**Task on Cloud Trail & Cloud Watch**

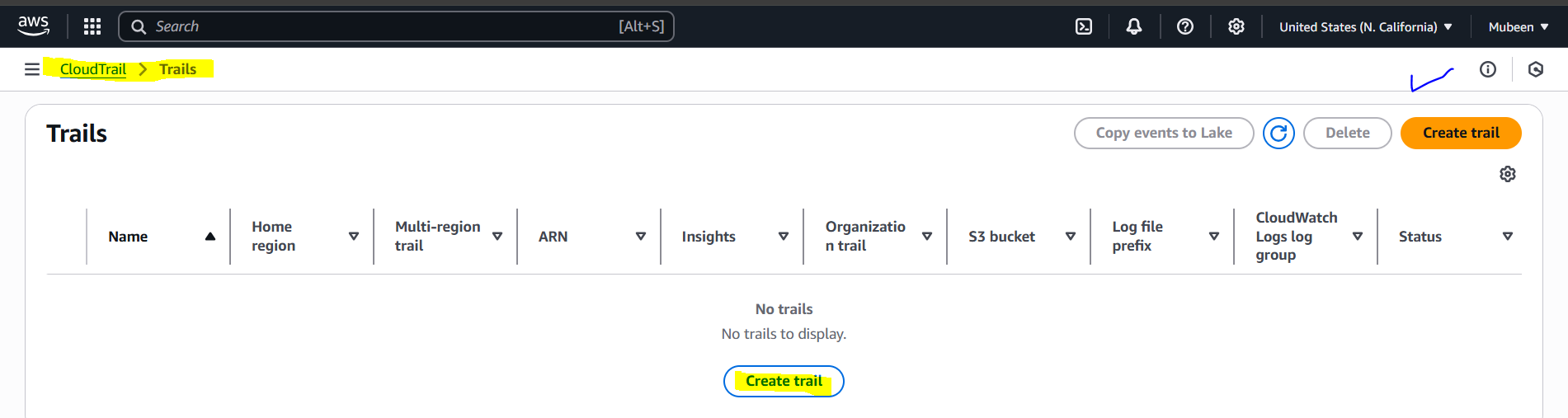
**1) Enable cloud trail monitoring and store the events in s3 and cloud watch log events.**

