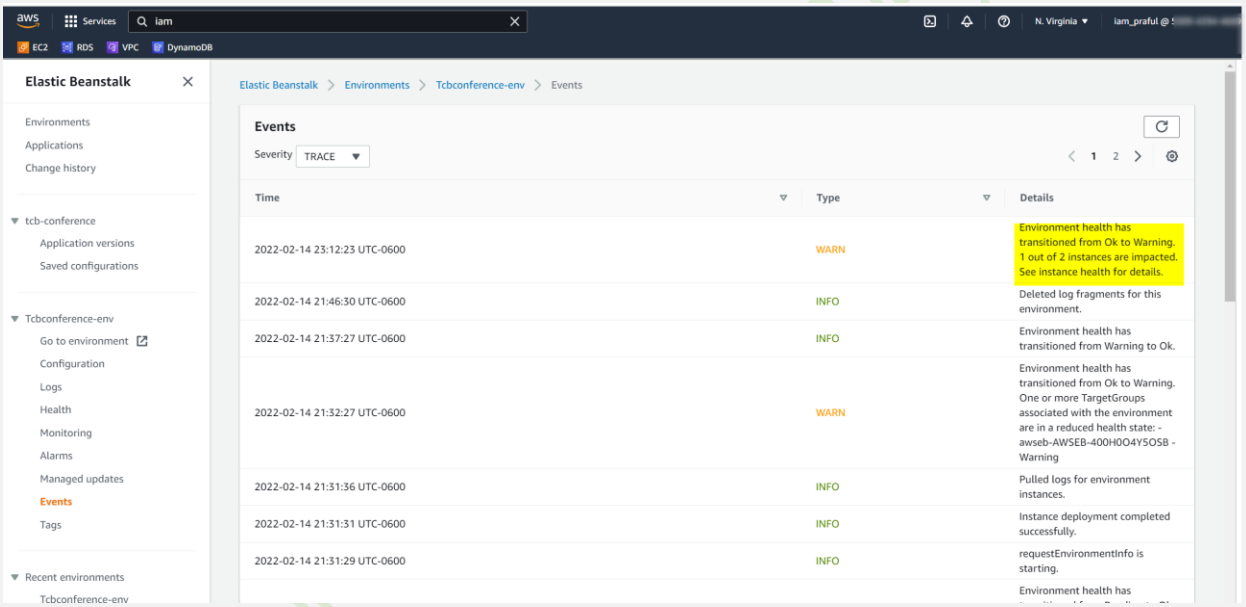
CPU has gone up to 100%



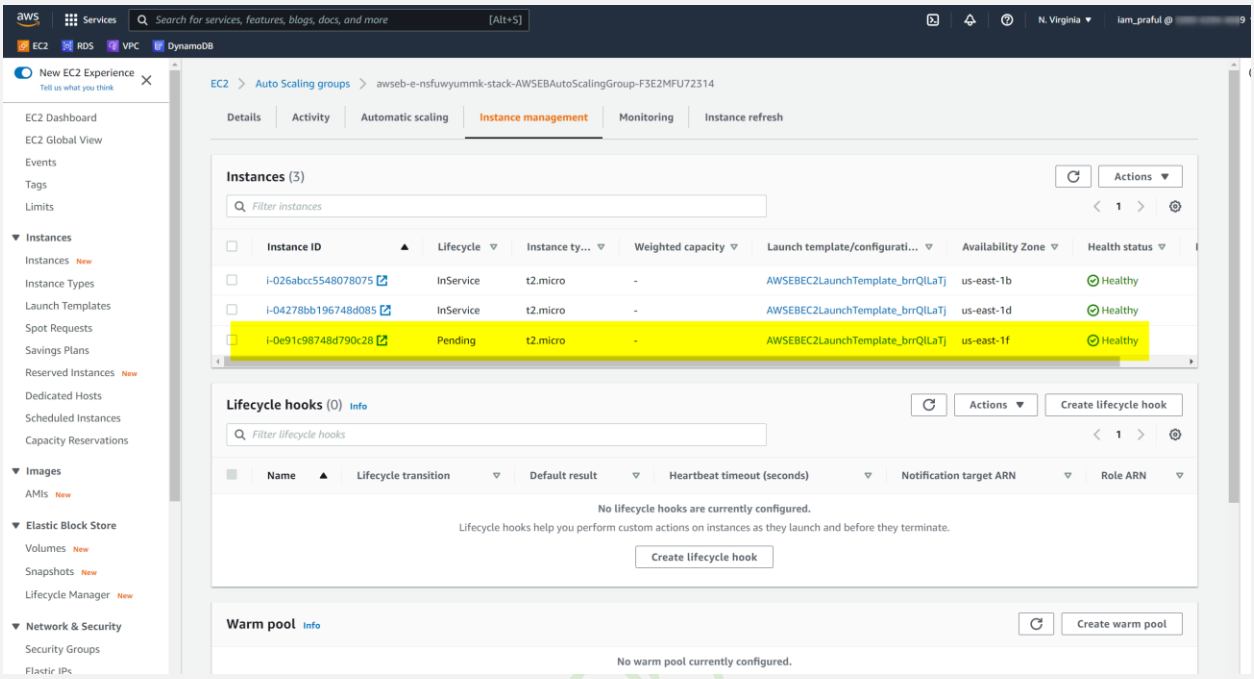
EC2 instance has considered as unhealthy and started to de register

