CloudTrail-CloudWatch-Task

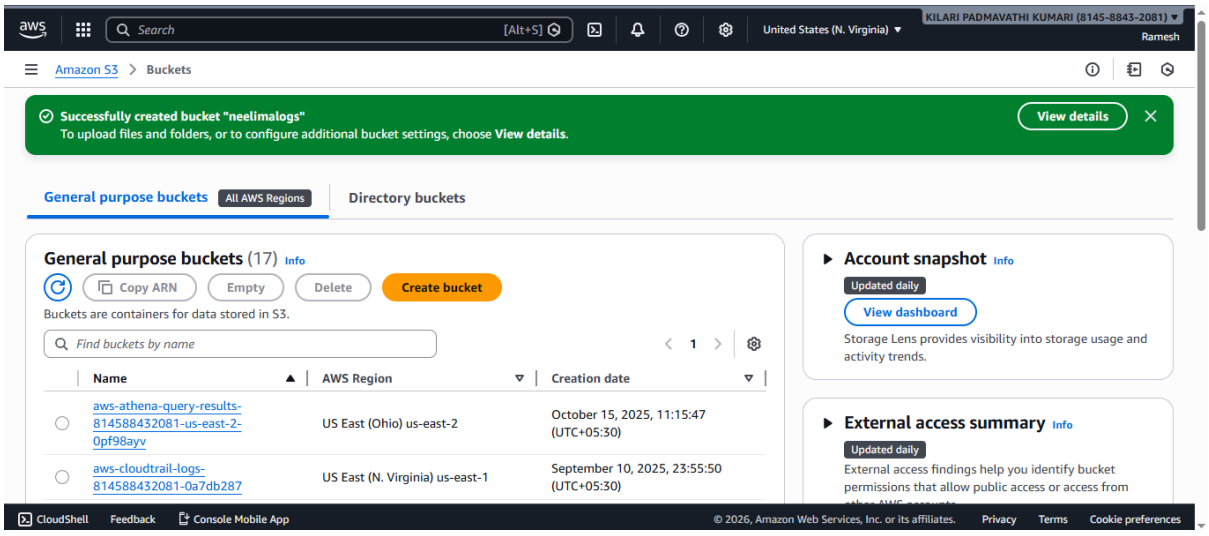
1. Enable CloudTrail monitoring and store the events in S3 and CloudWatch log events.

Architecture

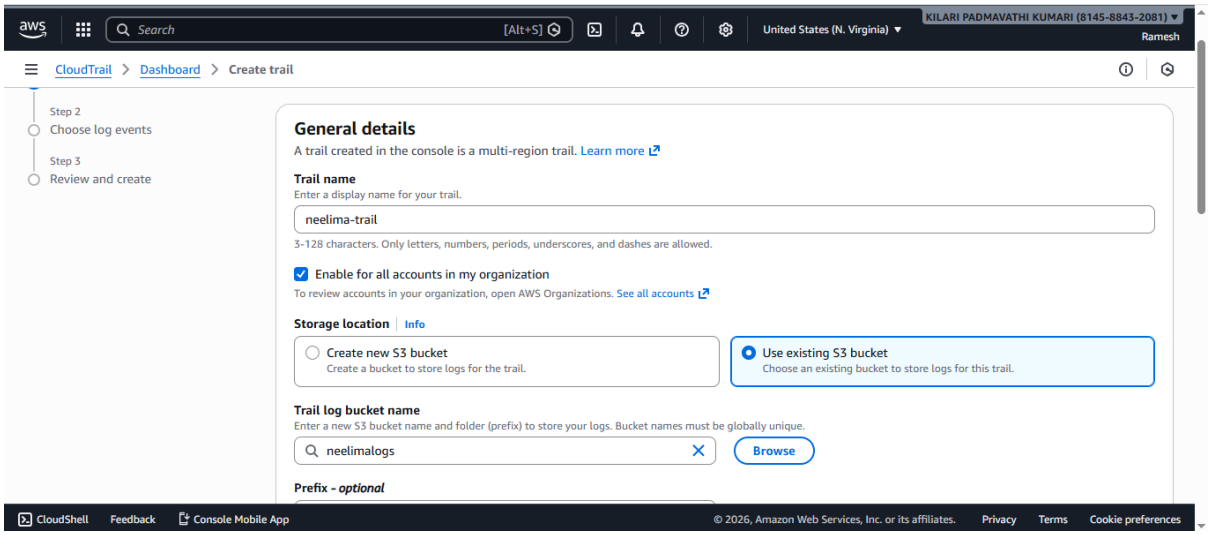
AWS Account
|
CloudTrail
|————> S3 Bucket (long-term storage & audits)
|
|————> CloudWatch Logs (real-time monitoring & alerts)

Step 1:

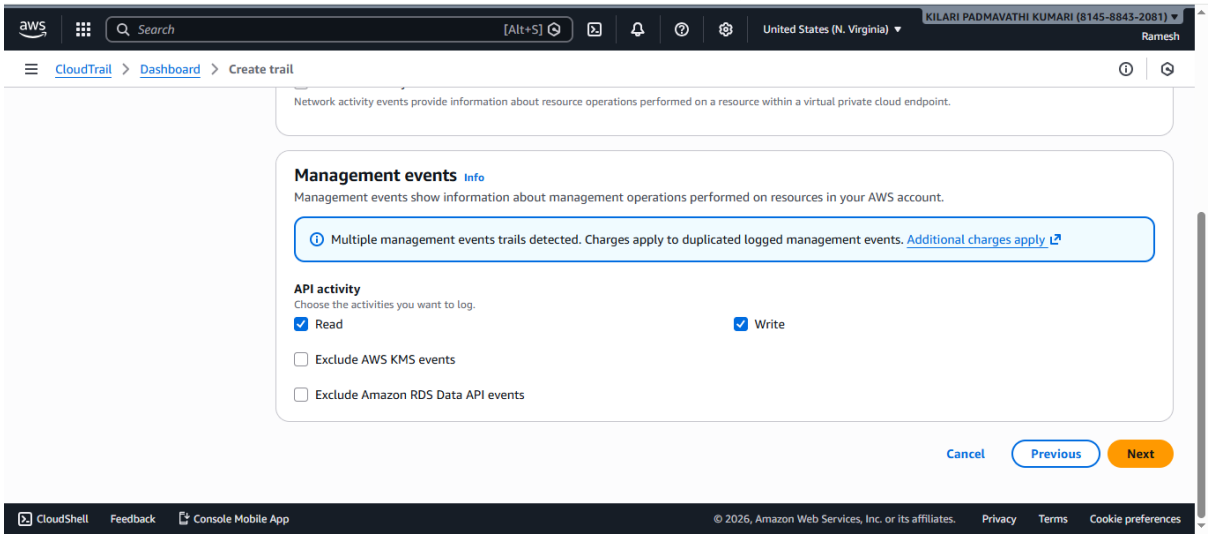
- Go to AWS console
- Search for S3(simple storage service)
- Click on Create bucket.
- Image shows successfully created bucket “neelimalogs”.



CloudTrail-CloudWatch-Task

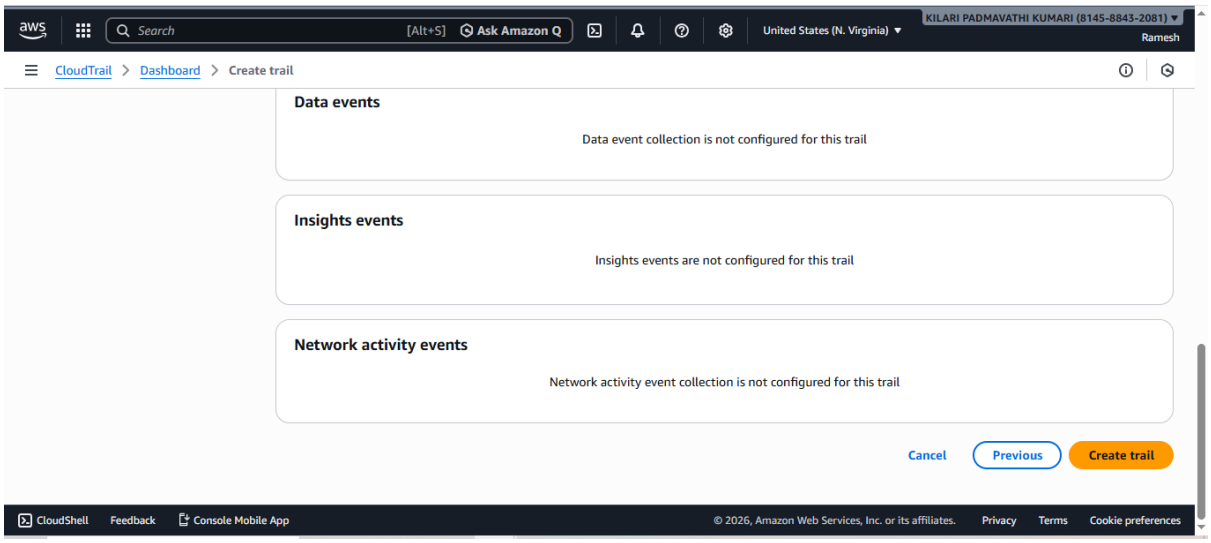


- Go to Cloud Trail
- Click on Dashboard and Create trail.
- Trail Name:neelima-trail.
- In storage location:use existing s3 bucket
- Browse trail log bucket name to store logs in that bucket.

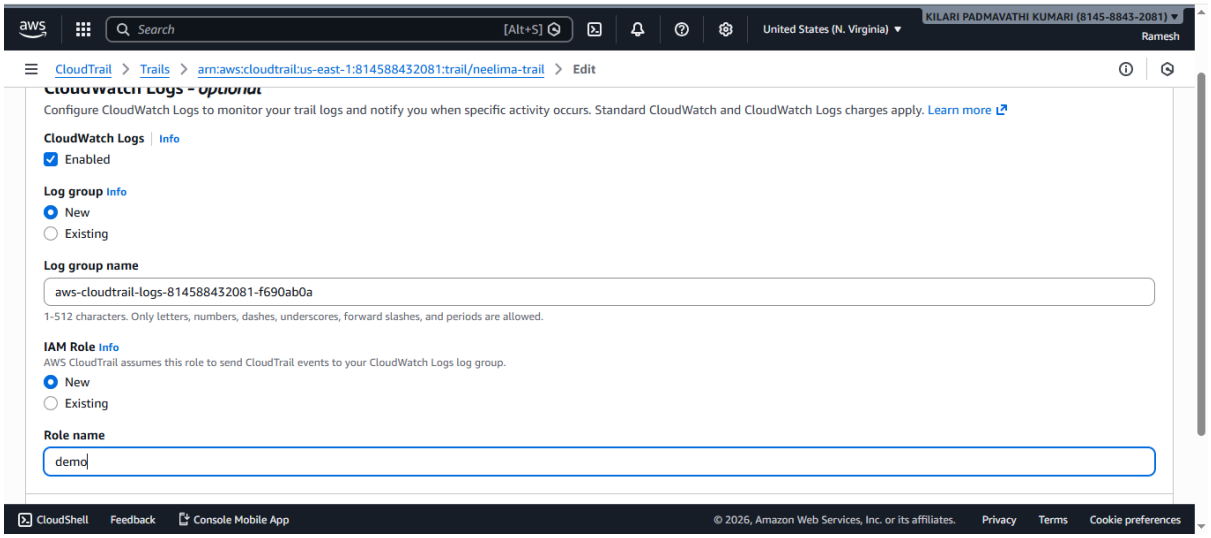


- In Events search for Management events
- API activity as Read and Write
- Click Next.

CloudTrail-CloudWatch-Task

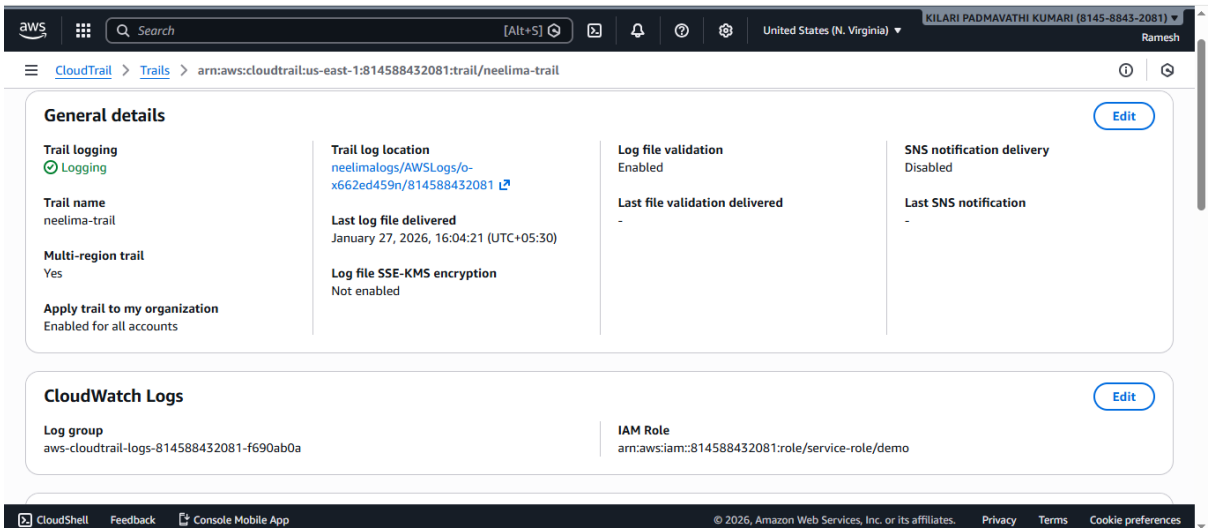


- Click on Create trail.

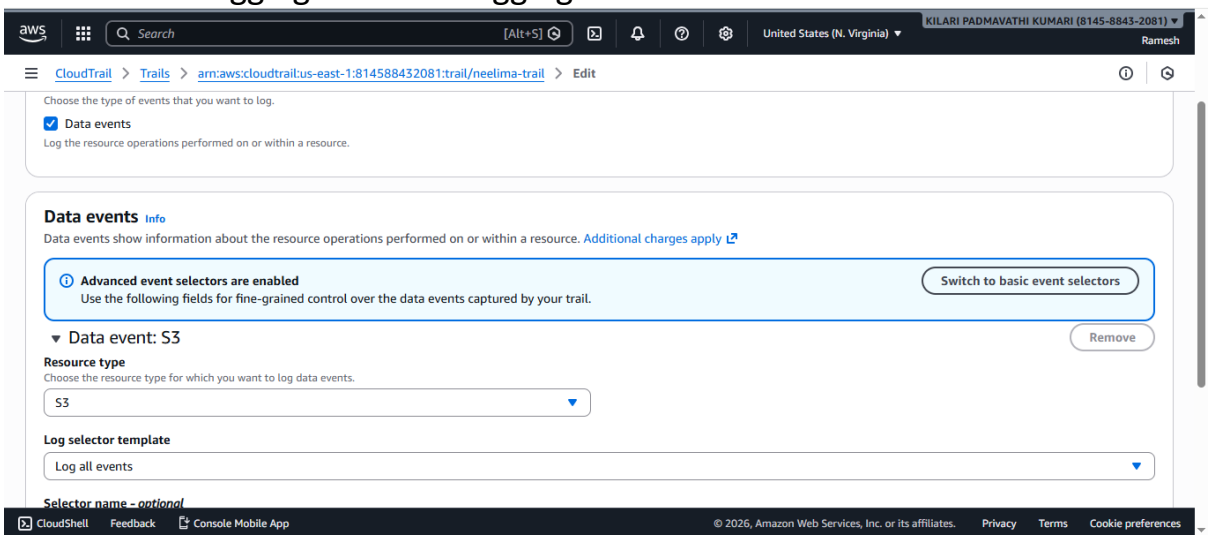


- After creating trail we can edit cloudwatch setting (optional)
- Enable cloudwatch logs to watch logs.
- Log group as new in that logs are stored.
- Create IAM role as new for access permission.
- Click on save.

CloudTrail-CloudWatch-Task

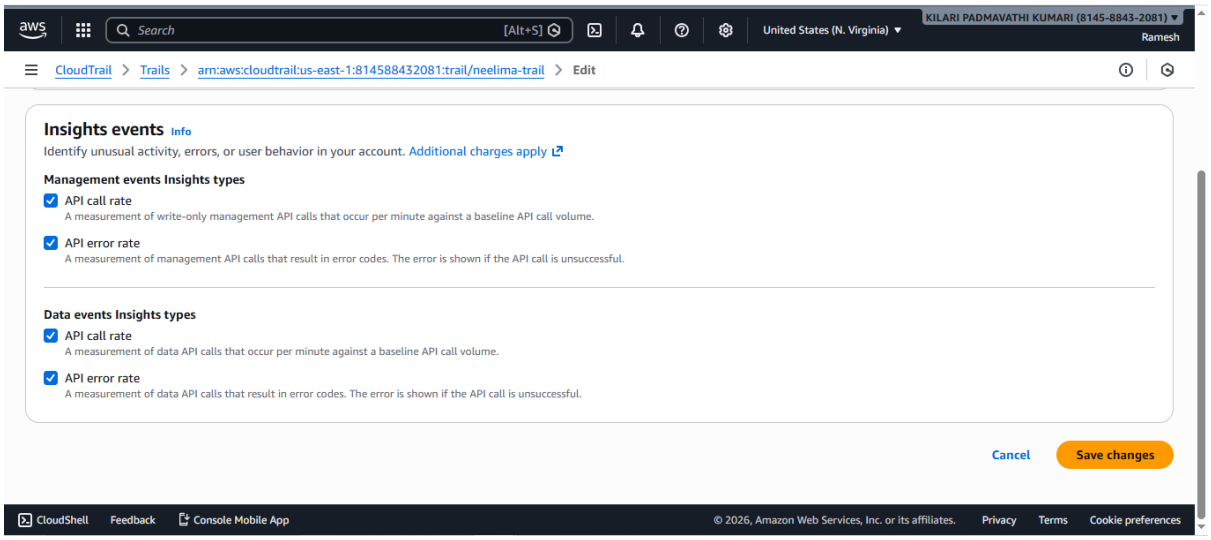


- The above image shows general details of cloud trail and cloud watch
- Trail logging status → logging.