**Step 1: Create a CloudTrail**

1. Log in to the AWS Management Console.

2. Navigate to the CloudTrail dashboard.

3. Click "Create trail".

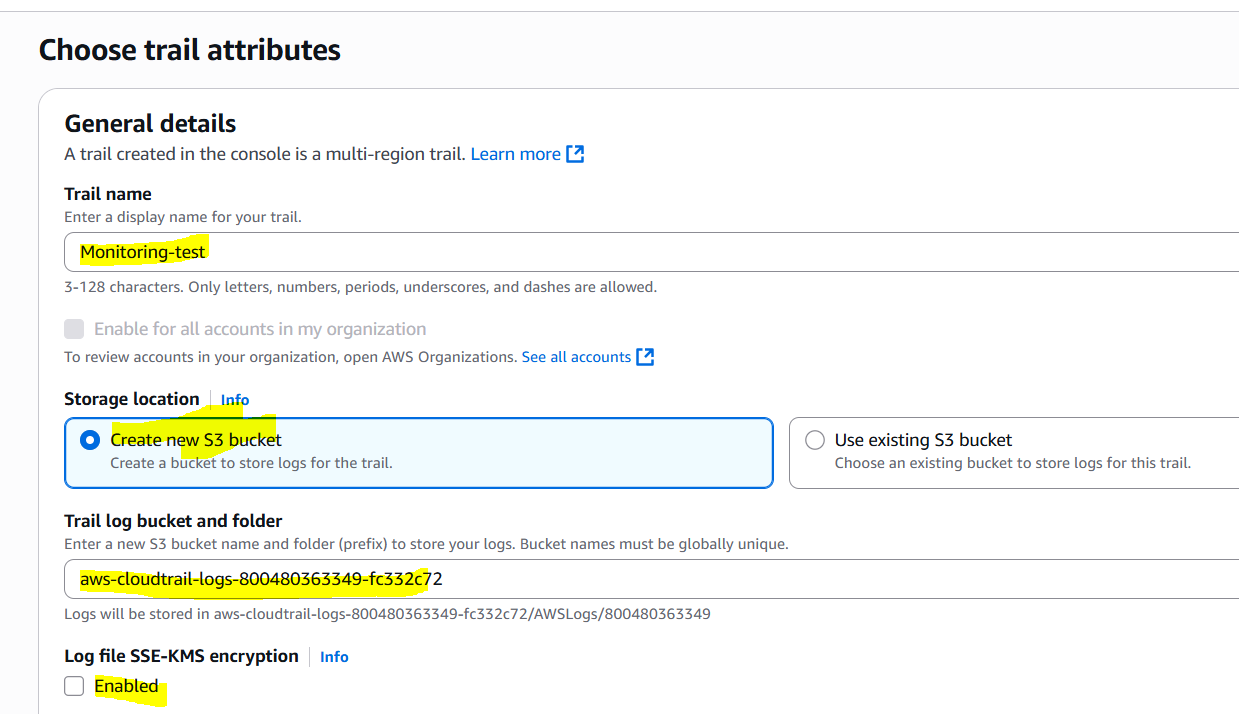


4. Enter a trail name.

5. Choose an S3 bucket:

- Select "Existing bucket" and choose a bucket, or

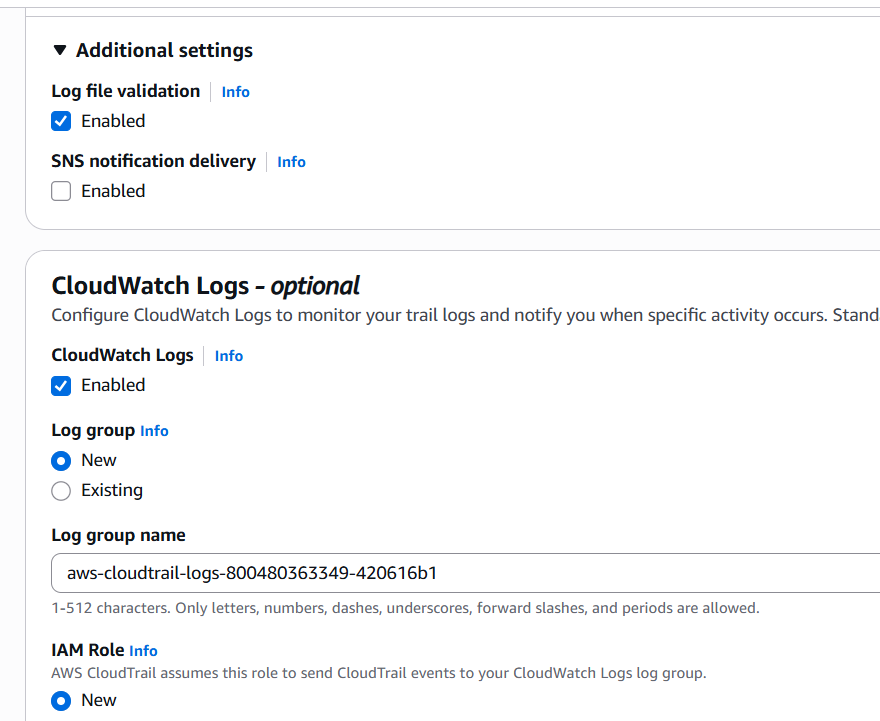
- Select "New bucket" and enter a unique bucket name.