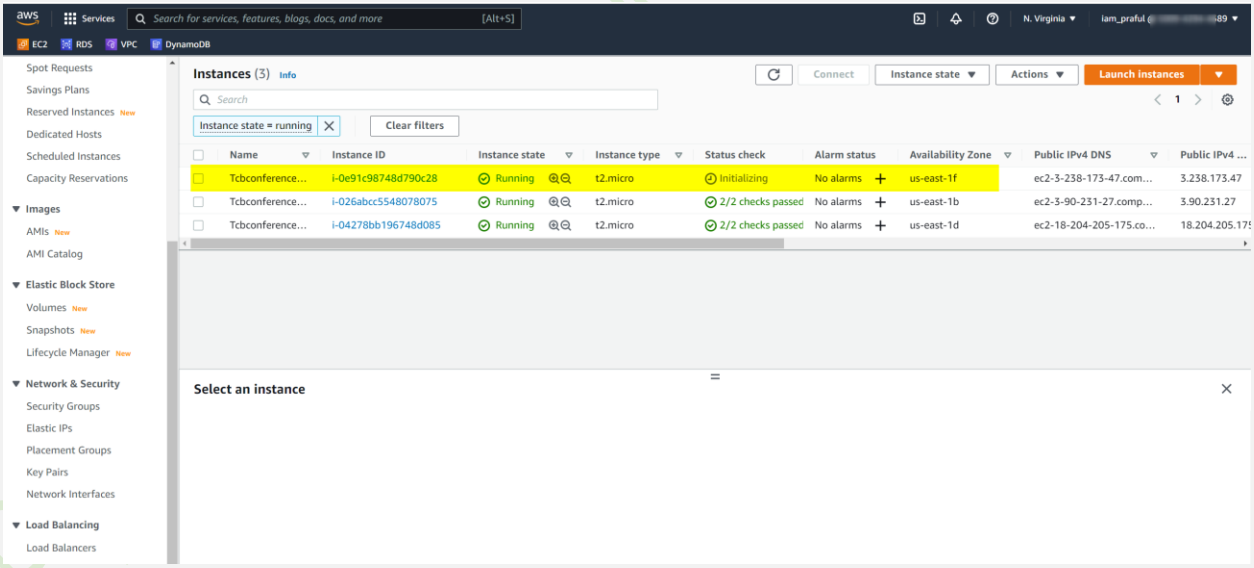


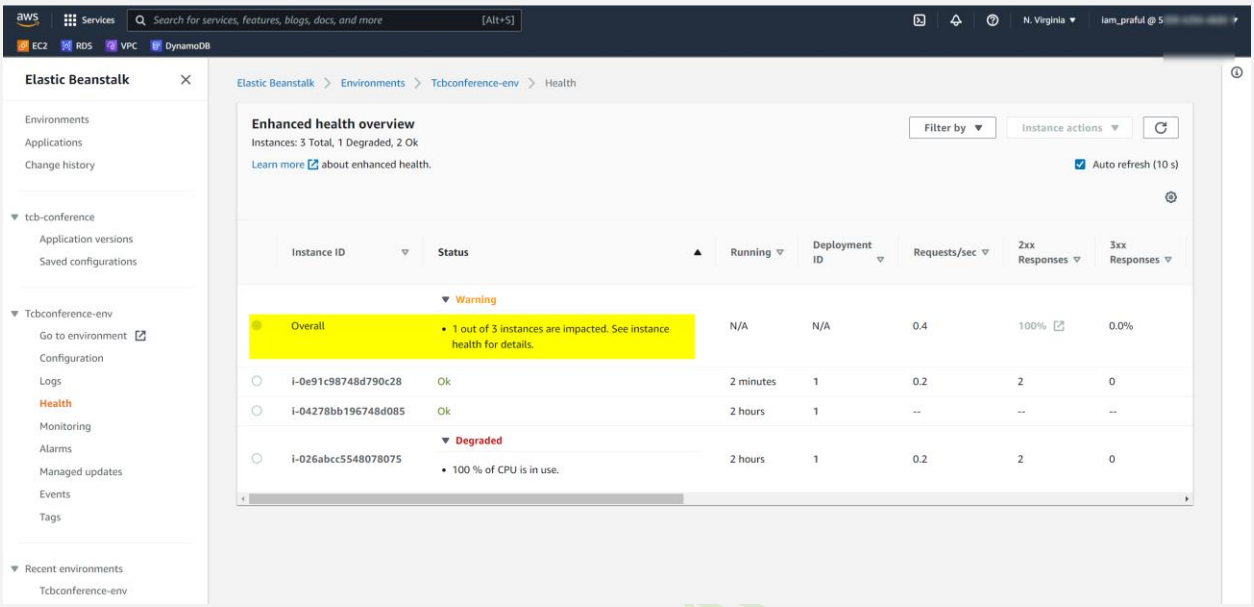




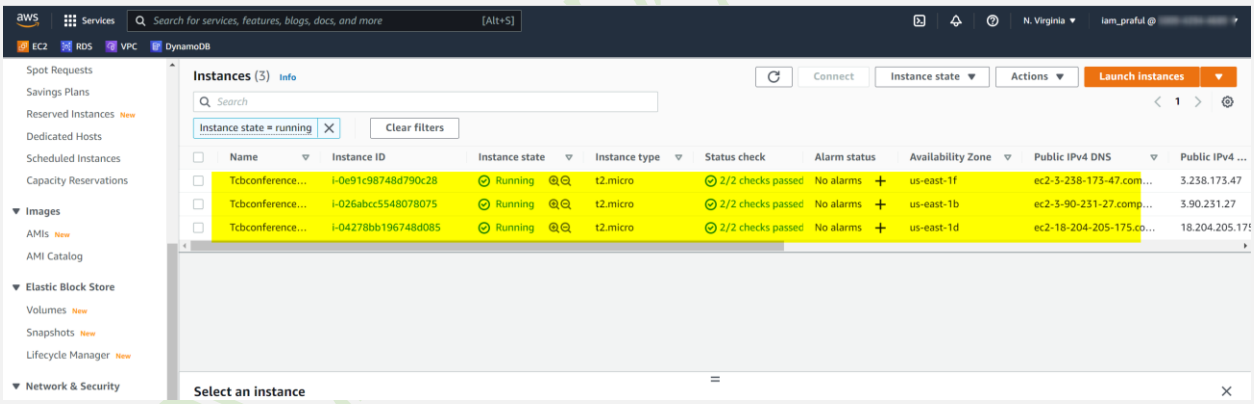
New EC 2 instance has started registering to auto scaling group and added to load balancer target