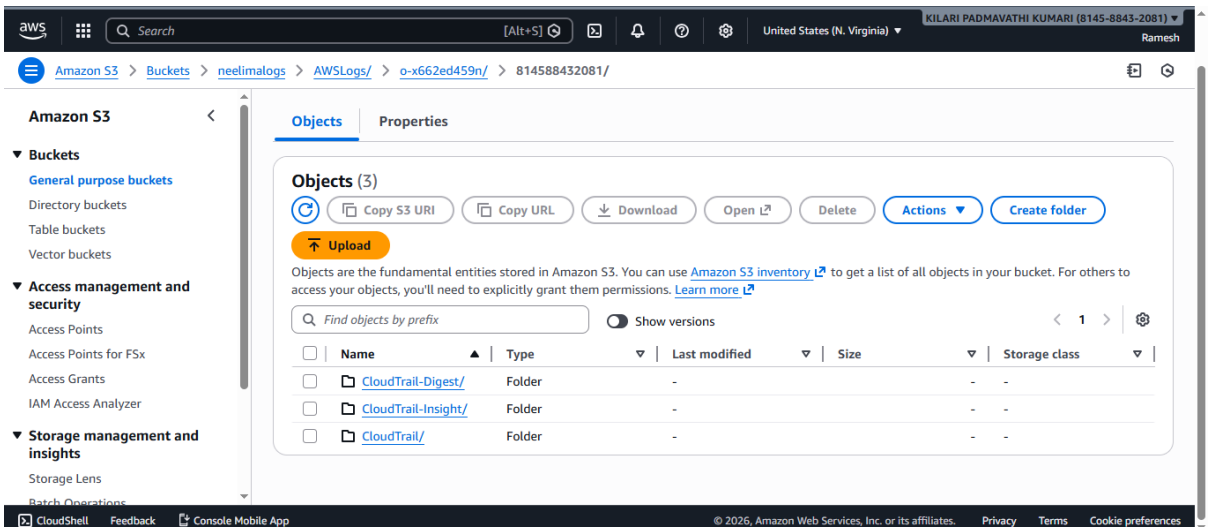


- To check s3 bucket logs we have edit data events
- Data event resource type:S3
- Log selector template:log all events.

CloudTrail-CloudWatch-Task

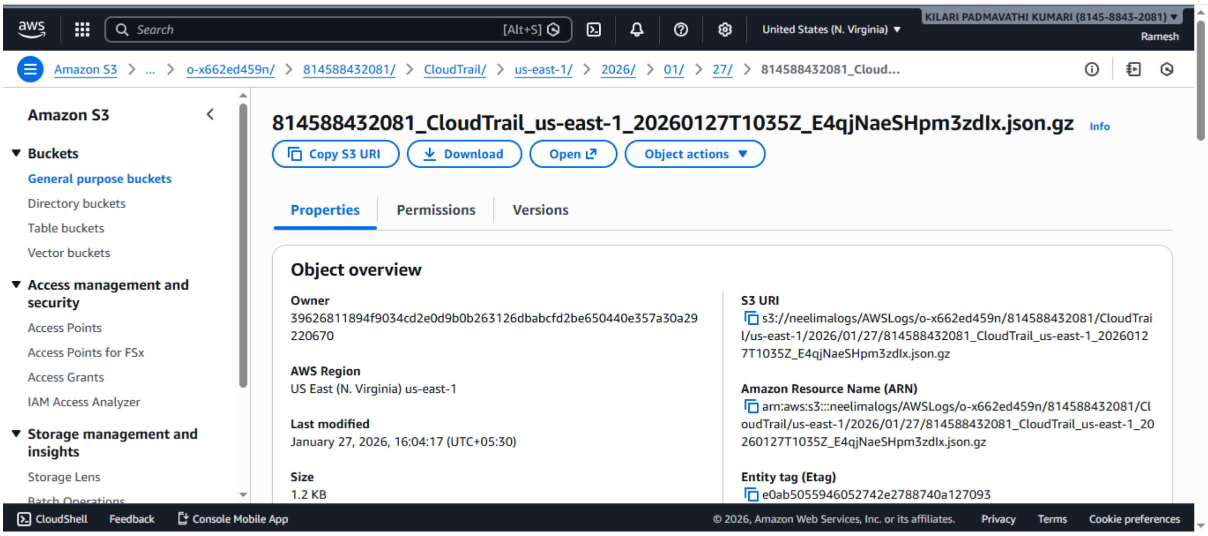


- In Insights events management events insight types
- Enable API call rate and API error rate
- Data event insight type
- Enable API call rate and API error rate
- Click on save changes.

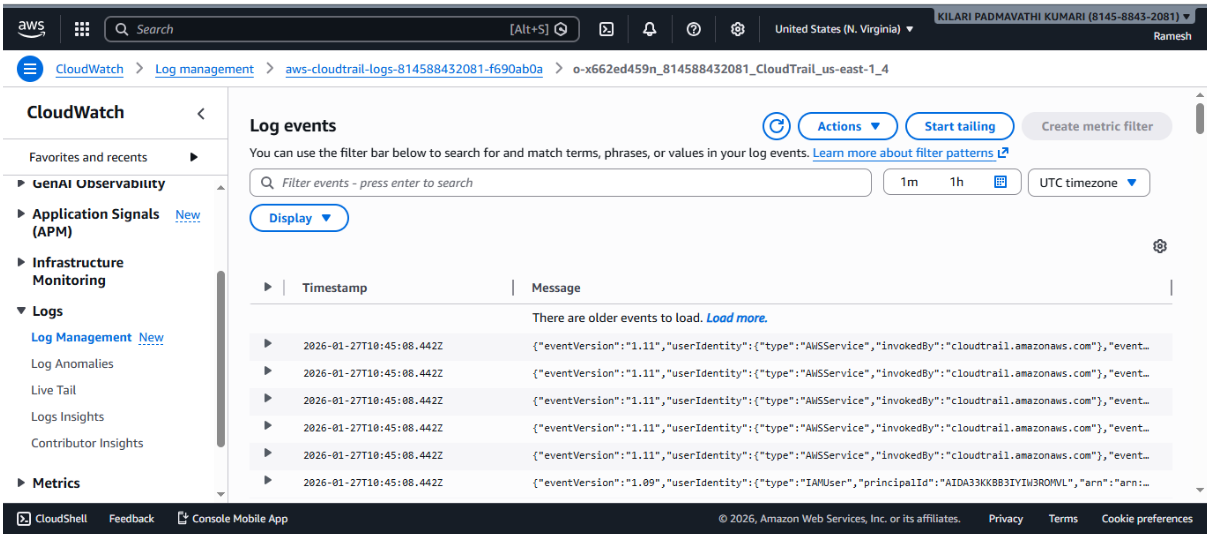


- Go to S3
- Click on buckets
- Select the bucket which we created as neelimalogs
- In object we can find cloudTrail-Digest/, cloudTrail-insight/,cloudTrail/ folders in it.

CloudTrail-CloudWatch-Task



- The above image shows cloudtrail logs.



- In cloudwatch we can see logs in Log management
- The above image displaying logs in cloudwatch.

CloudTrail-CloudWatch-Task

2. Enable SNS for CloudTrail to send alerts via email.

Objective

- Enable **Amazon SNS** so that **CloudTrail events trigger notifications via email** (usually through CloudWatch alarms or EventBridge).

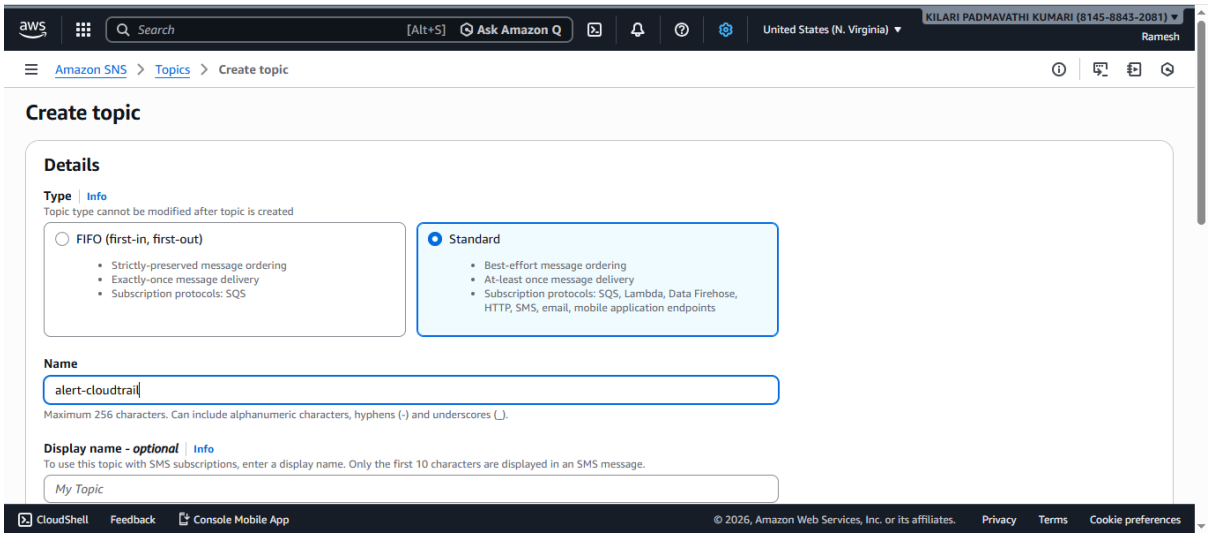
Architecture Flow

- **CloudTrail → CloudWatch Logs / EventBridge → SNS → Email**

Step 1: Create an SNS Topic

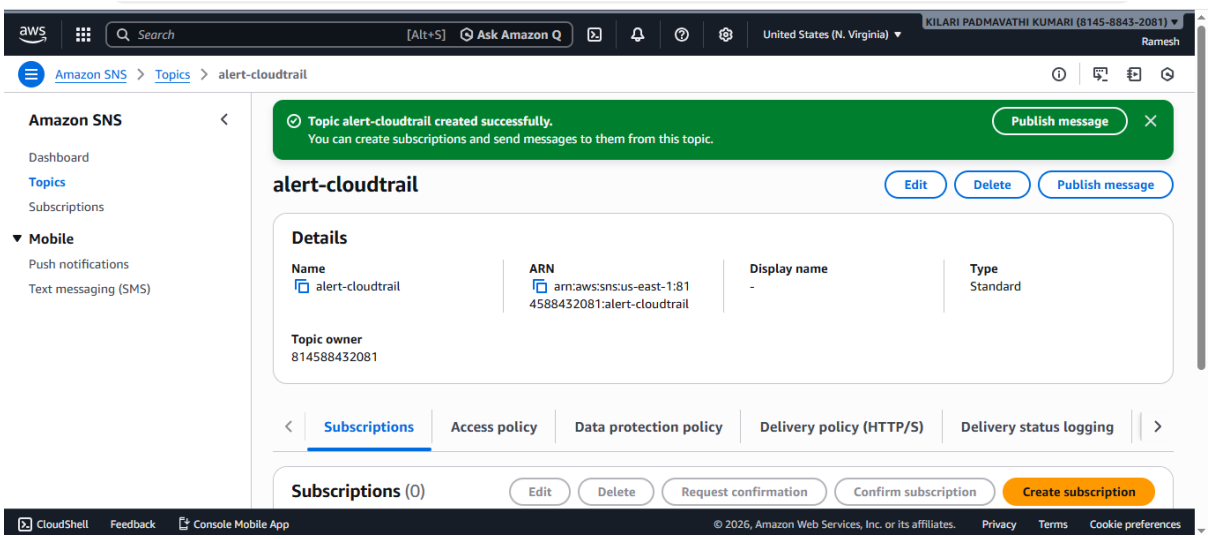
1. Go to **AWS Console → SNS**
2. Click **Topics → Create topic**

Choose:**Type:** Standard



- **Name:** cloudtrail-alerts-topic
- Click **Create topic**

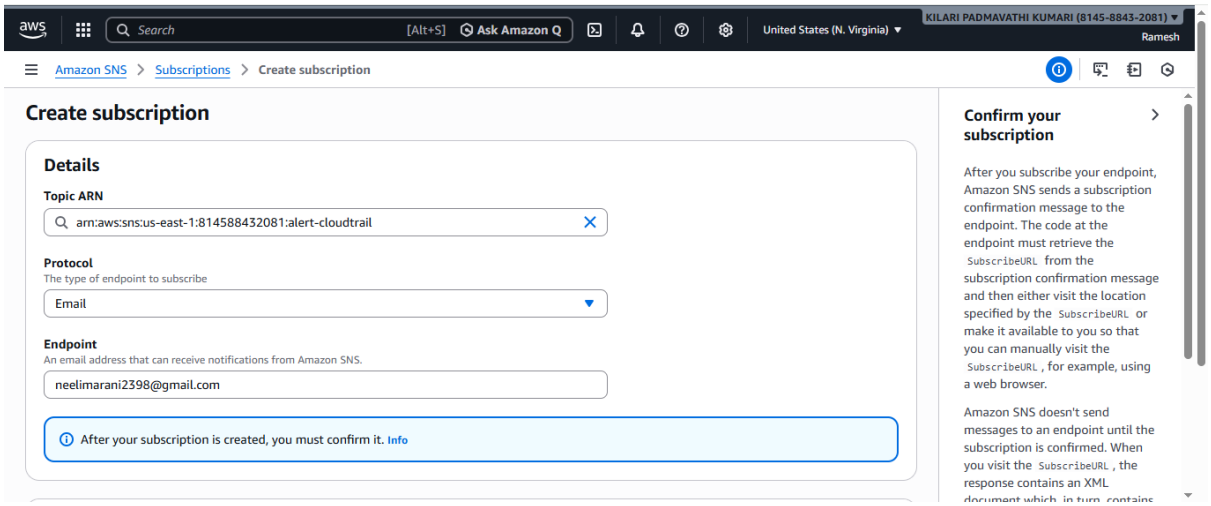
CloudTrail-CloudWatch-Task



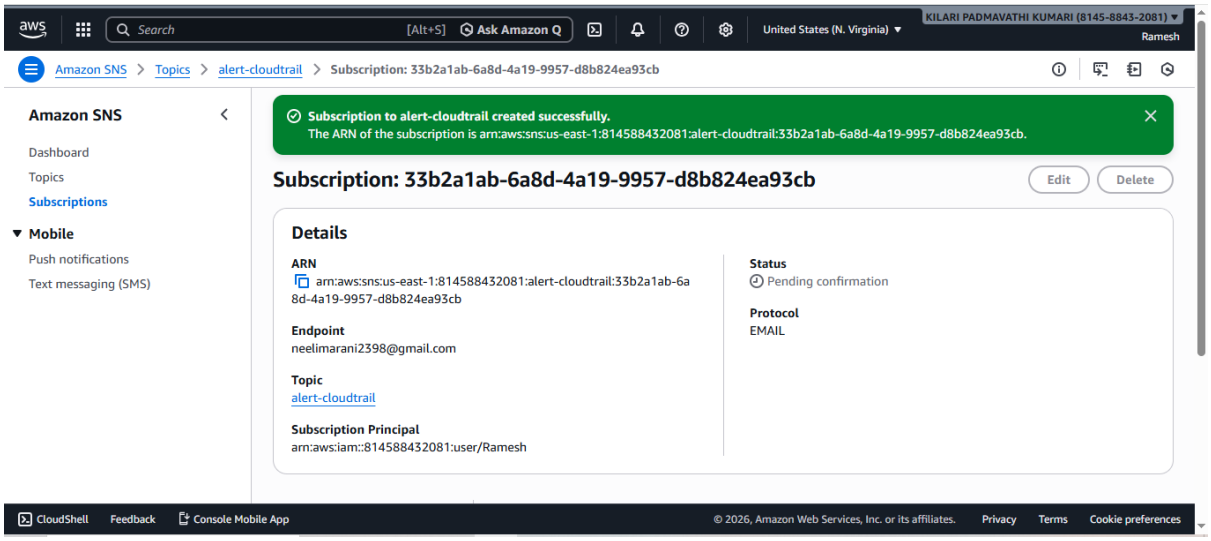
- The above image shows topic create successfully.