6. Under "CloudWatch Logs", choose:

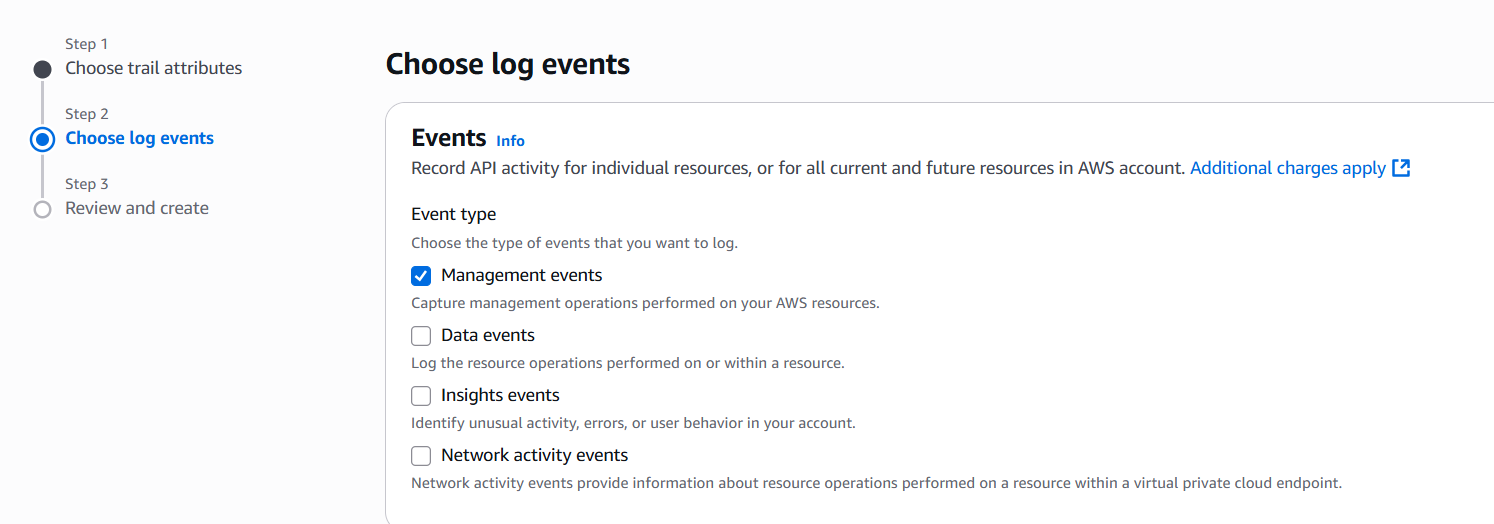
- "Existing log group" and select a log group, or

- "New log group" and enter a log group name.