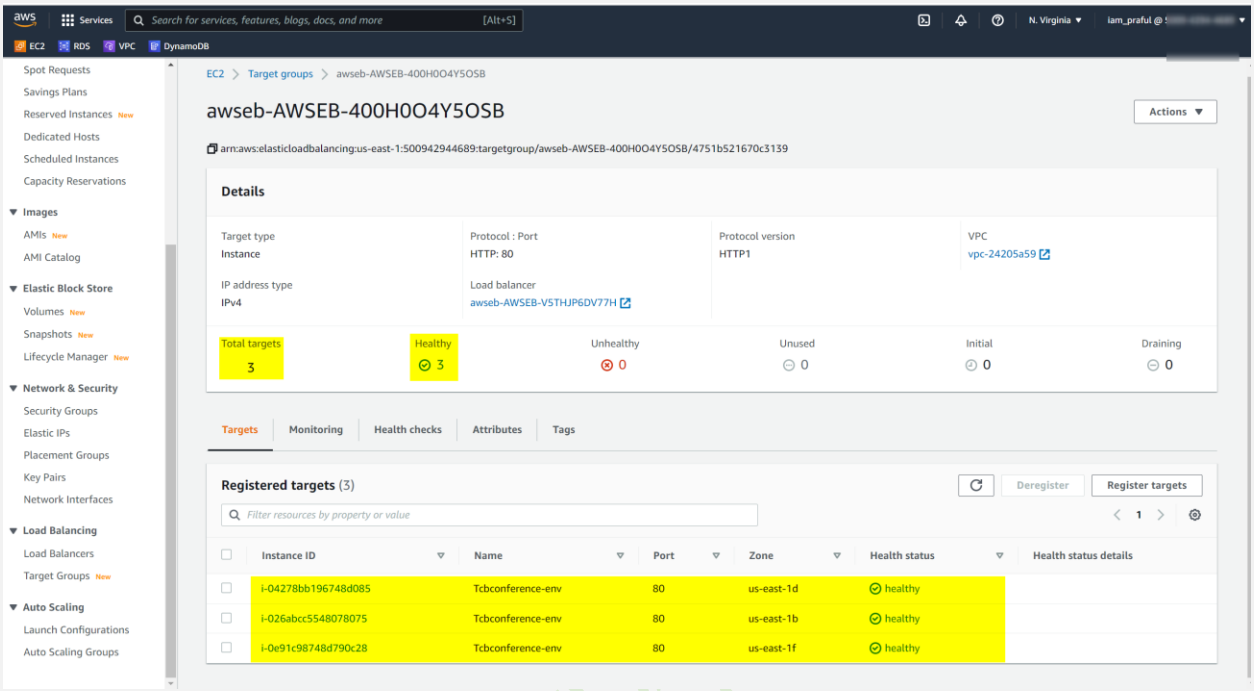

Services
Search for services, features, blogs, docs, and more [Alt+S]
N. Virginia
EC2 RDS VPC DynamoDB
New EC2 Experience
EC2 Dashboard
EC2 Global View
Events
Tags
Limits
Instances
Instances New
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances New
Dedicated Hosts
Scheduled Instances
Capacity Reservations
Images
AMIs New
Elastic Block Store
Volumes New
Snapshots New
Lifecycle Manager New
Network & Security
Security Groups
EC2 > Auto Scaling groups > awseb-e-nsfuwyummk-stack-AWSEBAutoScalingGroup-F3E2MFU72314
Details Activity Automatic scaling Instance management Monitoring Instance refresh
Instances (3)
Actions
Filter instances
Instance ID Lifecycle Instance ty... Weighted capacity Launch template/configurati... Availability Zone Health status
i-026abcc5548078075 InService t2.micro - AWSEBEC2LaunchTemplate_brrQlLaTj us-east-1b Healthy
i-04278bb196748d085 InService t2.micro - AWSEBEC2LaunchTemplate_brrQlLaTj us-east-1d Healthy
i-0e91c98748d790c28 Pending t2.micro - AWSEBEC2LaunchTemplate_brrQlLaTj us-east-1f Healthy
Lifecycle hooks (0) Info
Actions
Create lifecycle hook
Filter lifecycle hooks
Name Lifecycle transition Default result Heartbeat timeout (seconds) Notification target ARN Role ARN
No lifecycle hooks are currently configured.
Lifecycle hooks help you perform custom actions on instances as they launch and before they terminate.
Create lifecycle hook
Warm pool Info
Create warm pool
No warm pool currently configured.