Step 2: Create Email Subscription

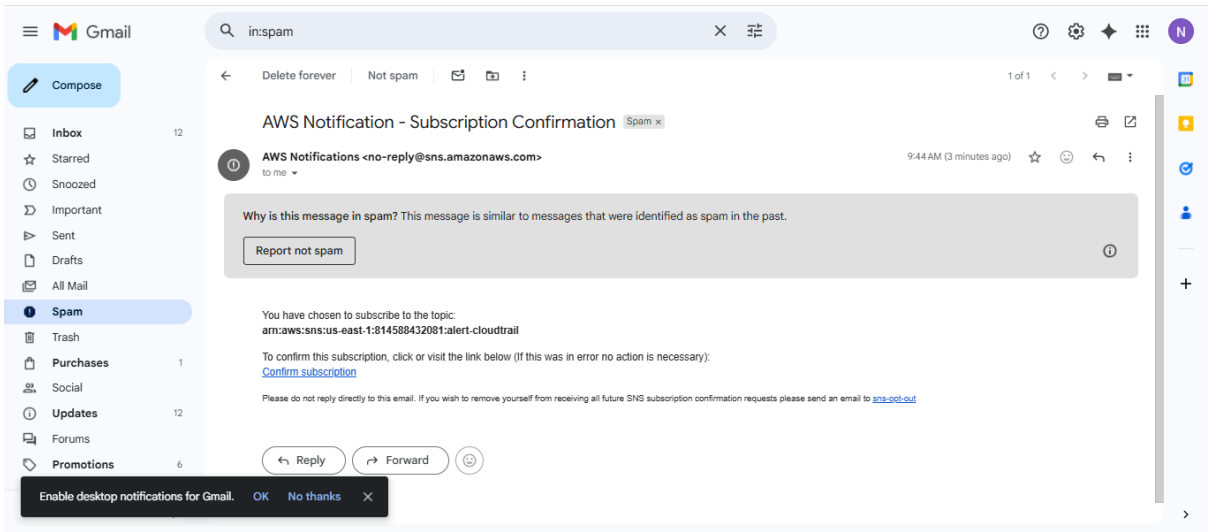
1. Open the created SNS topic
2. Go to **Subscriptions → Create subscription**
3. Select:
    - **Protocol:** Email
    - **Endpoint:** your email address
4. Click **Create subscription**

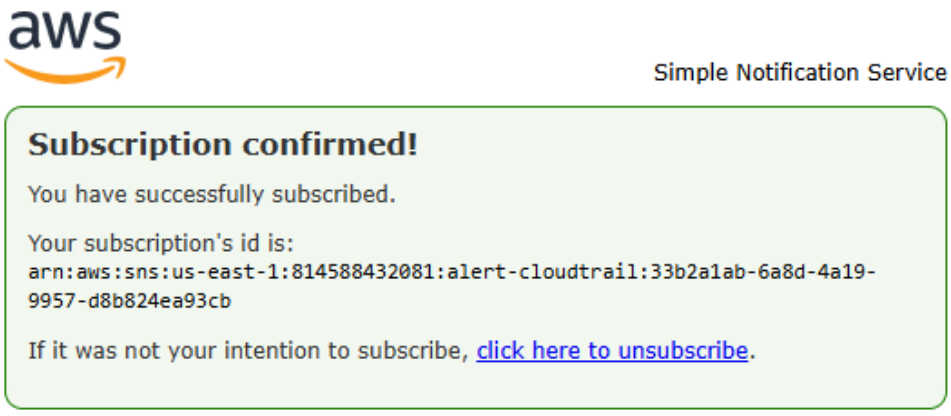


CloudTrail-CloudWatch-Task

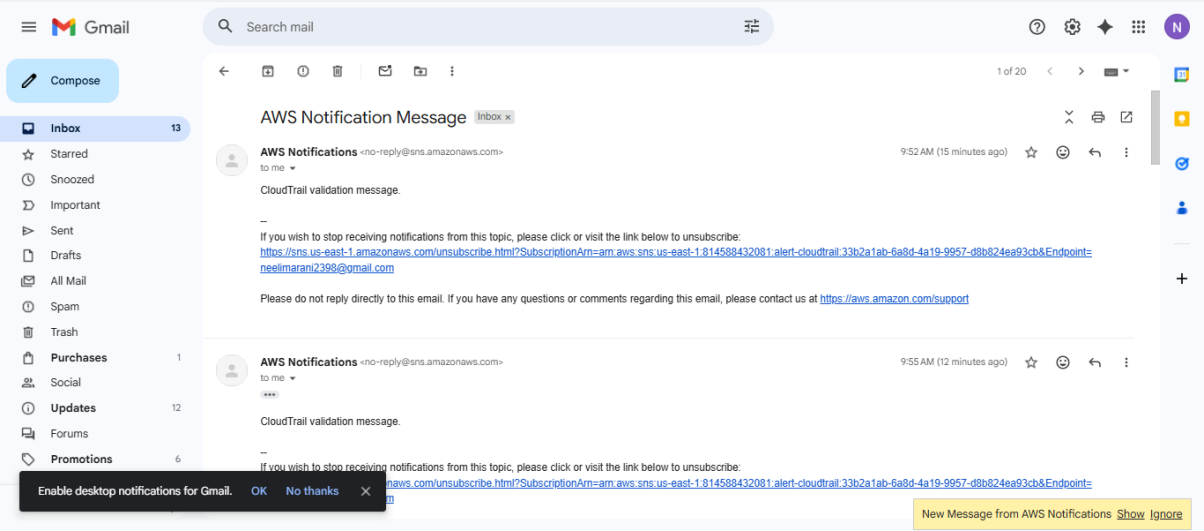


5. **Confirm the subscription** from the email you receive,Until confirmed, alerts will NOT be delivered.





CloudTrail-CloudWatch-Task

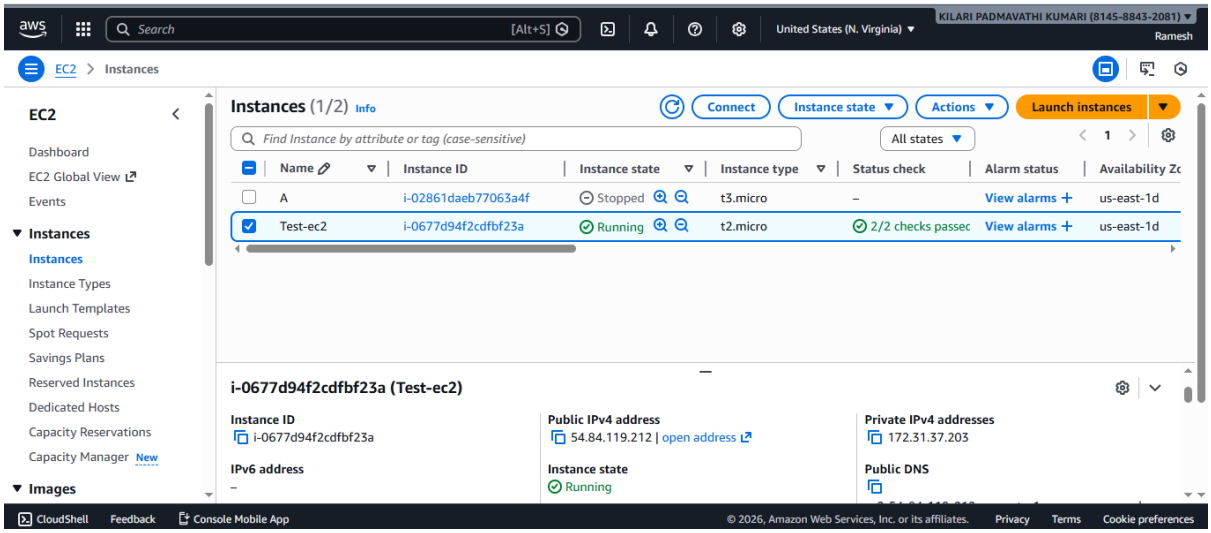


Conclusion: The above image shows that we are receiving AWS Notification message through Email.

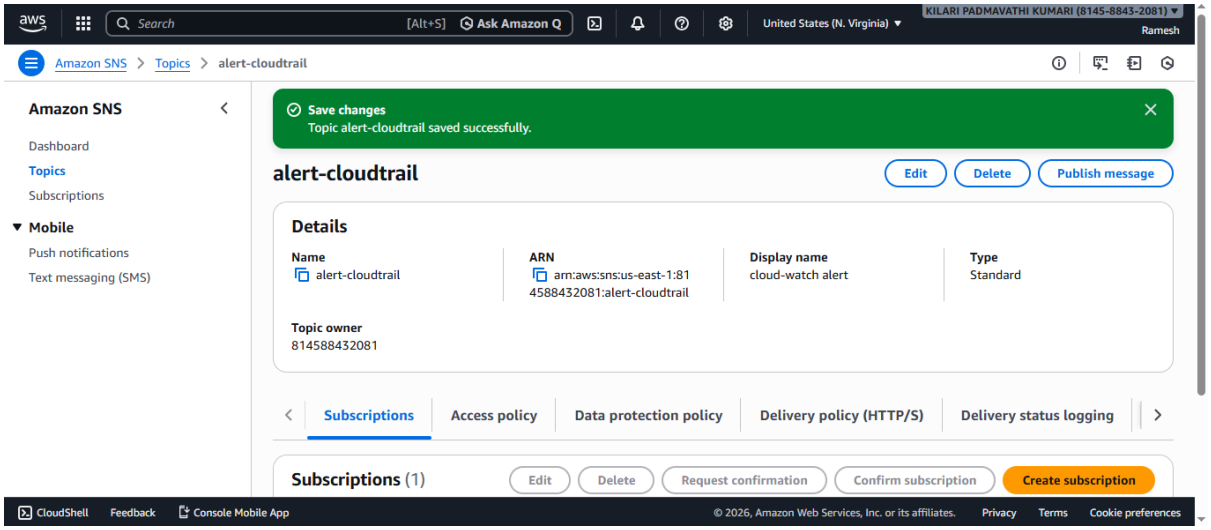
CloudTrail-CloudWatch-Task

3. Create one alarm to send an alert to email if the CPU utilization is more than 50 percent.

- Go to Aws console
- Search for EC2 and create an instance.

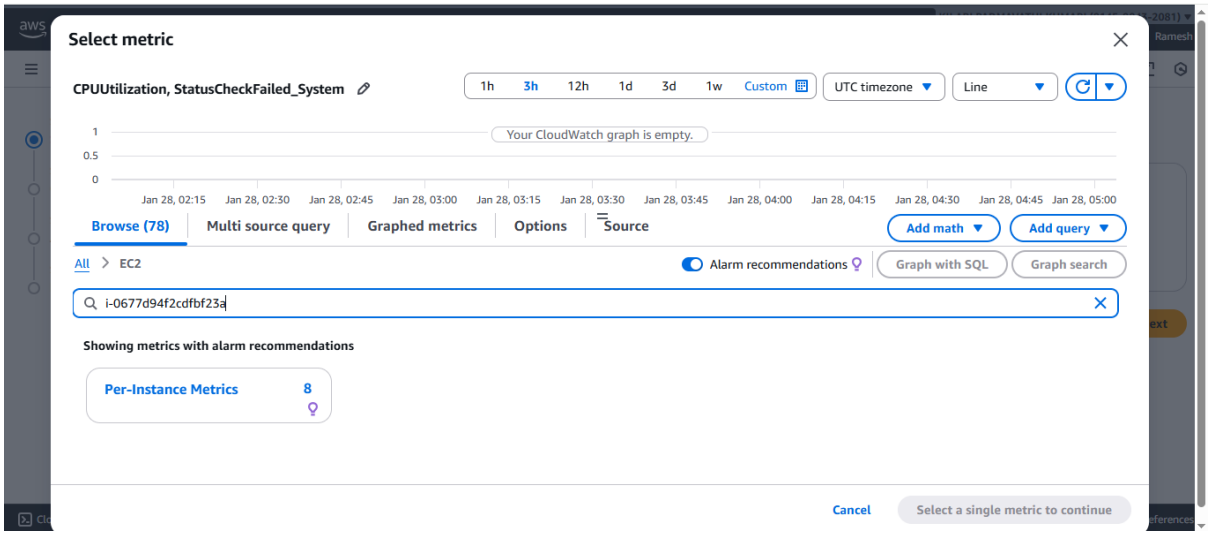


- Go to Aws console
- Search for SNS (simple Notification service)
- Create a topic: -alert CloudTrail and
- Create a Subscription
- click on save.

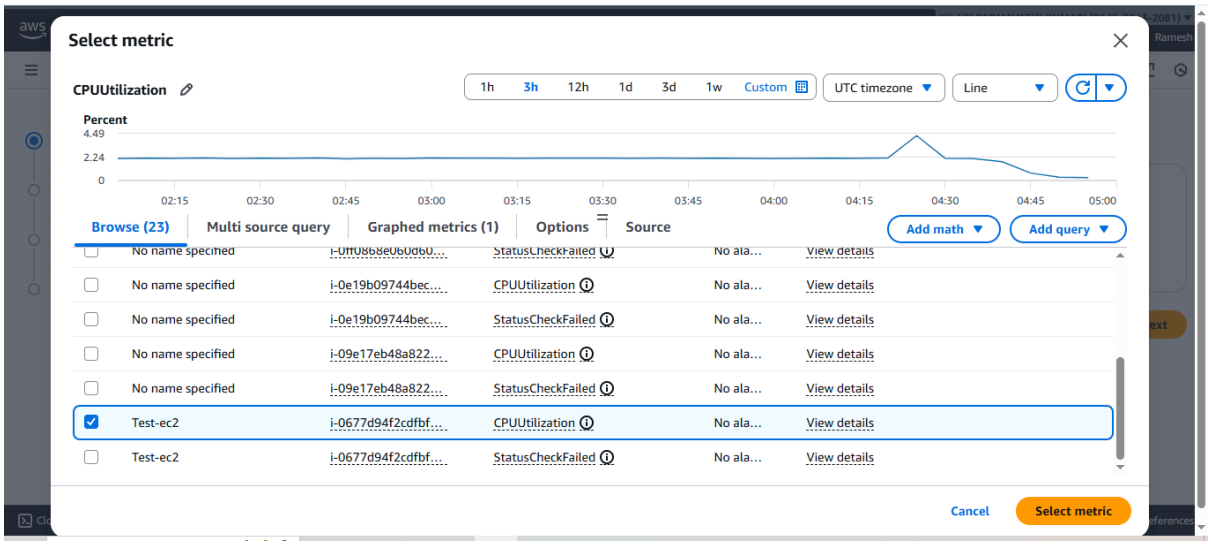


CloudTrail-CloudWatch-Task

- Go to CloudWatch
- And select Metric in that search for EC2.