7. Choose the events you want to log (e.g., management events, data events).

8. Review and Click "Create trail".

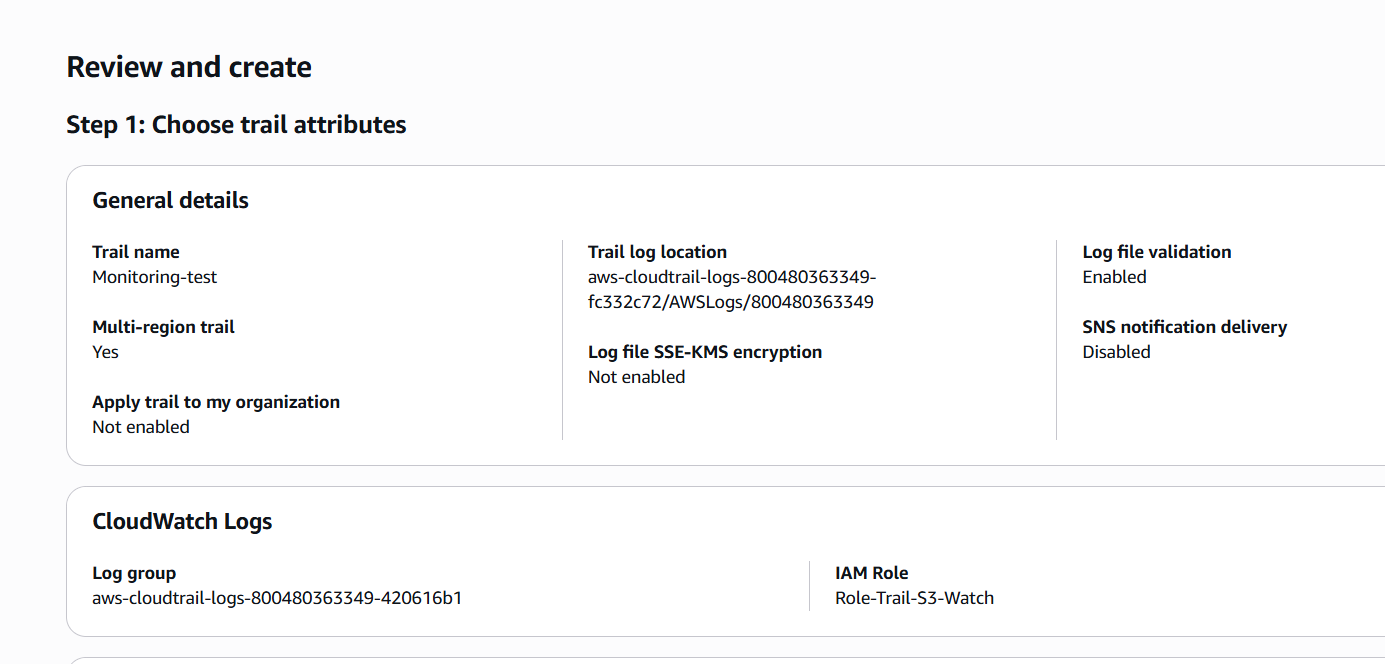


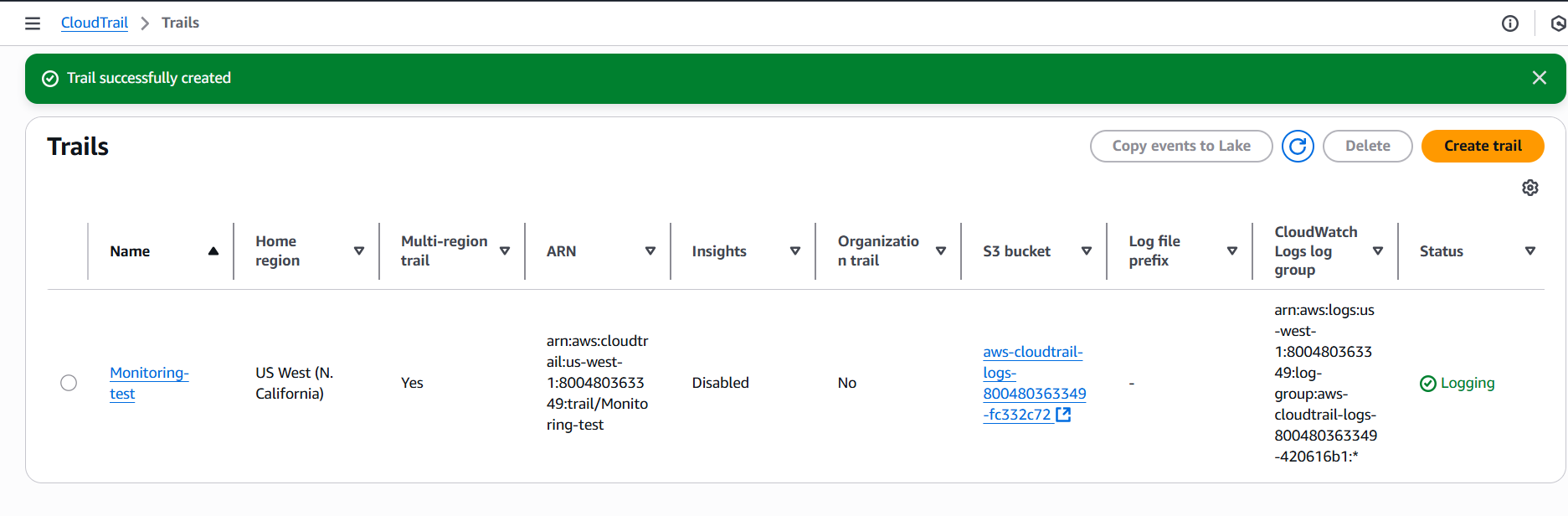
**Step 2: Configure CloudTrail Settings**