Services
Search for services, features, blogs, docs, and more [Alt+S]
N. Virginia
EC2 RDS VPC DynamoDB
Spot Requests
Savings Plans
Reserved Instances New
Dedicated Hosts
Scheduled Instances
Capacity Reservations
Images
AMIs New
AMI Catalog
Elastic Block Store
Volumes New
Snapshots New
Lifecycle Manager New
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
Load Balancers
Instances (3) Info
Connect
Instance state
Actions
Launch instances
Search
Instance state = running
Clear filters
Name Instance ID Instance state Instance type Status check Alarm status Availability Zone Public IPv4 DNS Public IPv4 ...
Tcbconference... i-0e91c98748d790c28 Running t2.micro Initializing No alarms us-east-1f ec2-3-238-173-47.com... 3.238.173.47
Tcbconference... i-026abcc5548078075 Running t2.micro 2/2 checks passed No alarms us-east-1b ec2-3-90-231-27.comp... 3.90.231.27
Tcbconference... i-04278bb196748d085 Running t2.micro 2/2 checks passed No alarms us-east-1d ec2-18-204-205-175.co... 18.204.205.175
Select an instance

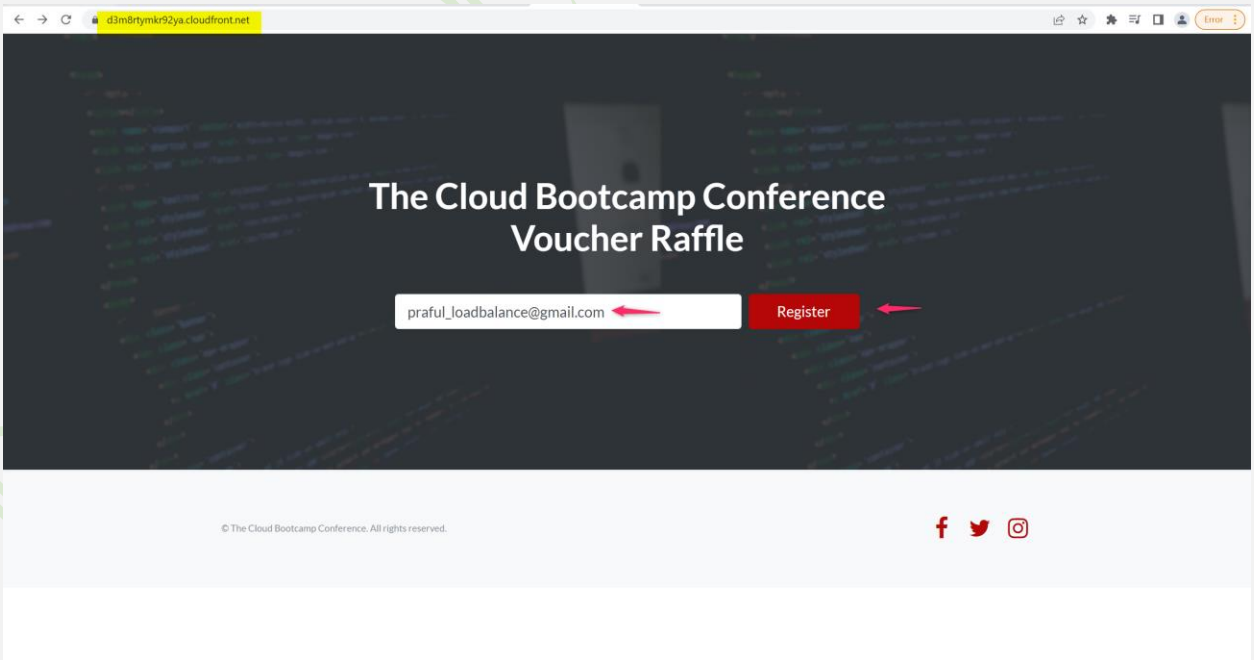


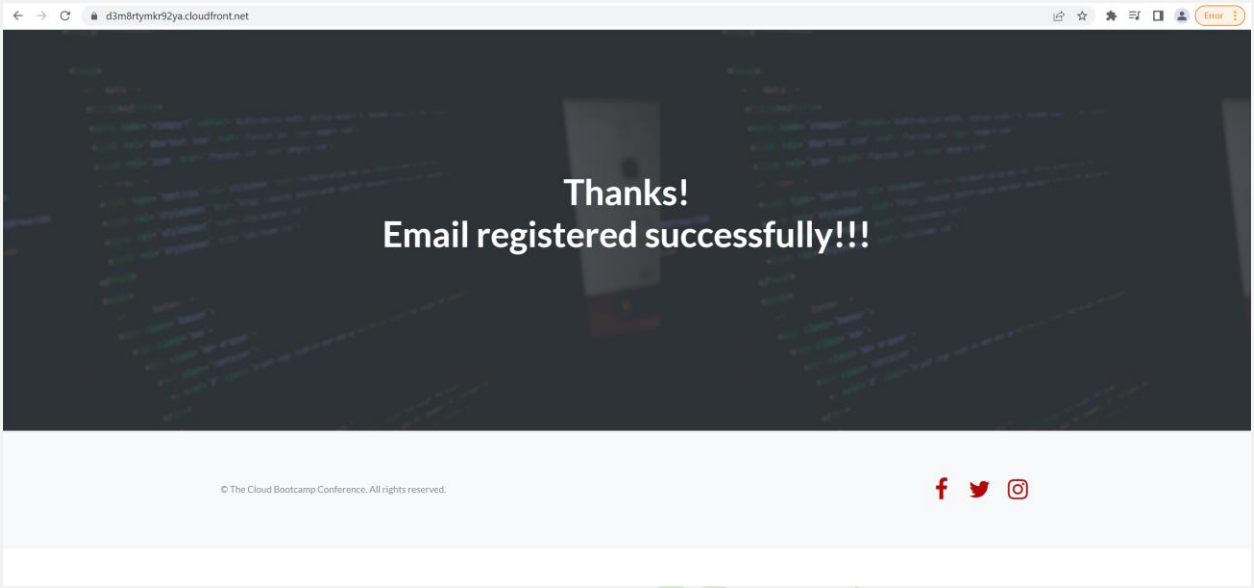
Now 3 EC2 instance has up and running due to load on web application