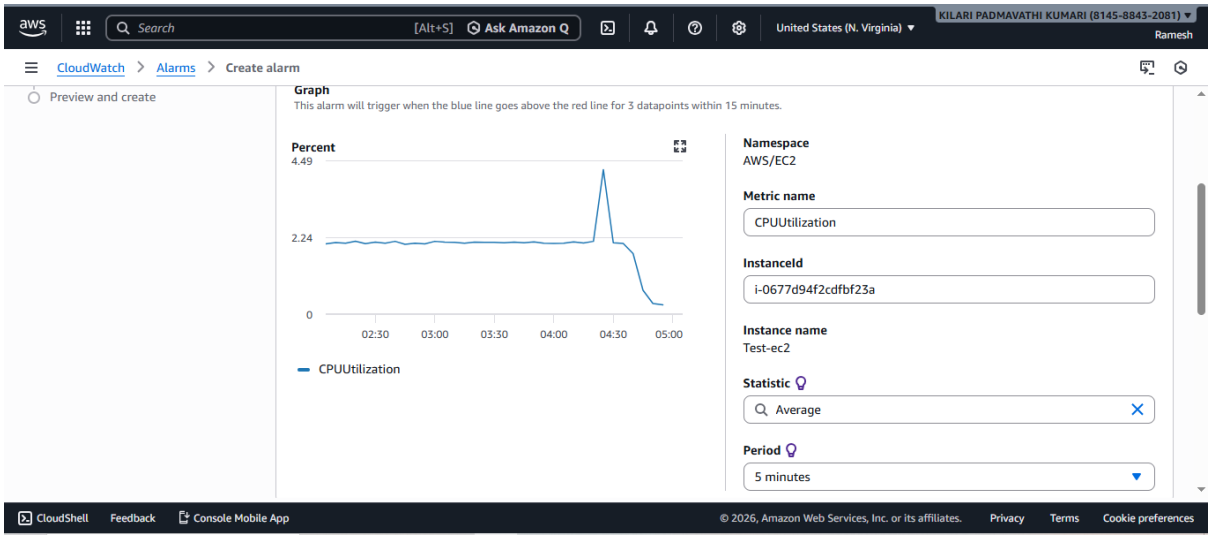


- Find desired EC2 we required and search CPU utilizations

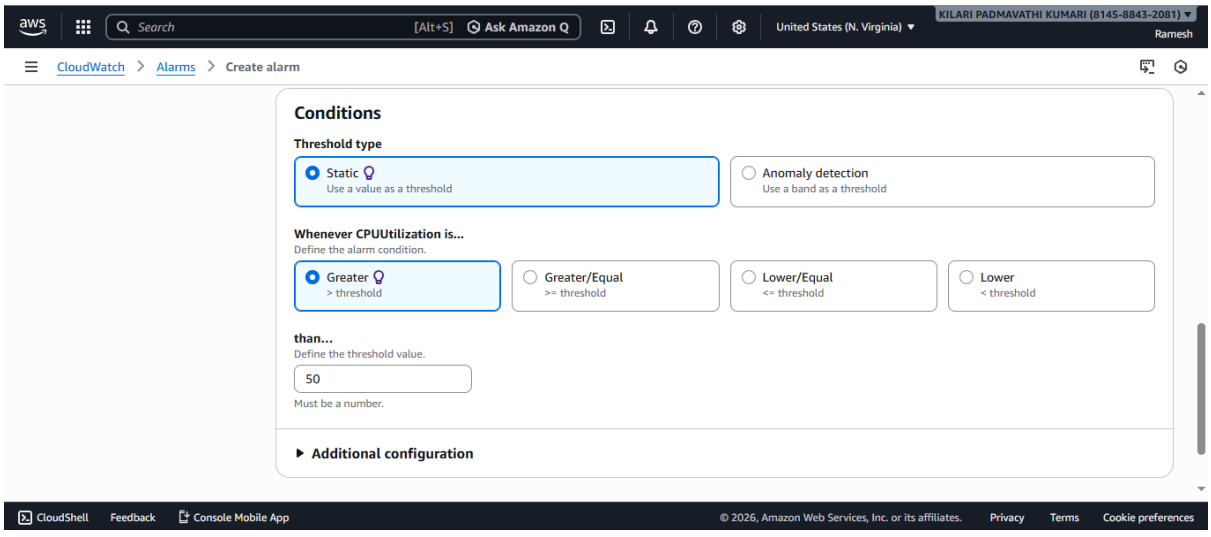


CloudTrail-CloudWatch-Task

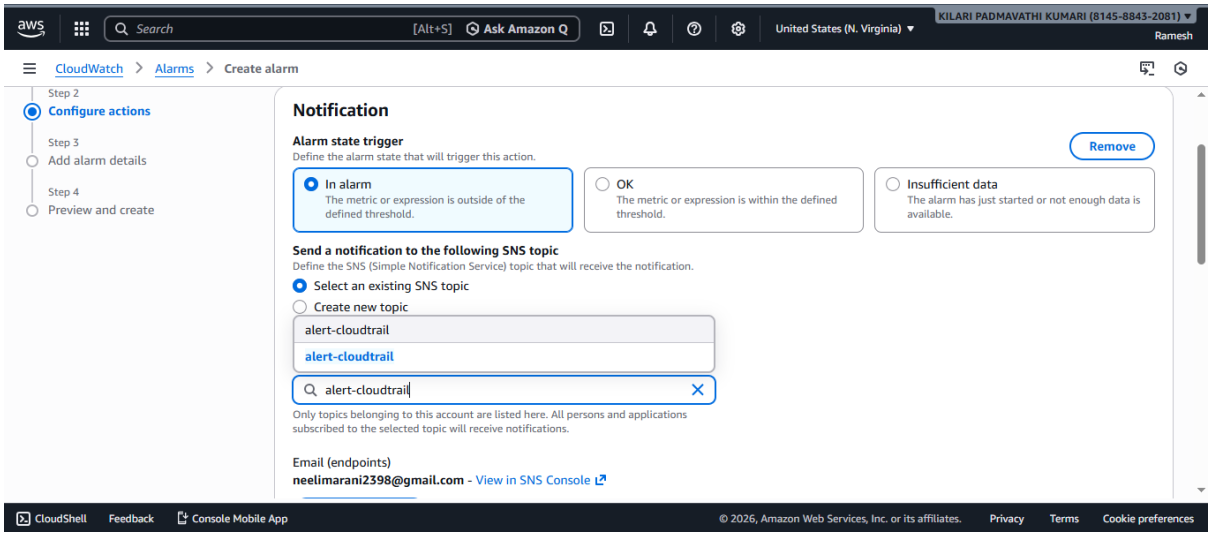
- In CloudWatch click on alarms
- Create Alaram
- Select Metric Name CPU utilization.



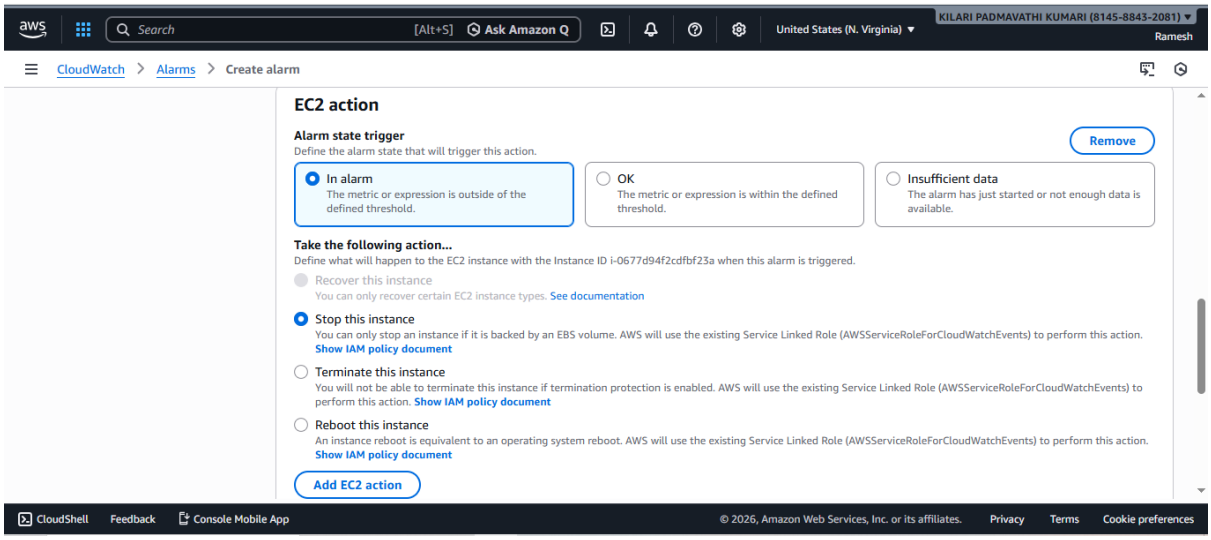
- In conditions Threshold Type as Static
- Whenever CPUUtilization is Greater and Condition in that define the threshold value.



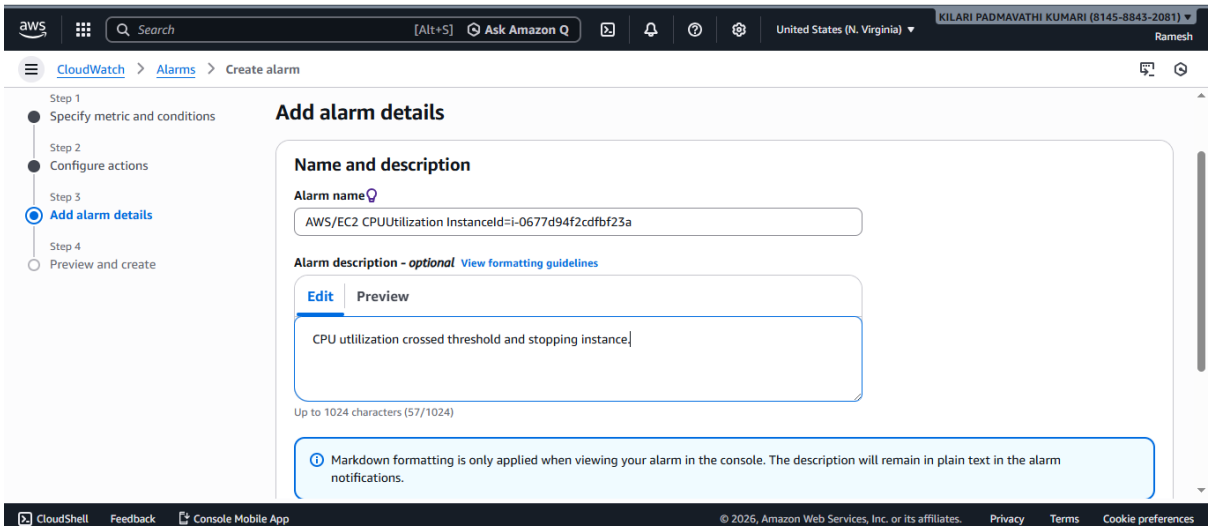
CloudTrail-CloudWatch-Task



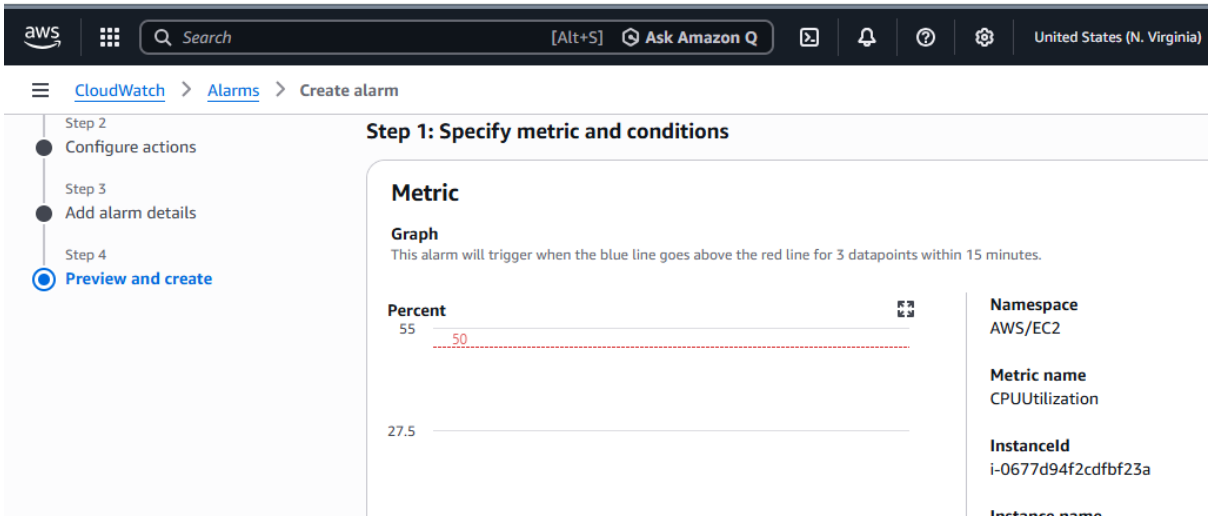
- Configure actions
- Send a notification to the following SNS topic we can select or create Existing SNS topic
- Save it.



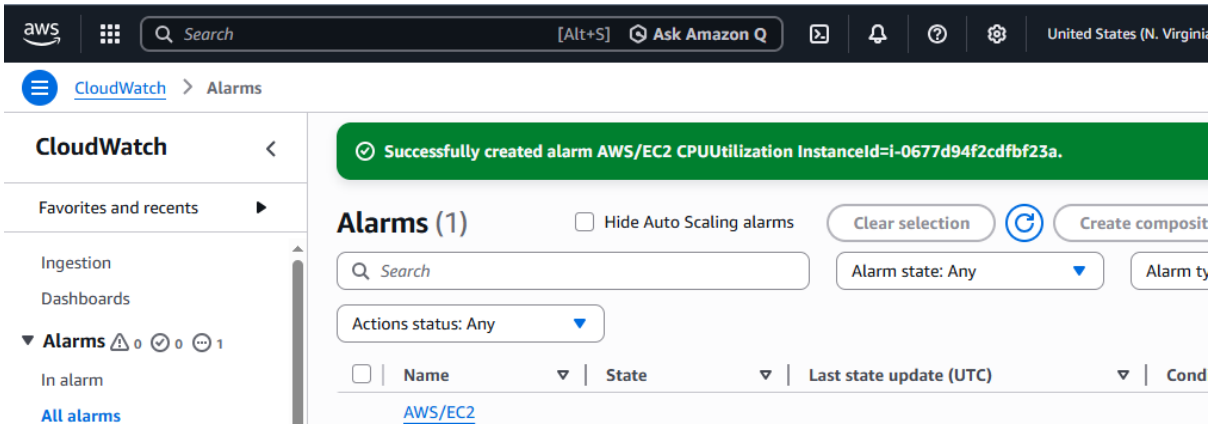
CloudTrail-CloudWatch-Task



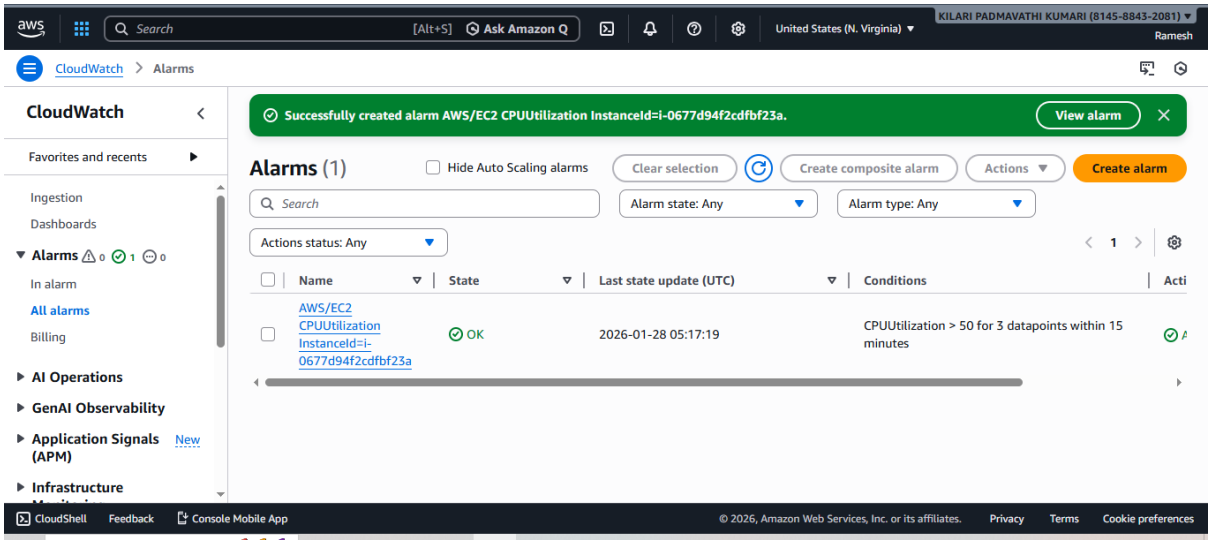
- Add alarm details Name and description
- Alarm description Edit Message.
- Preview and create

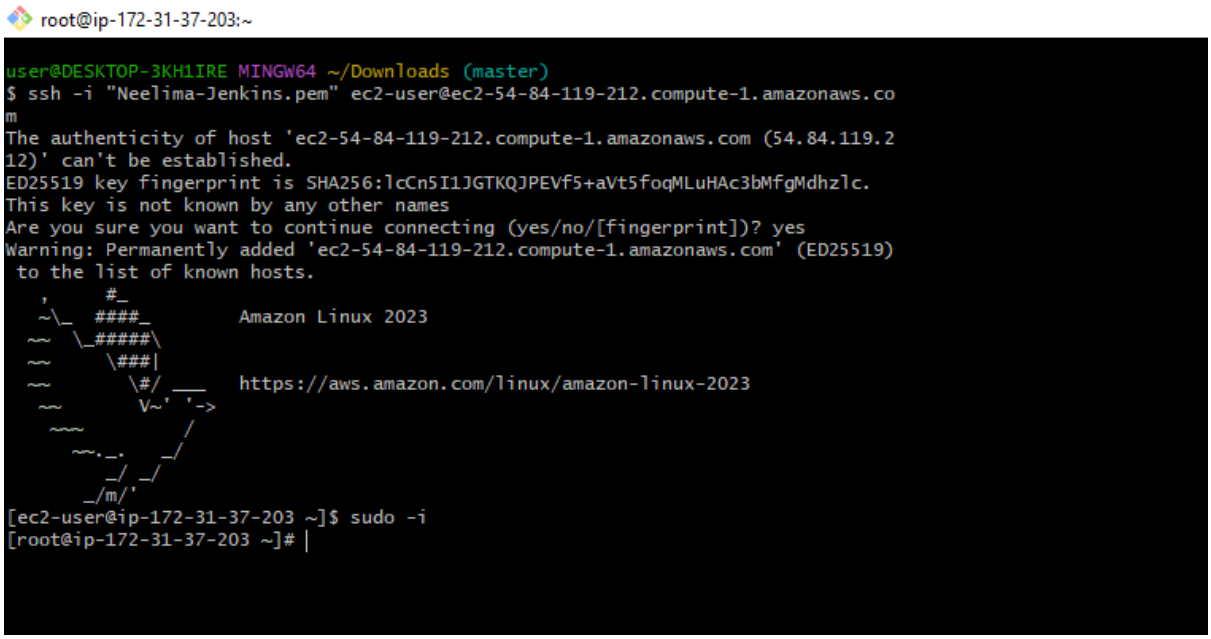


- The below image shows alarm successfully created alarm AWS/EC2 CPUutilizations for instance.



CloudTrail-CloudWatch-Task





- Go to Downloads open Gitbash and Connect ssh
- Change to root user to avoid errors.

CloudTrail-CloudWatch-Task

```
[root@ip-172-31-37-203 ~]# yum update
Amazon Linux 2023 Kernel Livepatch repository   232 kB/s |  30 kB     00:00
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-37-203 ~]# yum install stress-ng
Last metadata expiration check: 0:00:27 ago on Wed Jan 28 05:29:30 2026.
Dependencies resolved.
================================================================================
 Package          Arch       Version                       Repository       Size
================================================================================
Installing:
 stress-ng        x86_64     0.15.05-1.amzn2023            amazonlinux     2.3 M
Installing dependencies:
 Judy             x86_64     1.0.5-25.amzn2023.0.3         amazonlinux     153 k
 libbsd           x86_64     0.10.0-7.amzn2023.0.2         amazonlinux     109 k
 lksctp-tools     x86_64     1.0.18-9.amzn2023.0.3         amazonlinux      92 k

Transaction Summary
================================================================================
Install  4 Packages

Total download size: 2.7 M
Installed size: 9.7 M
Is this ok [y/N]: y
```

- To provide stress to CPU we have to install stress-ng.