1. Review the trail settings.

2. Ensure that the S3 bucket and CloudWatch log group are correctly configured.

3. Click "Create trail" to finalize the creation.

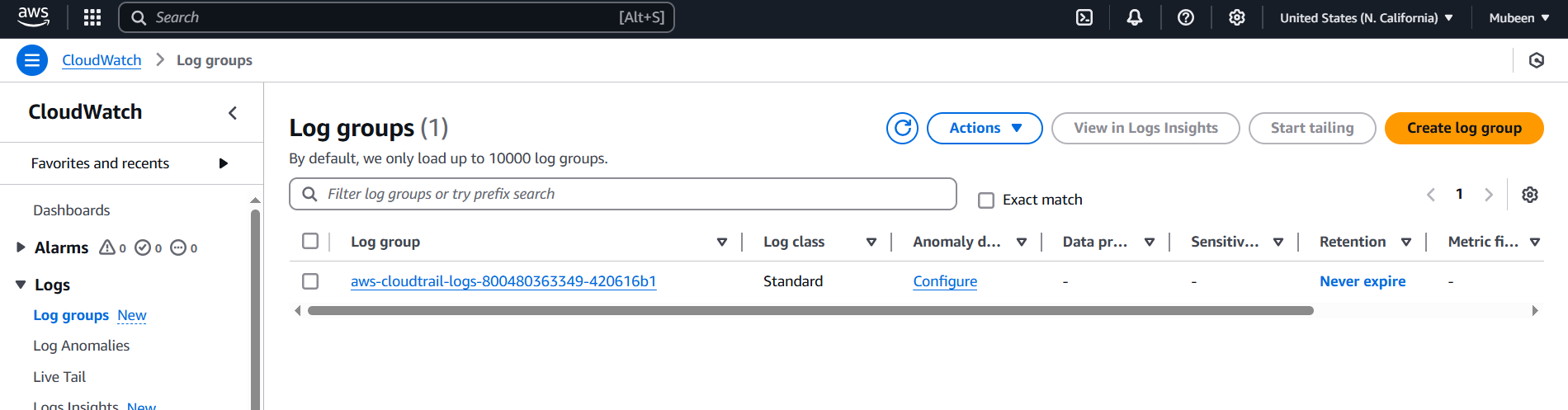