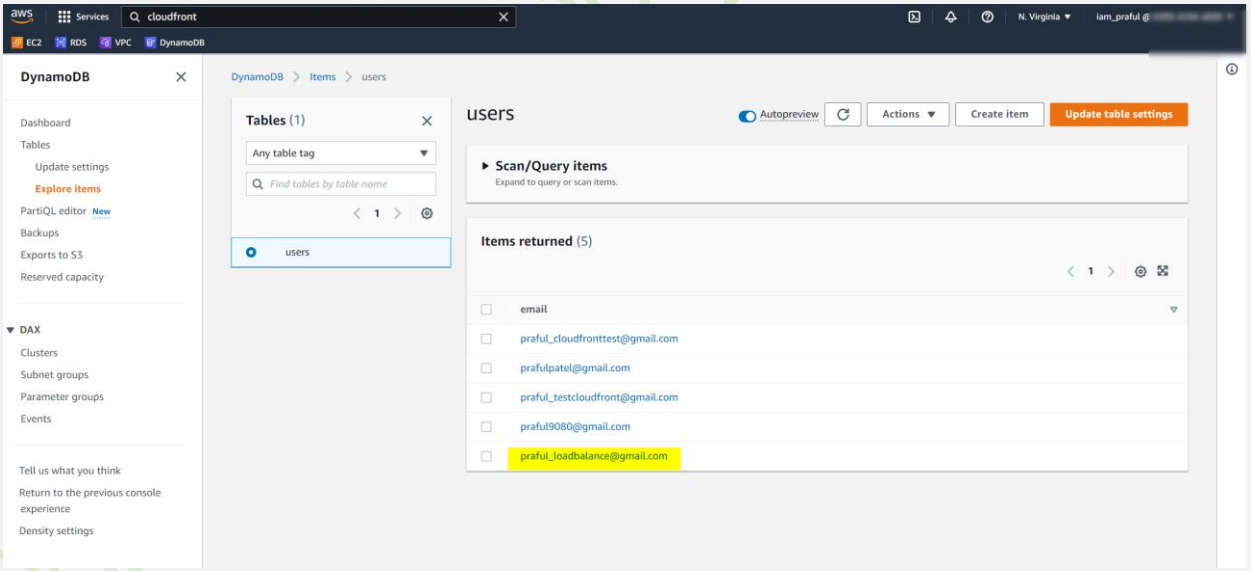


Let's verify through cloudfront web app url if the load balancer is working correctly

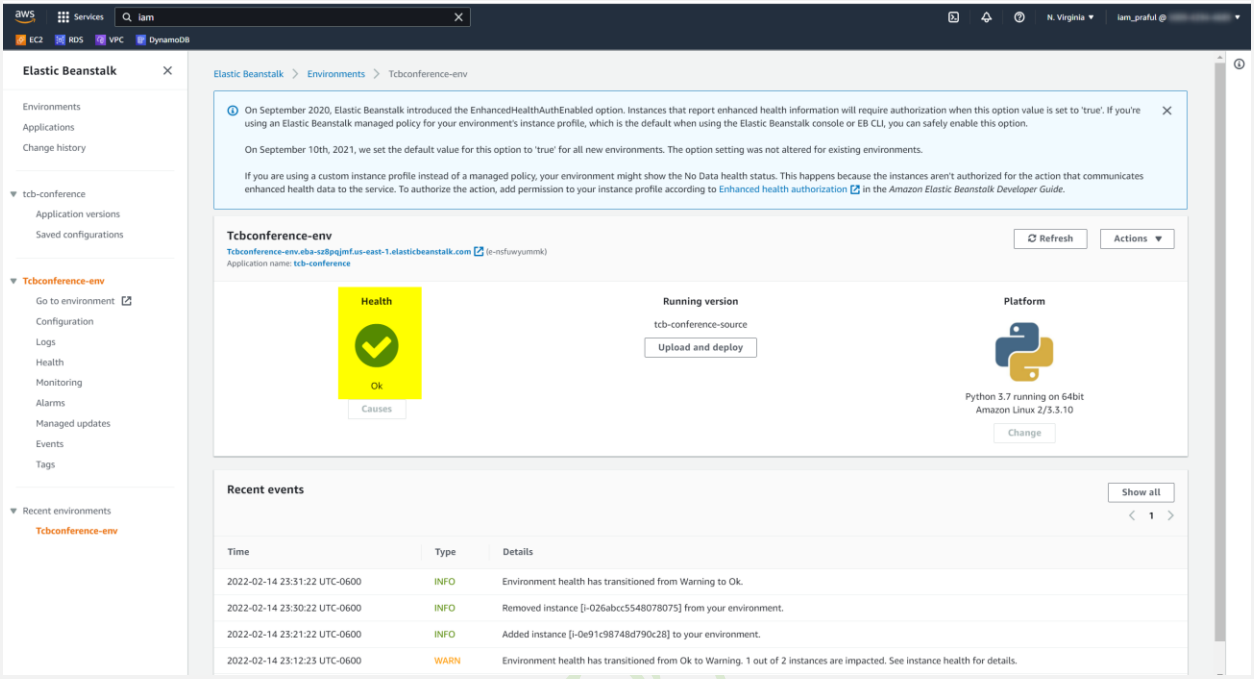




Verify that Data is added successfully to backend database using cloudfront url while loadbalancer is up