  Command:yum install stress-ng

```
root@ip-172-31-37-203:~
Last metadata expiration check: 0:00:27 ago on Wed Jan 28 05:29:30 2026.
Dependencies resolved.
================================================================================
 Package          Arch       Version                       Repository       Size
================================================================================
Installing:
 stress-ng        x86_64     0.15.05-1.amzn2023            amazonlinux     2.3 M
Installing dependencies:
 Judy             x86_64     1.0.5-25.amzn2023.0.3         amazonlinux     153 k
 libbsd           x86_64     0.10.0-7.amzn2023.0.2         amazonlinux     109 k
 lksctp-tools     x86_64     1.0.18-9.amzn2023.0.3         amazonlinux      92 k

Transaction Summary
================================================================================
Install  4 Packages

Total download size: 2.7 M
Installed size: 9.7 M
Is this ok [y/N]: y
Downloading Packages:
(1/4): libbsd-0.10.0-7.amzn2023.0.2.x86_64.rpm                  2.7 MB/s | 109 kB     00:00
(2/4): lksctp-tools-1.0.18-9.amzn2023.0.3.x86_64.rpm            2.0 MB/s |  92 kB     00:00
(3/4): Judy-1.0.5-25.amzn2023.0.3.x86_64.rpm                    2.3 MB/s | 153 kB     00:00
(4/4): stress-ng-0.15.05-1.amzn2023.x86_64.rpm                   10 MB/s | 2.3 MB     00:00
--------------------------------------------------------------------------------
Total                                                            8.7 MB/s | 2.7 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : lksctp-tools-1.0.18-9.amzn2023.0.3.x86_64              1/4
  Installing       : libbsd-0.10.0-7.amzn2023.0.2.x86_64                    2/4
  Installing       : Judy-1.0.5-25.amzn2023.0.3.x86_64                      3/4
  Installing       : stress-ng-0.15.05-1.amzn2023.x86_64                    4/4
  Running scriptlet: stress-ng-0.15.05-1.amzn2023.x86_64                    4/4
  Verifying        : Judy-1.0.5-25.amzn2023.0.3.x86_64                      1/4
  Verifying        : libbsd-0.10.0-7.amzn2023.0.2.x86_64                    2/4
  Verifying        : lksctp-tools-1.0.18-9.amzn2023.0.3.x86_64              3/4
  Verifying        : stress-ng-0.15.05-1.amzn2023.x86_64                    4/4

Installed:
  Judy-1.0.5-25.amzn2023.0.3.x86_64          libbsd-0.10.0-7.amzn2023.0.2.x86_64          lksctp-tools-1.0.18-9.amzn2023.0.3.x86_64          stress-ng-0.15.05-1.amzn2023.x86_64

Complete!
[root@ip-172-31-37-203 ~]#
```

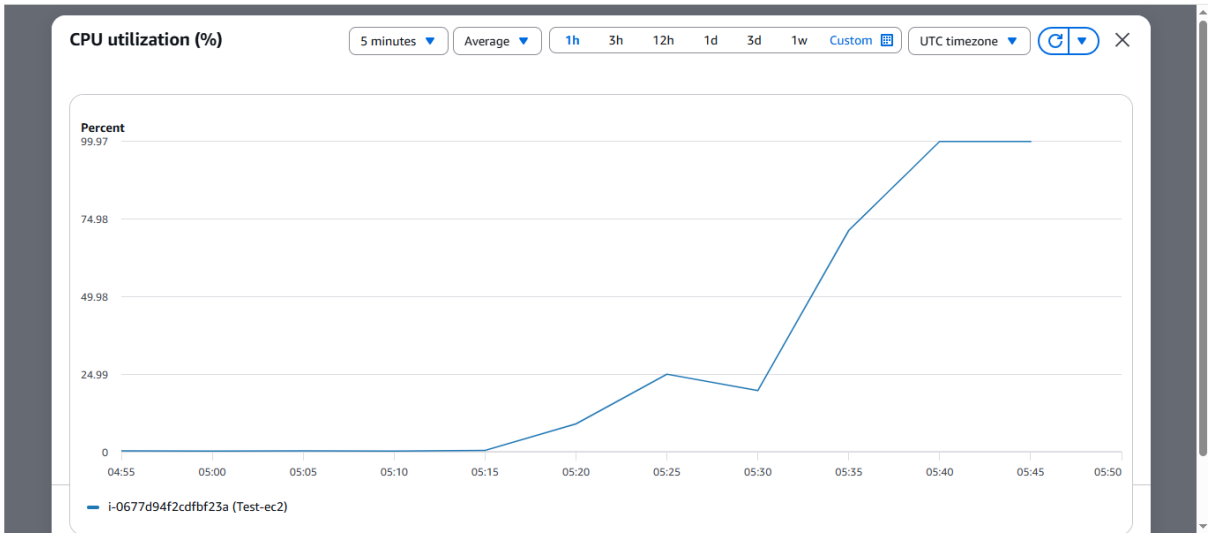
- Installation completed

CloudTrail-CloudWatch-Task

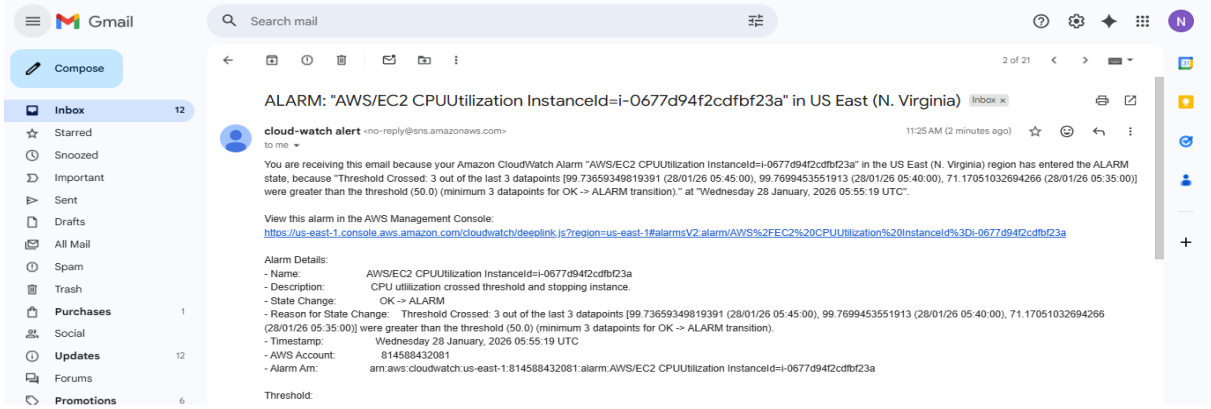
```
[root@ip-172-31-37-203 ~]# stress-ng --cpu 4 --cpu-load 100 --timeout 60s
stress-ng: info:  [27322] setting to a 60 second run per stressor
stress-ng: info:  [27322] dispatching hogs: 4 cpu
stress-ng: info:  [27322] successful run completed in 60.15s (1 min, 0.15 secs)
[root@ip-172-31-37-203 ~]# stress-ng --cpu 4 --cpu-load 100 --timeout 350s
stress-ng: info:  [27580] setting to a 350 second (5 mins, 50.00 secs) run per s
tressor
stress-ng: info:  [27580] dispatching hogs: 4 cpu

Broadcast message from root@ip-172-31-37-203.ec2.internal (Wed 2026-01-28 05:55:20 UTC):

The system will power off now!


Broadcast message from root@ip-172-31-37-203.ec2.internal (Wed 2026-01-28 05:55:20 UTC):

The system will power off now!

Connection to ec2-54-84-119-212.compute-1.amazonaws.com closed by remote host.
Connection to ec2-54-84-119-212.compute-1.amazonaws.com closed.

user@DESKTOP-3KH1IRE MINGW64 ~/Downloads (master)
$
```

- Command to increase stress as follows

  Stress-ng –cpu 4 –cpu-load 100 –timeout 60s

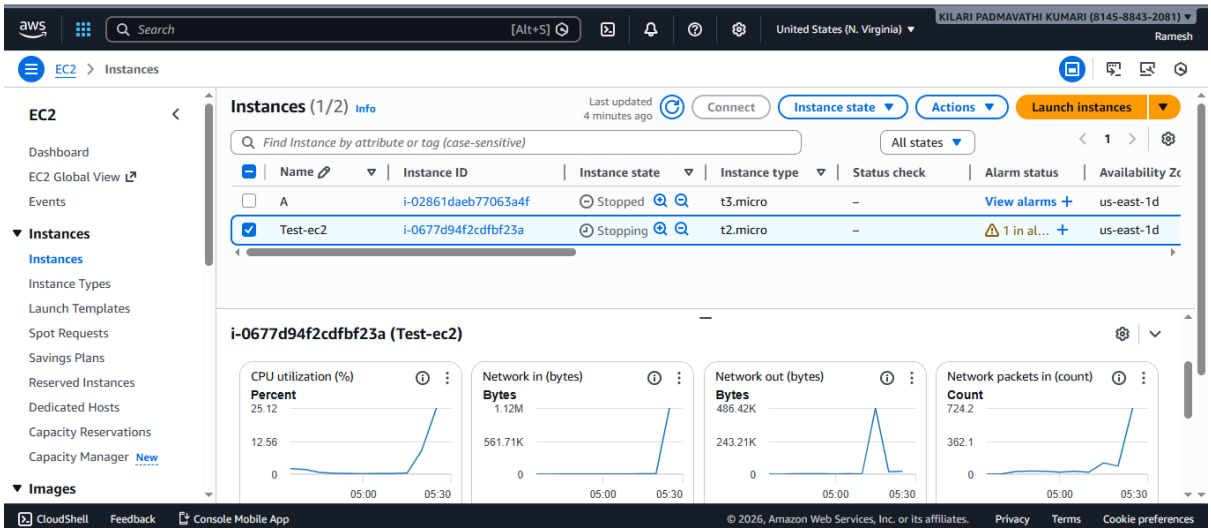


- In CPU monitoring we can see increase of CPU utilization(%)



- The above image shows trigger a notification for our Email through SNS.

CloudTrail-CloudWatch-Task



Conclusion:- After reaching threshold the instance stopped automatically and it provides alarm and SNS also.

CloudTrail-CloudWatch-Task

4. Configure CloudWatch monitoring and record the CPU utilization and other metrics of EC2.

Step 1: Verify CloudWatch Agent (default vs detailed)

Default Monitoring (automatic)

By default, EC2 sends **basic metrics every 5 minutes**:

- CPUUtilization
- NetworkIn / NetworkOut
- DiskReadOps / DiskWriteOps
- StatusCheckFailed

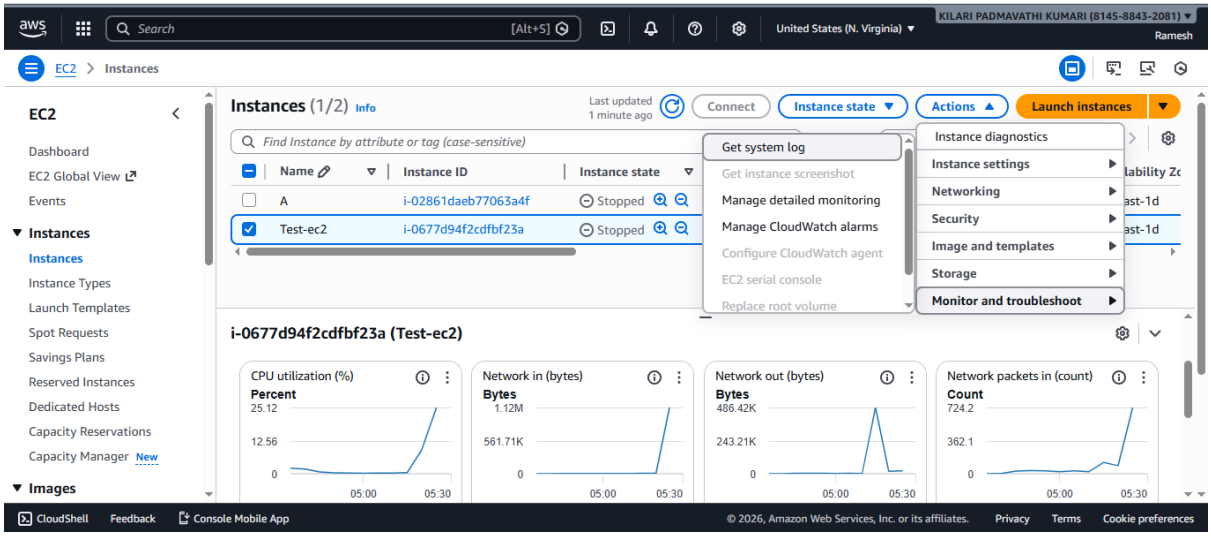
No setup needed

Detailed Monitoring (recommended)

- Records metrics **every 1 minute**.

  Enable it:

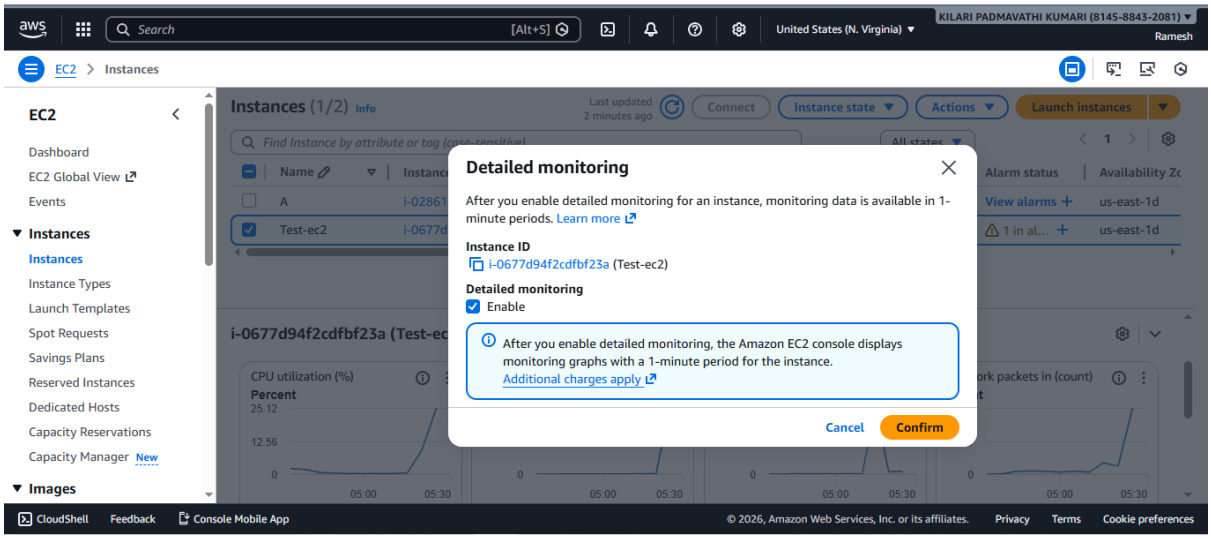
1. Go to **EC2 Console**
2. Select your instance
3. Click **Actions → Monitor and troubleshoot → Manage detailed monitoring**