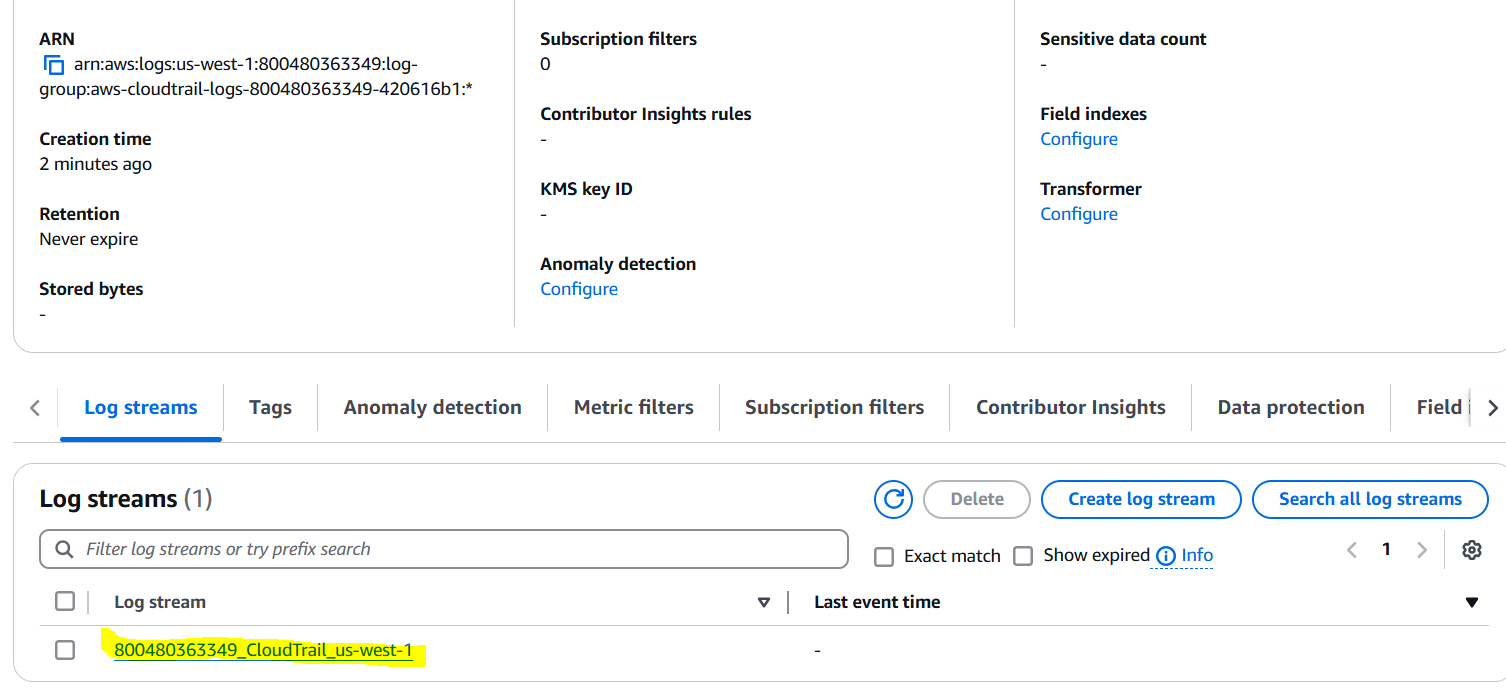


**Step 3: Verify CloudTrail and CloudWatch Configuration**

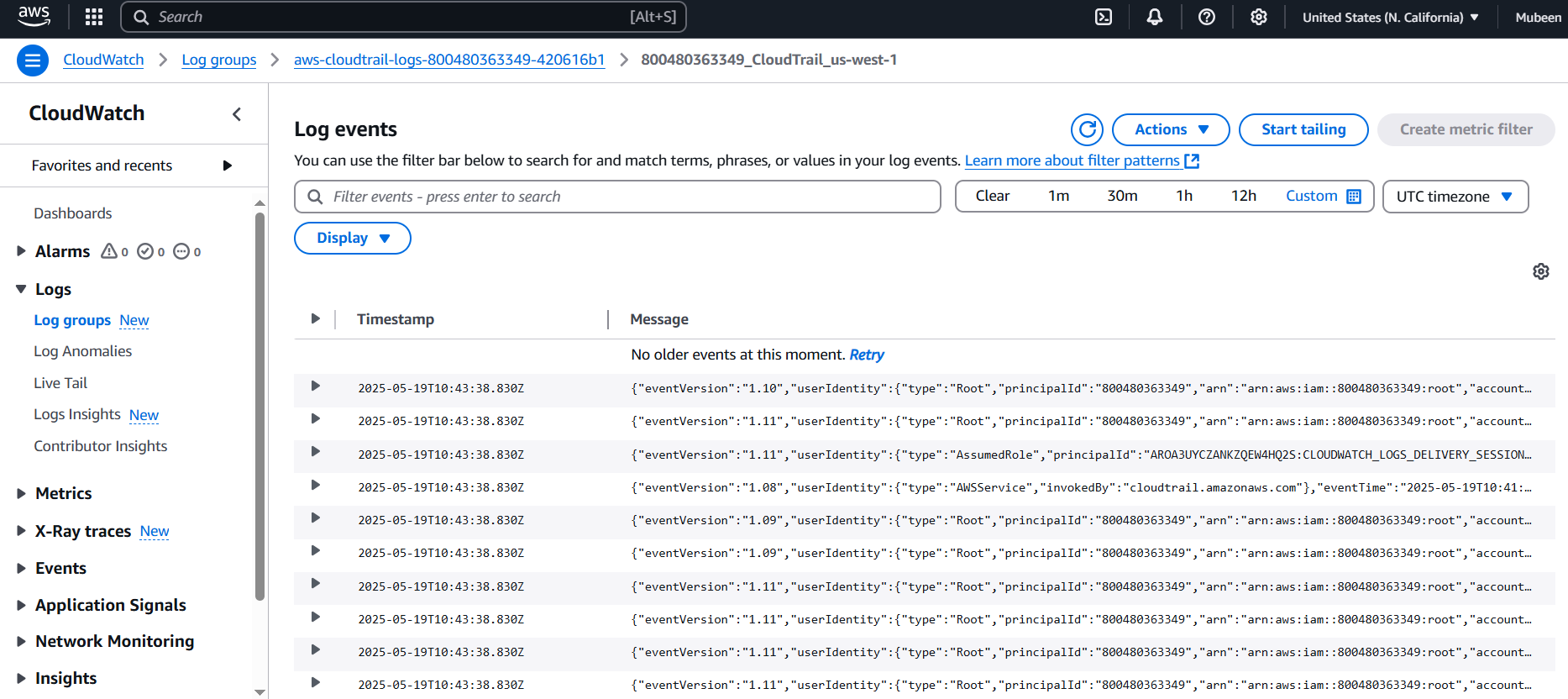
1. Navigate to the CloudWatch dashboard.