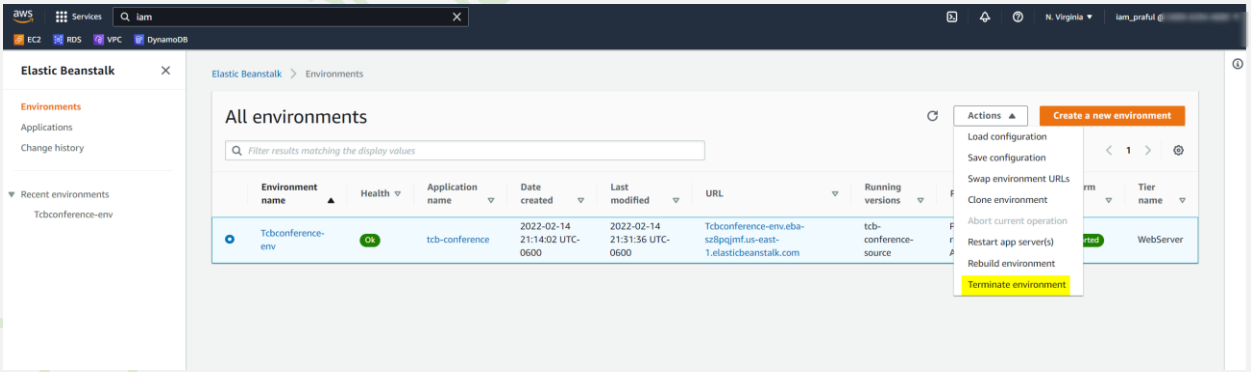
Verify that after few minutes the Elastic Beanstalk is back to Healthy status

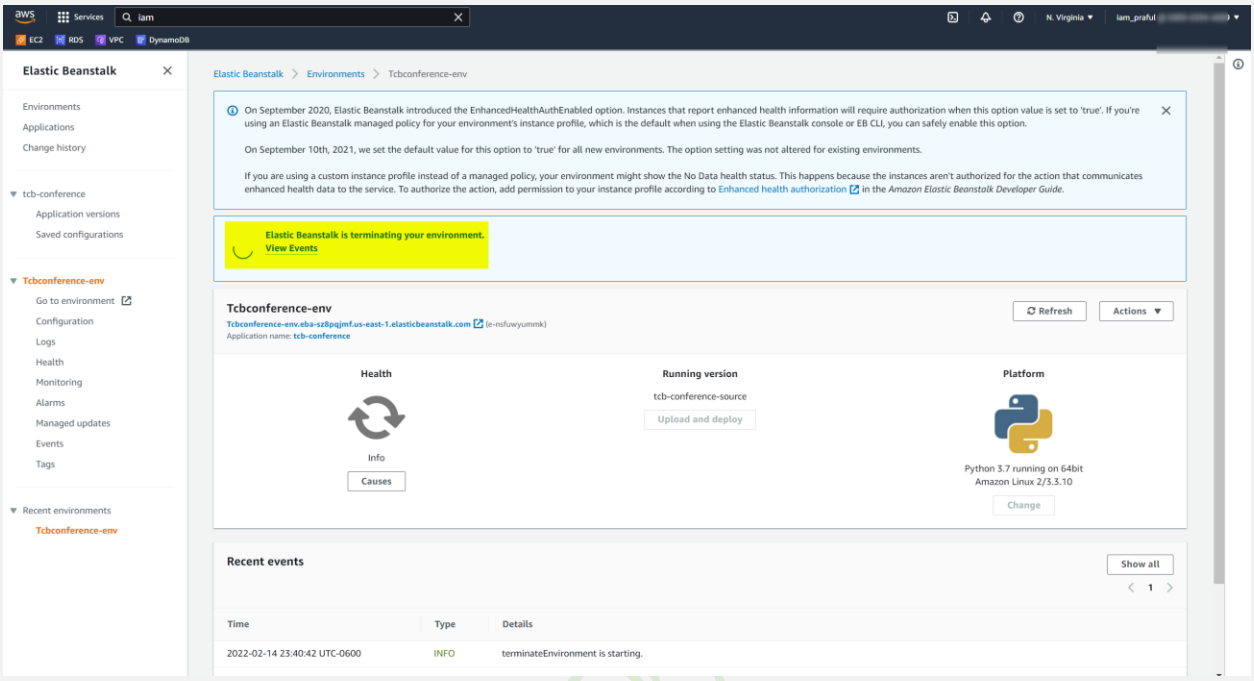


Clean up:

Clean up all resources in order to avoid unnecessary charges on the account.

Terminate Environment





---------- X ----------X ----------X ----------