4. Enable **Detailed monitoring**

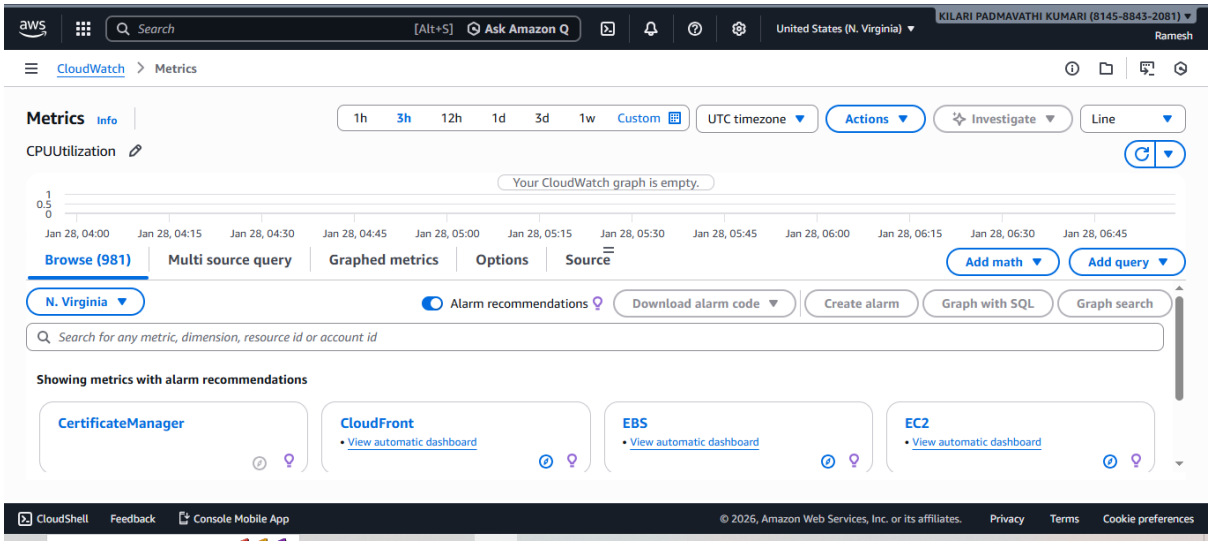
CloudTrail-CloudWatch-Task

5. Save



Step 2: View CPU Utilization

1. Go to **CloudWatch → Metrics**
2. Select **EC2**



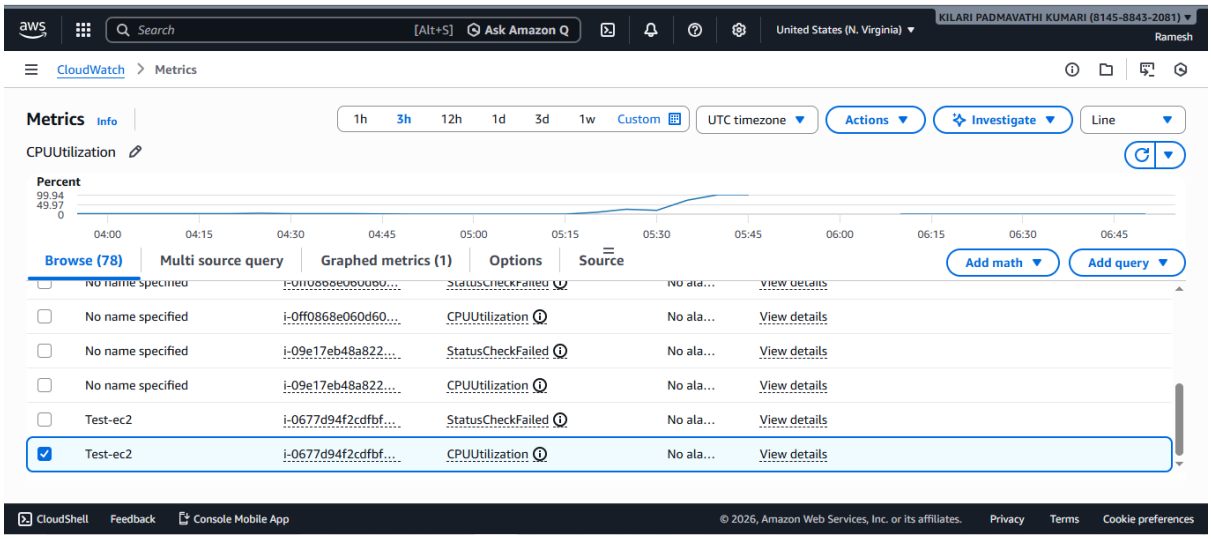
3. Choose **Per-Instance Metrics**
4. Select:

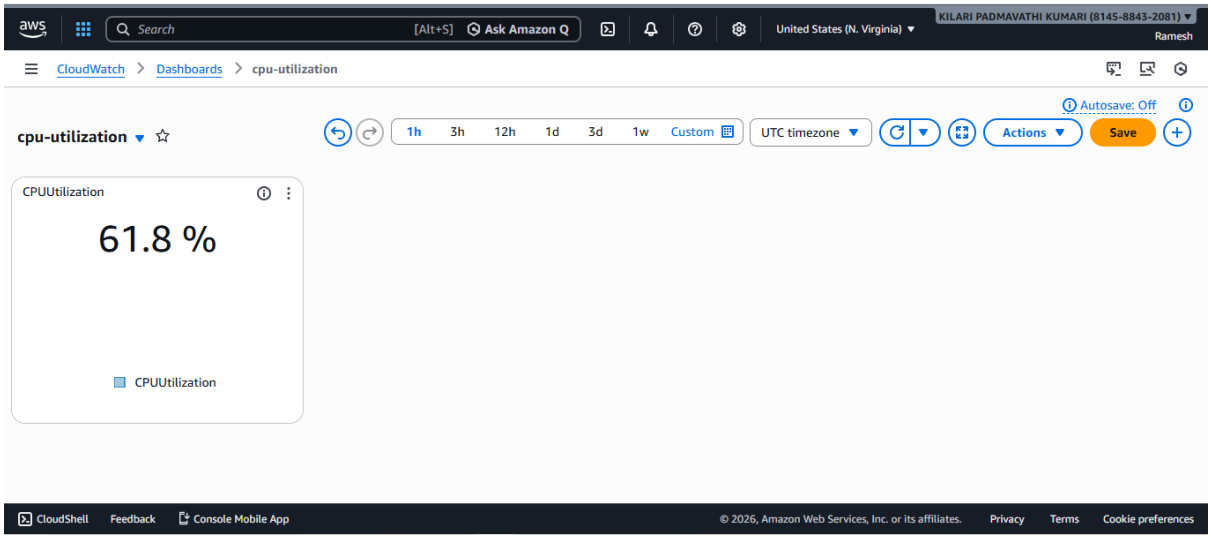
- CPUUtilization

5. Choose your instance

   You'll see real-time CPU graphs 📊

CloudTrail-CloudWatch-Task



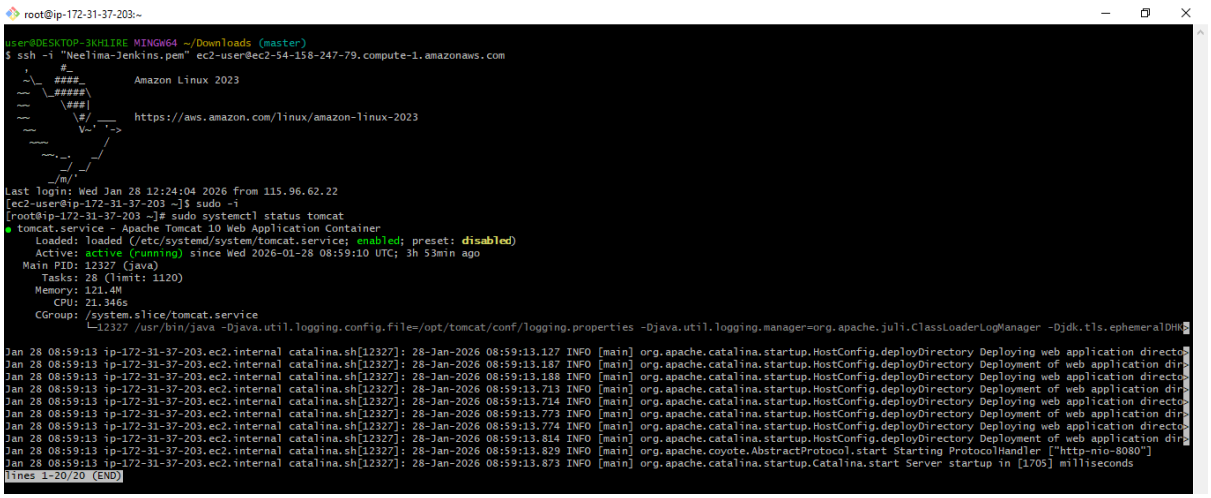


Conclusion:The above image shows Dashboard for CPUutilizations.

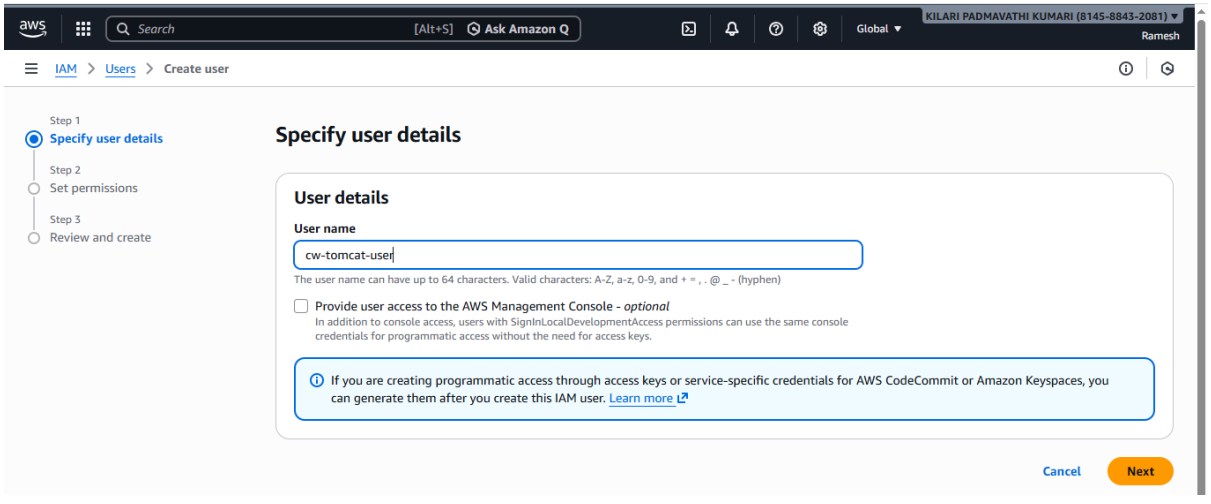
CloudTrail-CloudWatch-Task

5. Create a Dashboard and monitor the Tomcat service whether it is running or not and send the alert.

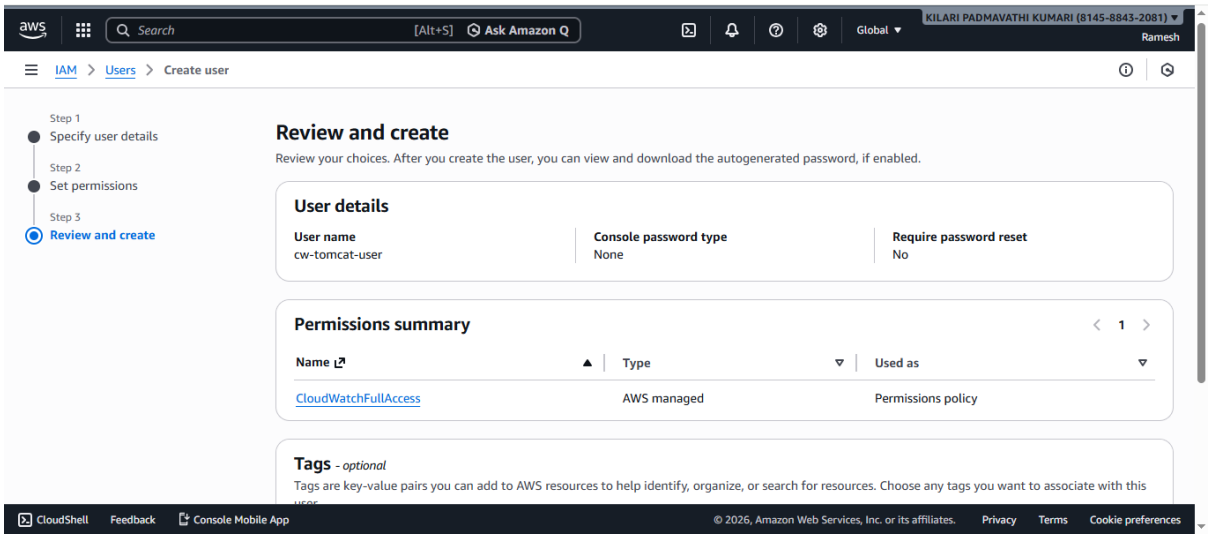
- Go AWS console and create instance
- Install tomcat in EC2
- Start tomcat by command sudo systemctl start tomcat.
- Check whether it is running or not.



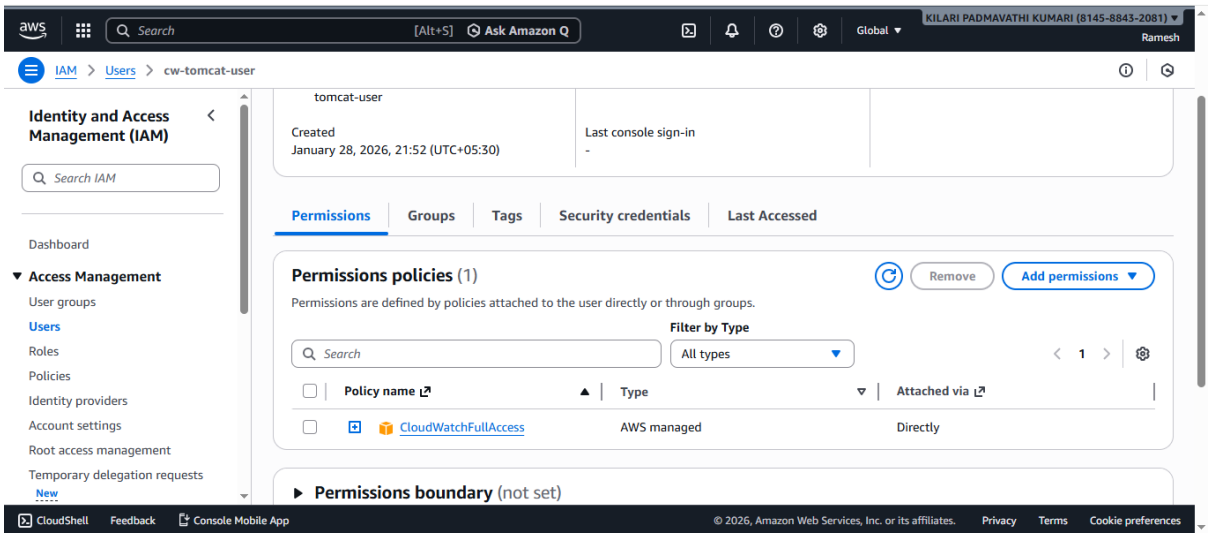
- Go to IAM
- Create user as CW-tomcat-user
- Click on Next.



CloudTrail-CloudWatch-Task



- Set permission as cloudwatchfullaccess



- Click on create user and verify permission policies.

```
user@DESKTOP-3KH1IRE MINGW64 ~/Downloads (master)
$ aws configure
AWS Access Key ID [****************YT7C]: AKIA33KKBB3I7IJYYT7C
AWS Secret Access Key [****************KSt9]: nRPm+hey+jw+yVoU2ISd7+frv6AgzscOTy
kgKSt9
Default region name [us-west-2]: us-east-1
Default output format [json]: json

user@DESKTOP-3KH1IRE MINGW64 ~/Downloads (master)
$ aws sts get-caller-identity

{
    "UserId": "AIDA33KKBB3I6AJPYKG5H",
    "Account": "814588432081",
    "Arn": "arn:aws:iam::814588432081:user/cw-tomcat-user"
}
```

- In EC2 we have to configure as Aws configure as shown above.

CloudTrail-CloudWatch-Task

```
user@DESKTOP-3KH1IRE MINGW64 ~/Downloads (master)
$ ssh -i "Neelima-Jenkins.pem" ec2-user@ec2-54-158-247-79.compute-1.amazonaws.com
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Wed Jan 28 12:52:51 2026 from 115.96.62.22
[ec2-user@ip-172-31-37-203 ~]$ sudo -i
[root@ip-172-31-37-203 ~]# aws configure
AWS Access Key ID [None]: AKIA33KKBB3I7IJYYT7C
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json
[root@ip-172-31-37-203 ~]# aws sts get-caller-identity
{
    "UserId": "AIDA33KKBB3I6AJPYKG5H",
    "Account": "814588432081",
    "Arn": "arn:aws:iam::814588432081:user/cw-tomcat-user"
}
[root@ip-172-31-37-203 ~]#
```

- Check add configuration by command
  Aws sts get-caller-identity

Create Tomcat Metric Script

- sudo vi /opt/tomcat_monitor.sh

root@ip-172-31-37-203:~

```
#!/bin/bash

if systemctl is-active --quiet tomcat; then
  aws cloudwatch put-metric-data \
    --namespace "TomcatService" \
    --metric-name "TomcatRunning" \
    --value 1
else
  aws cloudwatch put-metric-data \
    --namespace "TomcatService" \
    --metric-name "TomcatRunning" \
    --value 0
fi
~
~
~
```

Make executable:

sudo chmod +x /opt/tomcat_monitor.sh

CloudTrail-CloudWatch-Task

Test Script Manually

sudo /opt/tomcat_monitor.sh

```
[root@ip-172-31-37-203 ~]# sudo vi /opt/tomcat_monitor.sh
[root@ip-172-31-37-203 ~]# sudo chmod +x /opt/tomcat_monitor.sh
[root@ip-172-31-37-203 ~]# sudo /opt/tomcat_monitor.sh
[root@ip-172-31-37-203 ~]#
```

No output = **SUCCESS**

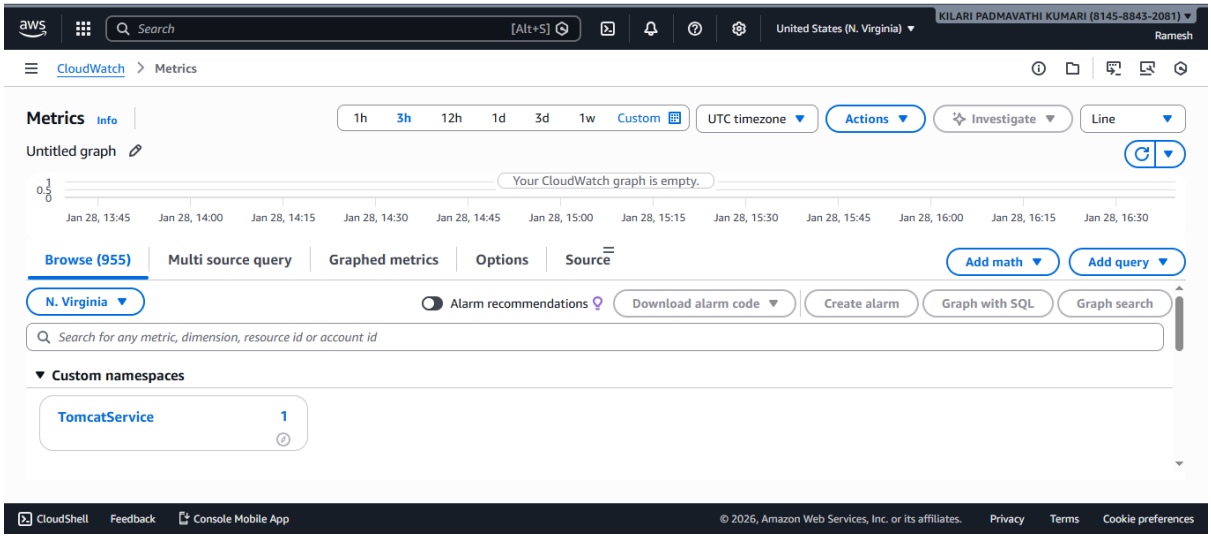
Add Cron Job

sudo crontab -e

Add:

***/1 * * * * /opt/tomcat_monitor.sh**

```
[root@ip-172-31-37-203 ~]# sudo crontab -e
crontab: installing new crontab
[root@ip-172-31-37-203 ~]# sudo crontab -l
*/1 * * * * /opt/tomcat_monitor.sh

[root@ip-172-31-37-203 ~]#
```



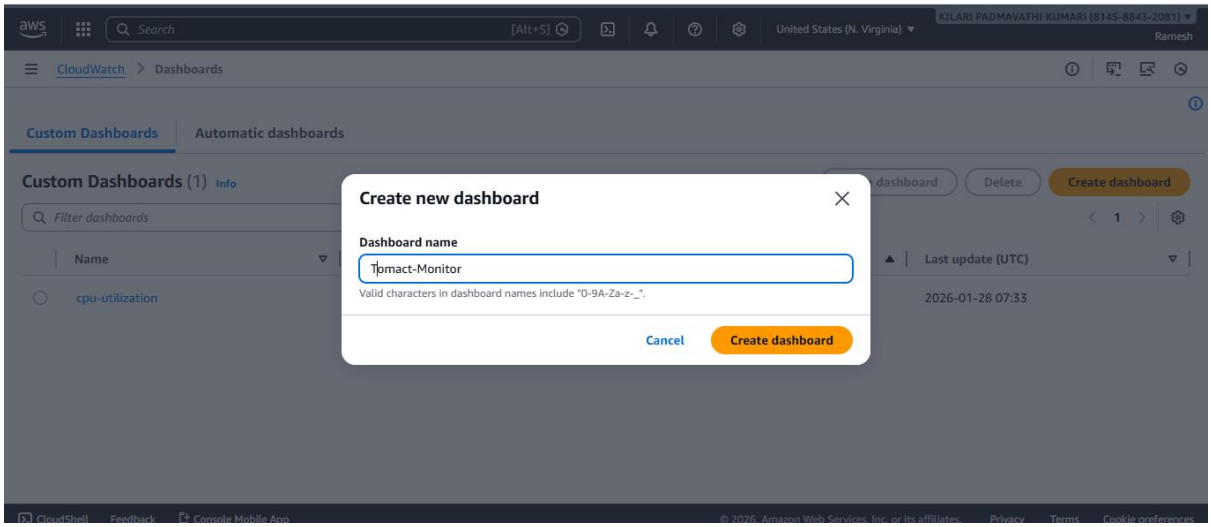
- The above image shows custom namespaces in that we have tomcat service as 1 it means it is running.

Create CloudWatch Dashboard

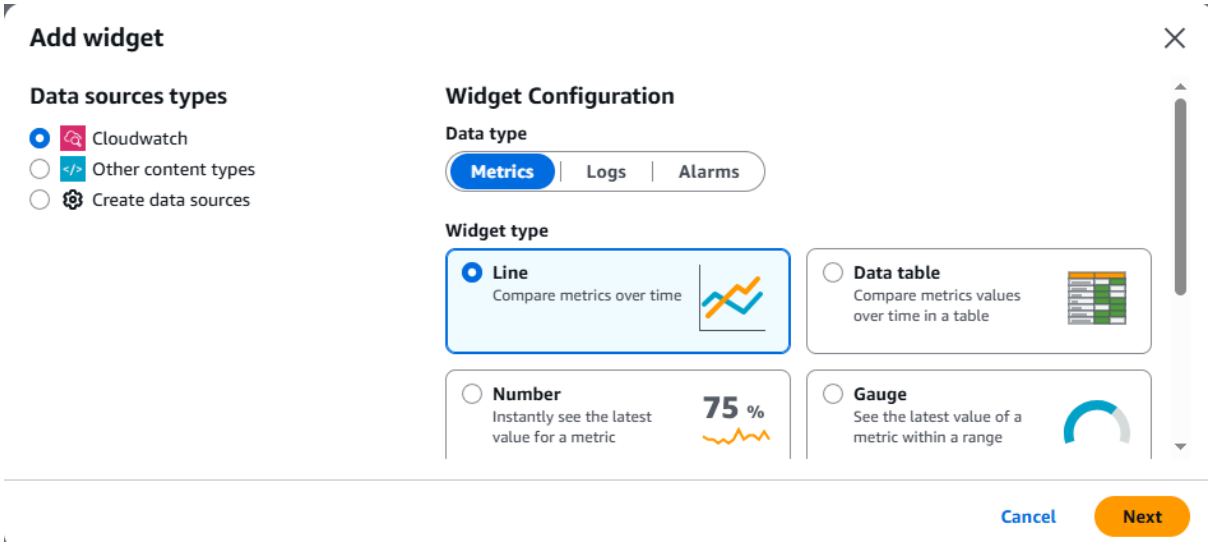
1. AWS Console → **CloudWatch**
2. Left menu → **Dashboards**
3. Click **Create dashboard**

CloudTrail-CloudWatch-Task

4. Name: Tomcat-Monitor
5. Click create dashboard.

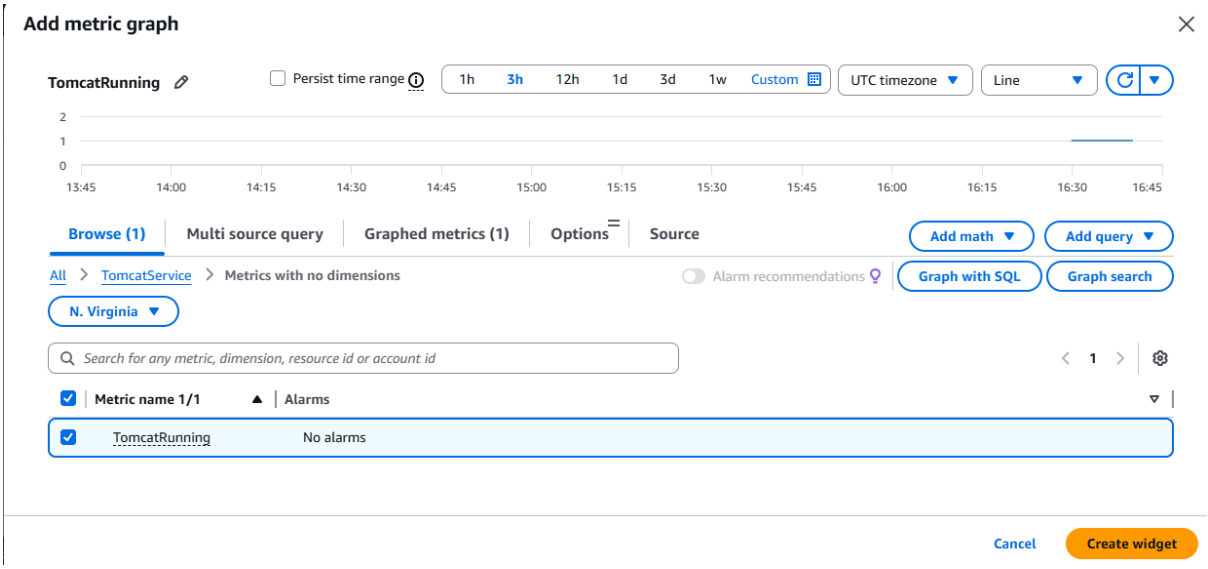


6. Click **Add widget**
7. Choose **Line**
8. Select:



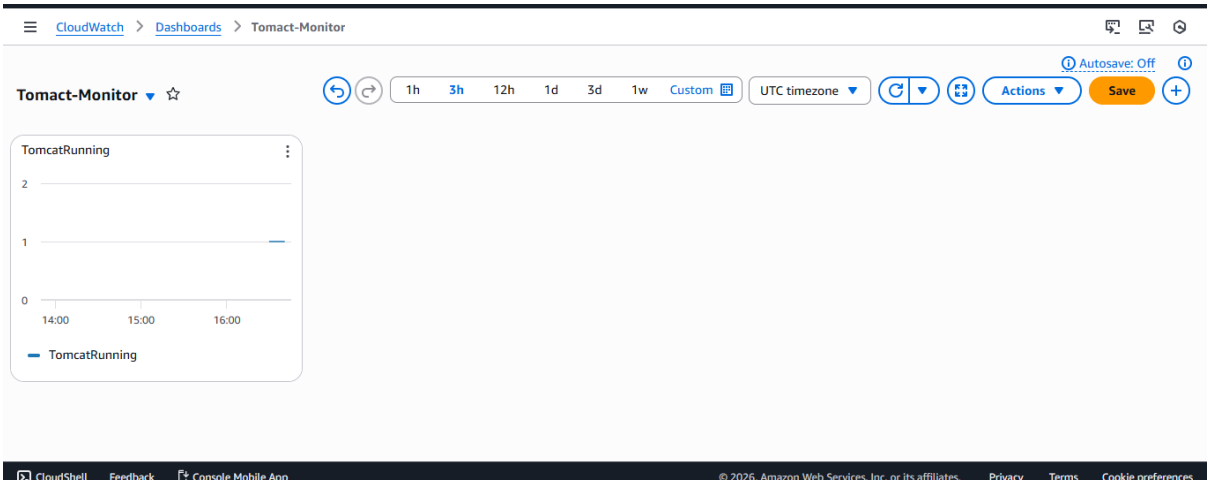
9. Custom namespaces → TomcatService → TomcatRunning
10. Click **Create widget**

CloudTrail-CloudWatch-Task



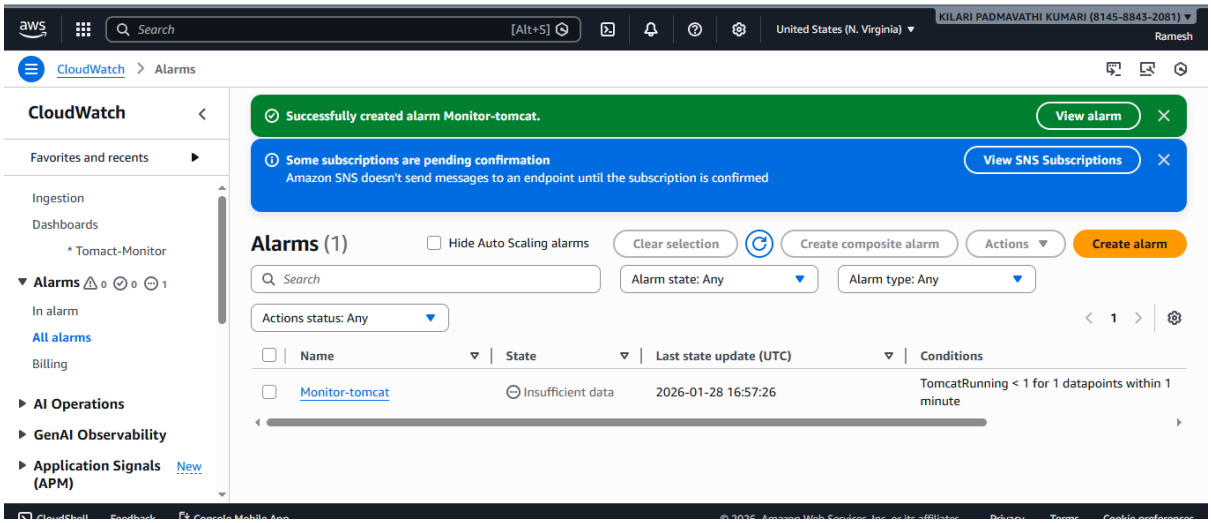
Now you'll see Tomcat status graph:

- 1 = Running
- 0 = Down

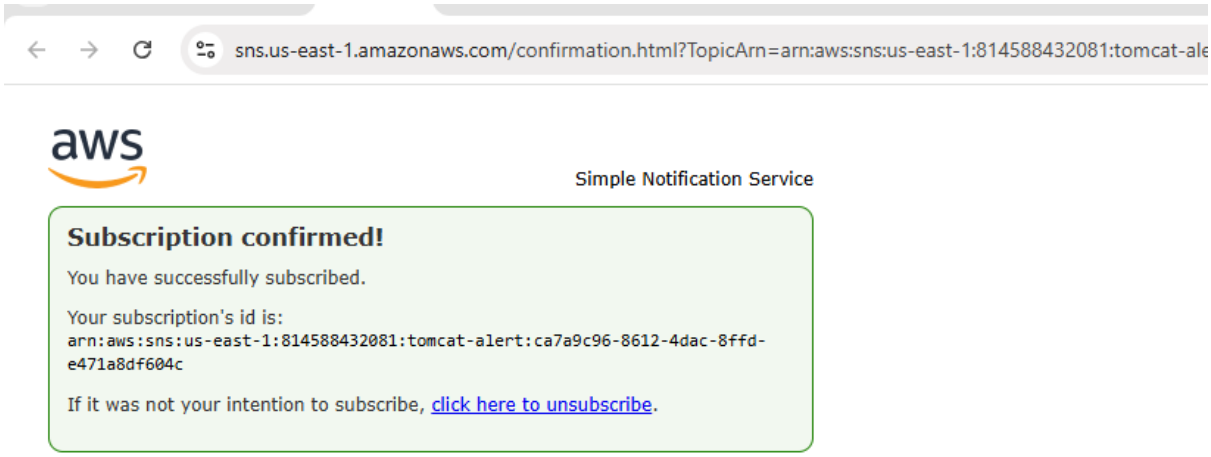


- **Create Alarm**
- CloudWatch → **Alarms** → **Create alarm**

CloudTrail-CloudWatch-Task

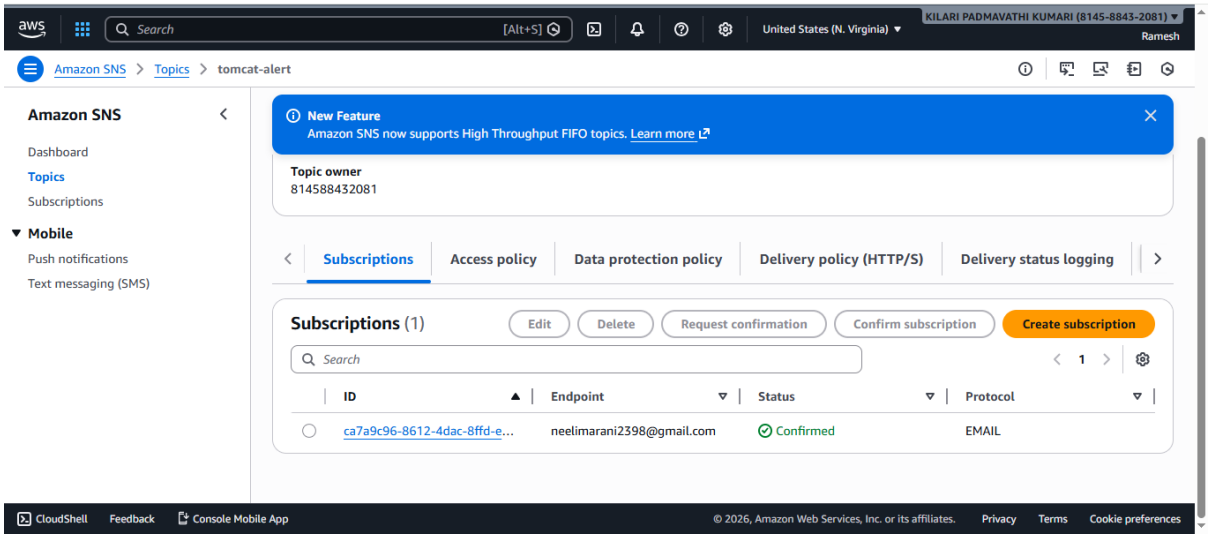


- Create SNS by creating topic

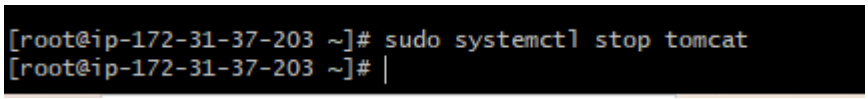


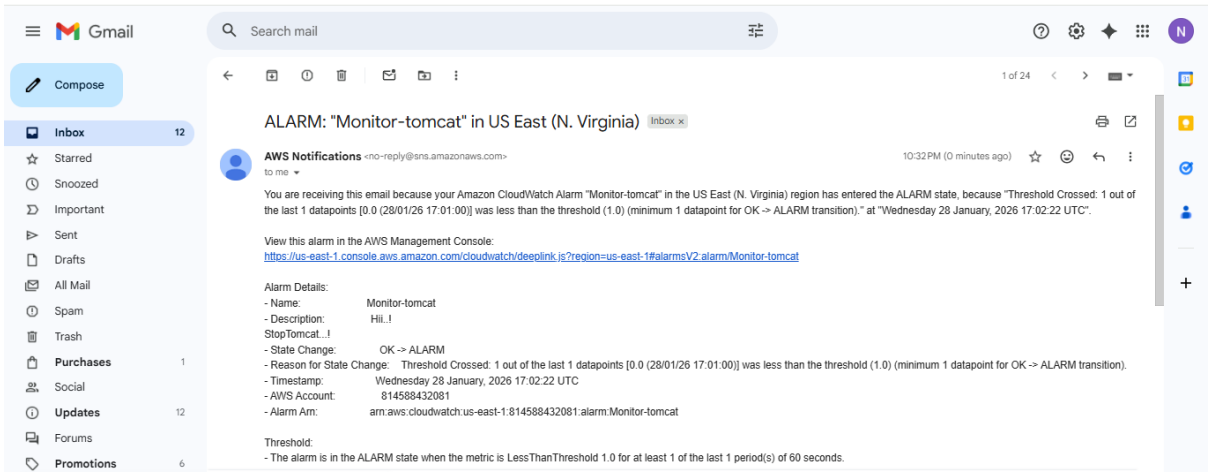
- Add subscription to confirm subscription.