2. Select "Logs" and find the log group created for CloudTrail.





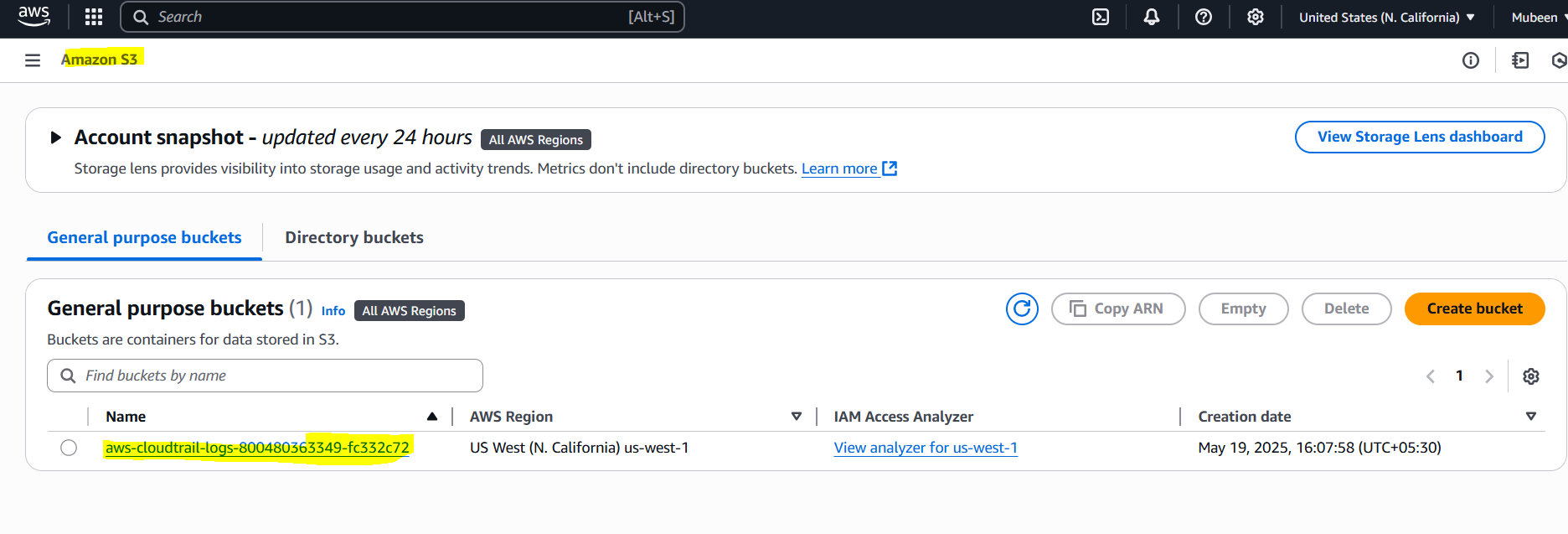
3. Verify that log events are being generated and stored in CloudWatch.