CloudTrail-CloudWatch-Task



- To verify
- Stop tomcat service by sudo systemctl stop tomcat.





Conclusion: SNS trigger a Email which in created by cloudwatch alarm.

CloudTrail-CloudWatch-Task

6. Create a Dashboard and monitor the Nginx service to send the alert if Nginx is not running.

Monitor **Nginx service** on EC2 and:

- Push status to **CloudWatch Custom Metric**
- Show on **Dashboard**
- Send **Email alert** if Nginx stops

```
user@DESKTOP-3KH1IRE MINGW64 ~/Downloads (master)
$ ssh -i "Neelima-Jenkins.pem" ec2-user@ec2-54-87-109-74.compute-1.amazonaws.com
The authenticity of host 'ec2-54-87-109-74.compute-1.amazonaws.com (54.87.109.74)' can't be established.
ED25519 key fingerprint is SHA256:jMXHPDUx2GIYIL34hgKBQEyWY3CMIXGjw/Puf8g7KUY.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-87-109-74.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-40-146 ~]$
```

Step 1: Install & Start Nginx (if not already)

sudo yum install -y nginx

sudo systemctl start nginx

sudo systemctl enable nginx

Check:

systemctl status nginx

```
Complete!
[root@ip-172-31-40-146 ~]# sudo systemctl start nginx
[root@ip-172-31-40-146 ~]#
[root@ip-172-31-40-146 ~]# sudo systemctl enable nginx
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr/lib/systemd/system/nginx.service.
[root@ip-172-31-40-146 ~]# systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
     Active: active (running) since Wed 2026-01-28 17:13:40 UTC; 53s ago
   Main PID: 26159 (nginx)
      Tasks: 2 (limit: 1120)
     Memory: 2.5M
        CPU: 59ms
     CGroup: /system.slice/nginx.service
             ├─26159 "nginx: master process /usr/sbin/nginx"
             └─26163 "nginx: worker process"

Jan 28 17:13:40 ip-172-31-40-146.ec2.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Jan 28 17:13:40 ip-172-31-40-146.ec2.internal nginx[26136]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Jan 28 17:13:40 ip-172-31-40-146.ec2.internal nginx[26136]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Jan 28 17:13:40 ip-172-31-40-146.ec2.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[root@ip-172-31-40-146 ~]#
```

CloudTrail-CloudWatch-Task

Step 2: Create Nginx Metric Script

sudo vi /opt/nginx_monitor.sh

root@ip-172-31-40-146:~

```
#!/bin/bash

if systemctl is-active --quiet nginx; then
  aws cloudwatch put-metric-data \
    --namespace "NginxService" \
    --metric-name "NginxRunning" \
    --value 1
else
  aws cloudwatch put-metric-data \
    --namespace "NginxService" \
    --metric-name "NginxRunning" \
    --value 0
fi
~
```

Step 3: Test Script

```
[root@ip-172-31-40-146 ~]# sudo /opt/nginx_monitor.sh

Unable to locate credentials. You can configure credentials by running "aws login".
[root@ip-172-31-40-146 ~]# aws configure
AWS Access Key ID [None]: AKIA33KKBB3I7IJYYT7C
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json
[root@ip-172-31-40-146 ~]# aws sts get-caller-identity
{
    "UserId": "AIDA33KKBB3I6AJPYKG5H",
    "Account": "814588432081",
    "Arn": "arn:aws:iam::814588432081:user/cw-tomcat-user"
}
[root@ip-172-31-40-146 ~]# sudo /opt/nginx_monitor.sh
[root@ip-172-31-40-146 ~]#
```

Step 4: Add Cron Job

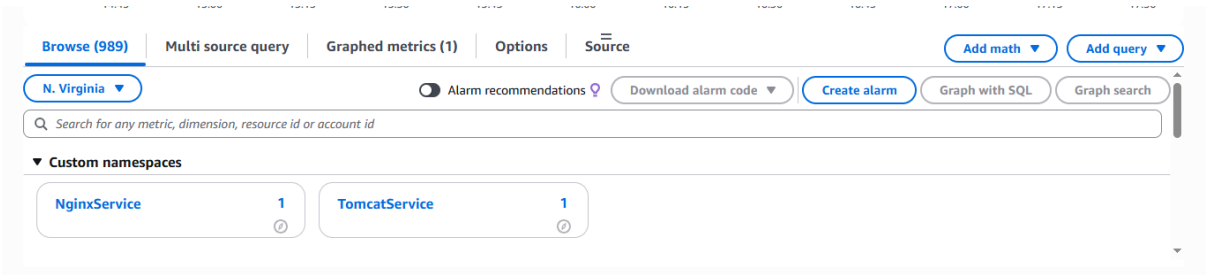
```
[root@ip-172-31-40-146 ~]# sudo crontab -e
no crontab for root - using an empty one
crontab: installing new crontab
[root@ip-172-31-40-146 ~]# sudo crontab -l
*/1 * * * * /opt/nginx_monitor.sh

[root@ip-172-31-40-146 ~]#
```

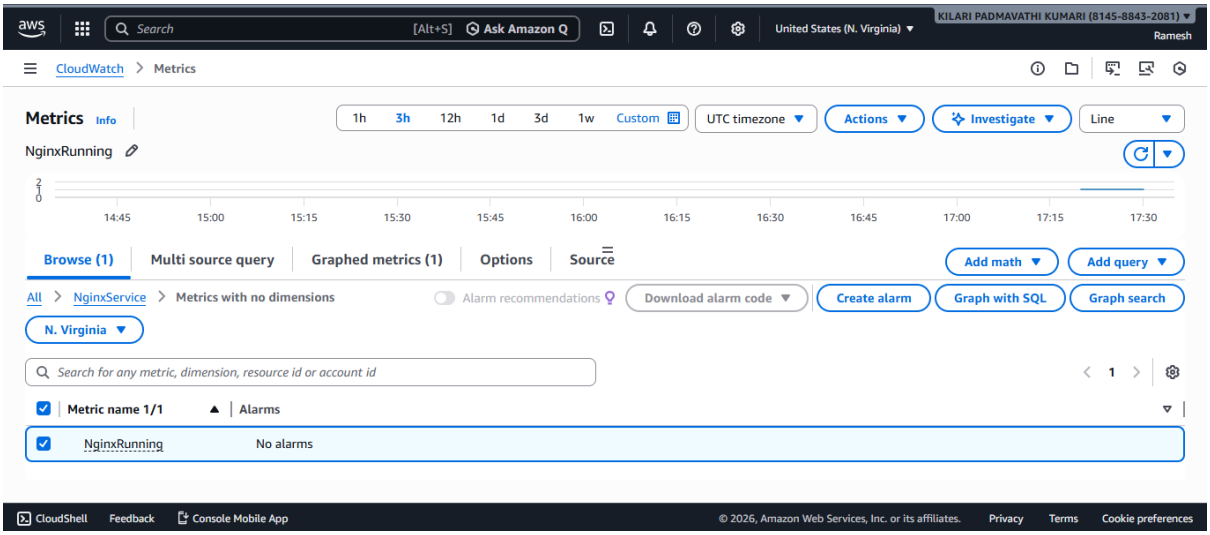
CloudTrail-CloudWatch-Task

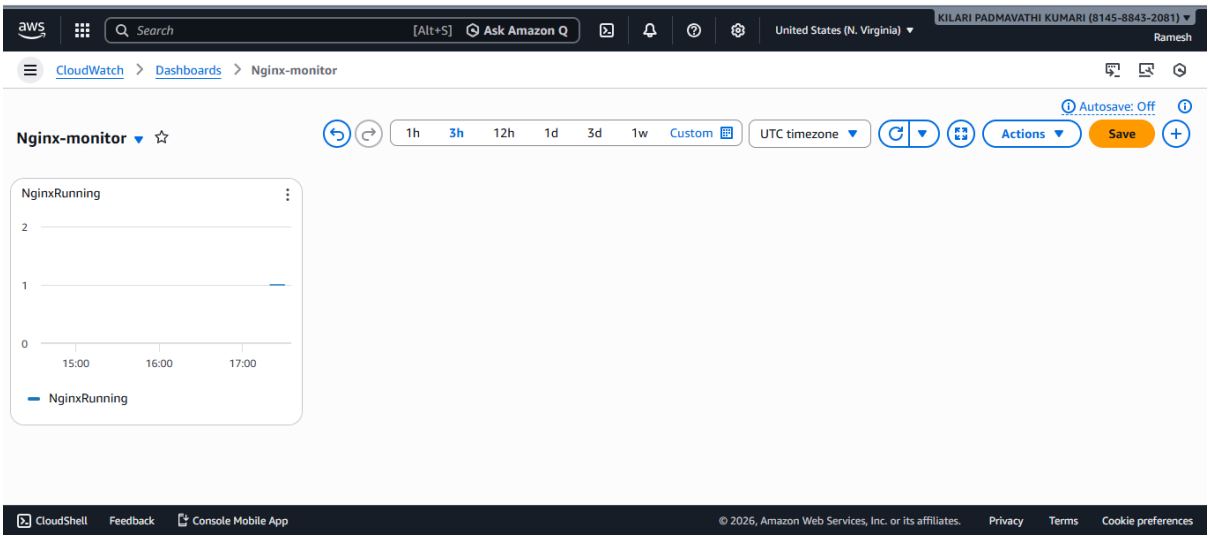
Step 5: Verify in CloudWatch

AWS Console → **CloudWatch → Metrics**

Custom namespaces → NginxService → NginxRunning

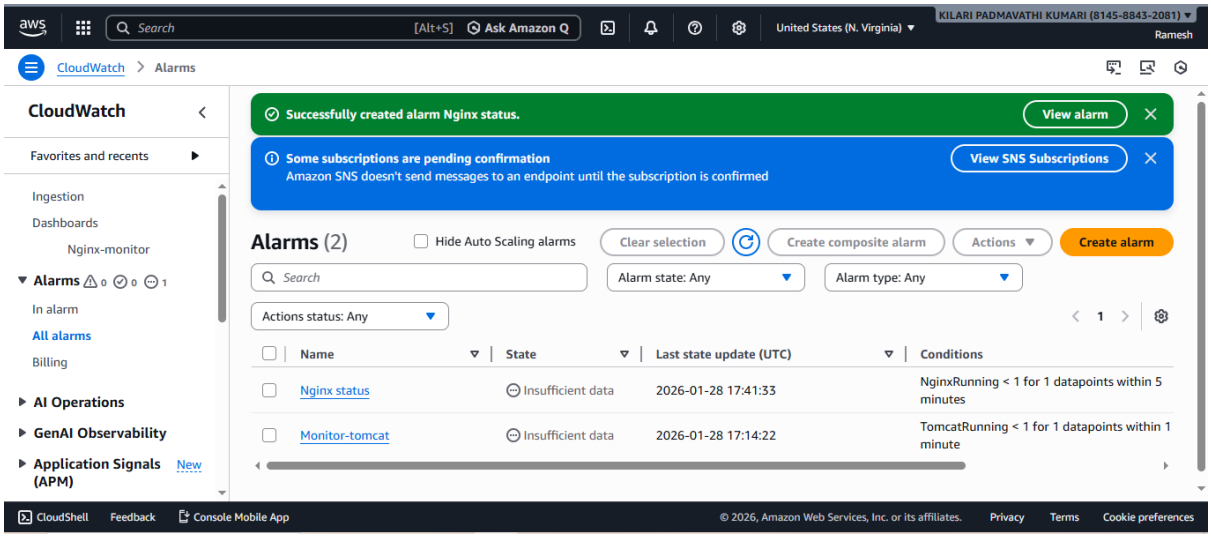




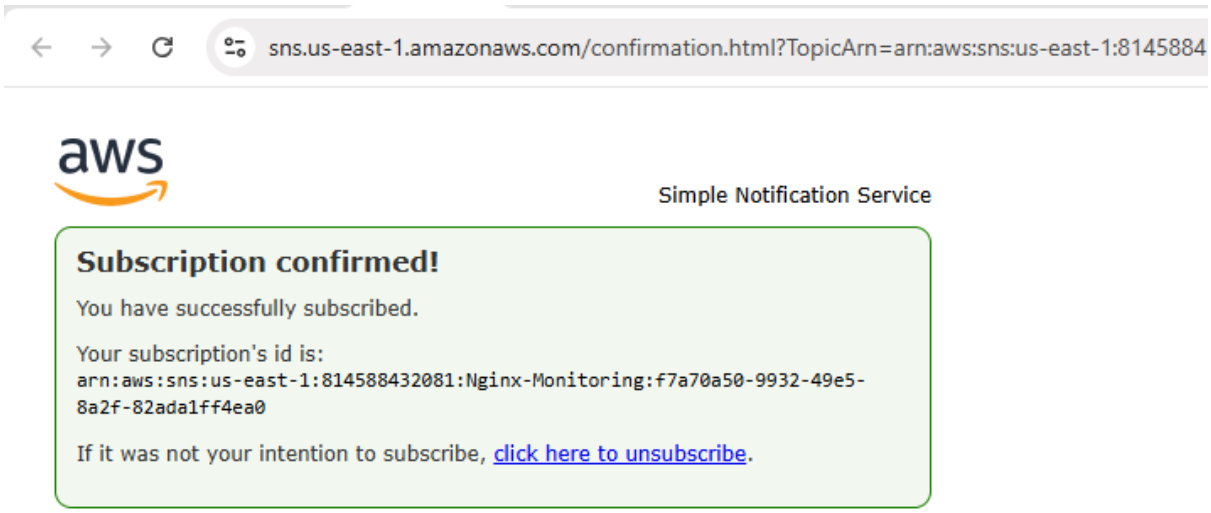


- Go Dashboard →Nginx Monitor
- Click on save.

CloudTrail-CloudWatch-Task



- Create Alarm for Nginx.
- Add SNS topic and confirm subscription.



Test

Stop nginx:

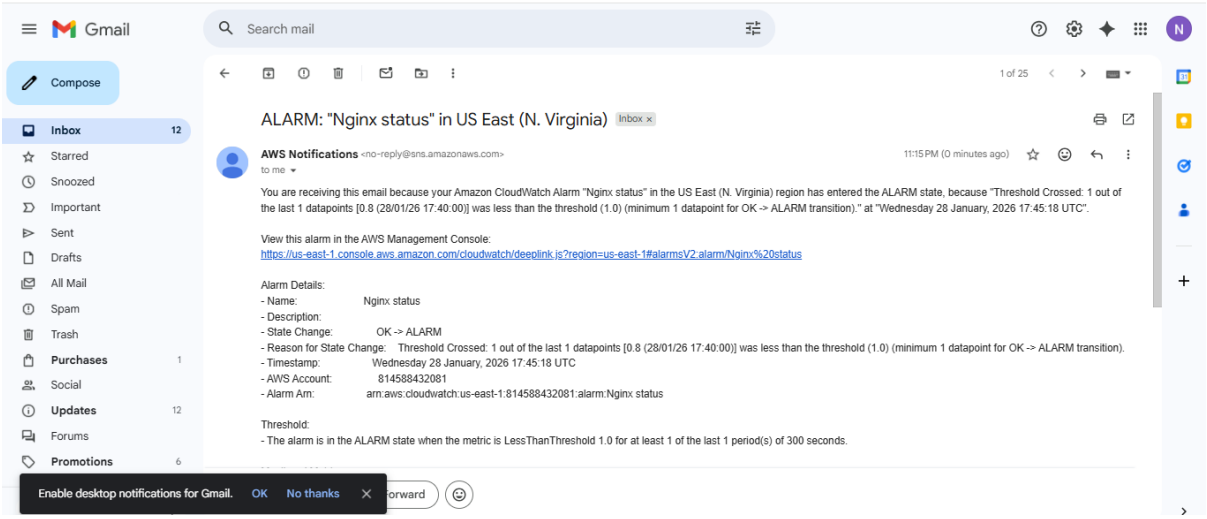
sudo systemctl stop nginx

```
[root@ip-172-31-40-146 ~]# sudo systemctl stop nginx
[root@ip-172-31-40-146 ~]#
```

- Alarm → **ALARM**

CloudTrail-CloudWatch-Task

- Email received



- Dashboard shows 0

Conclusion: Dashboard and monitor the Nginx service to send the alert if Nginx is not running.