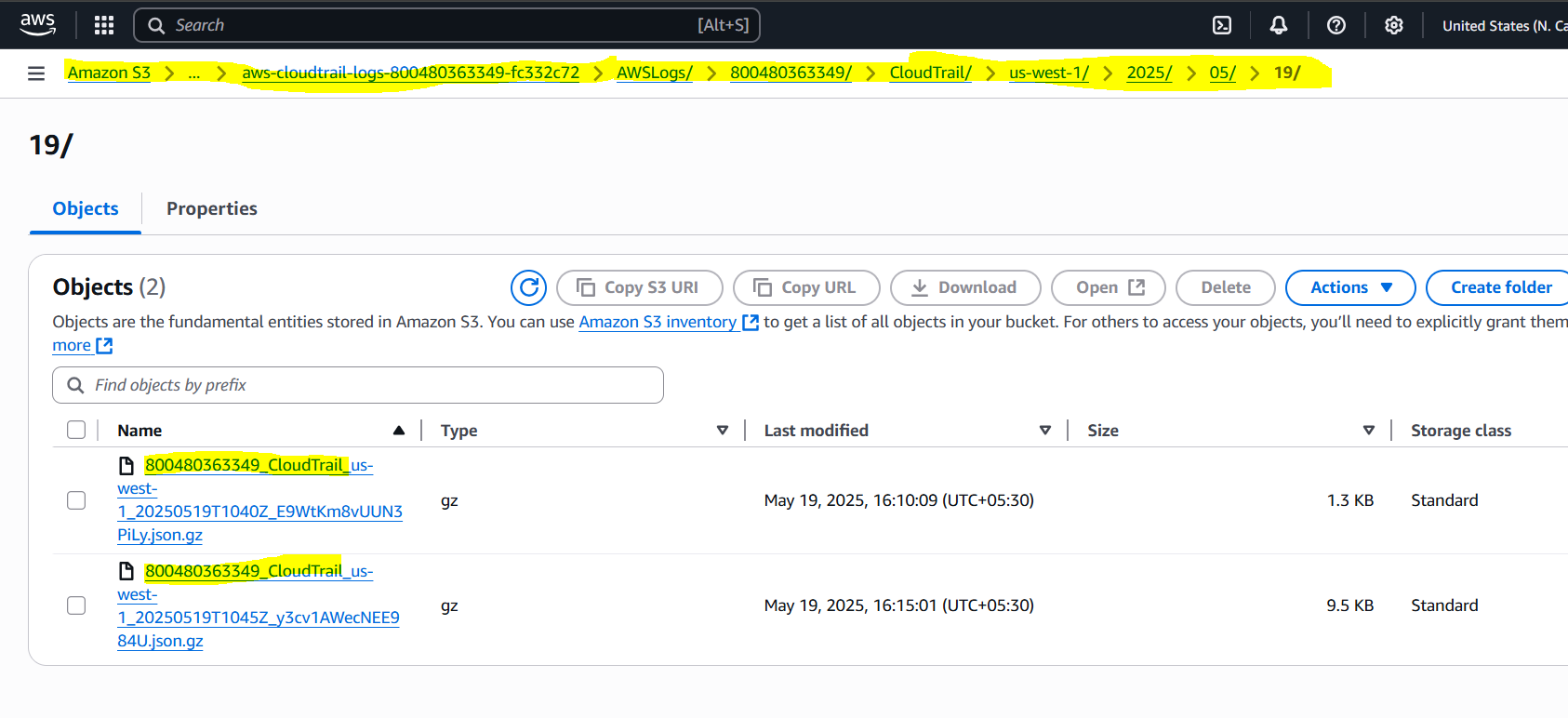


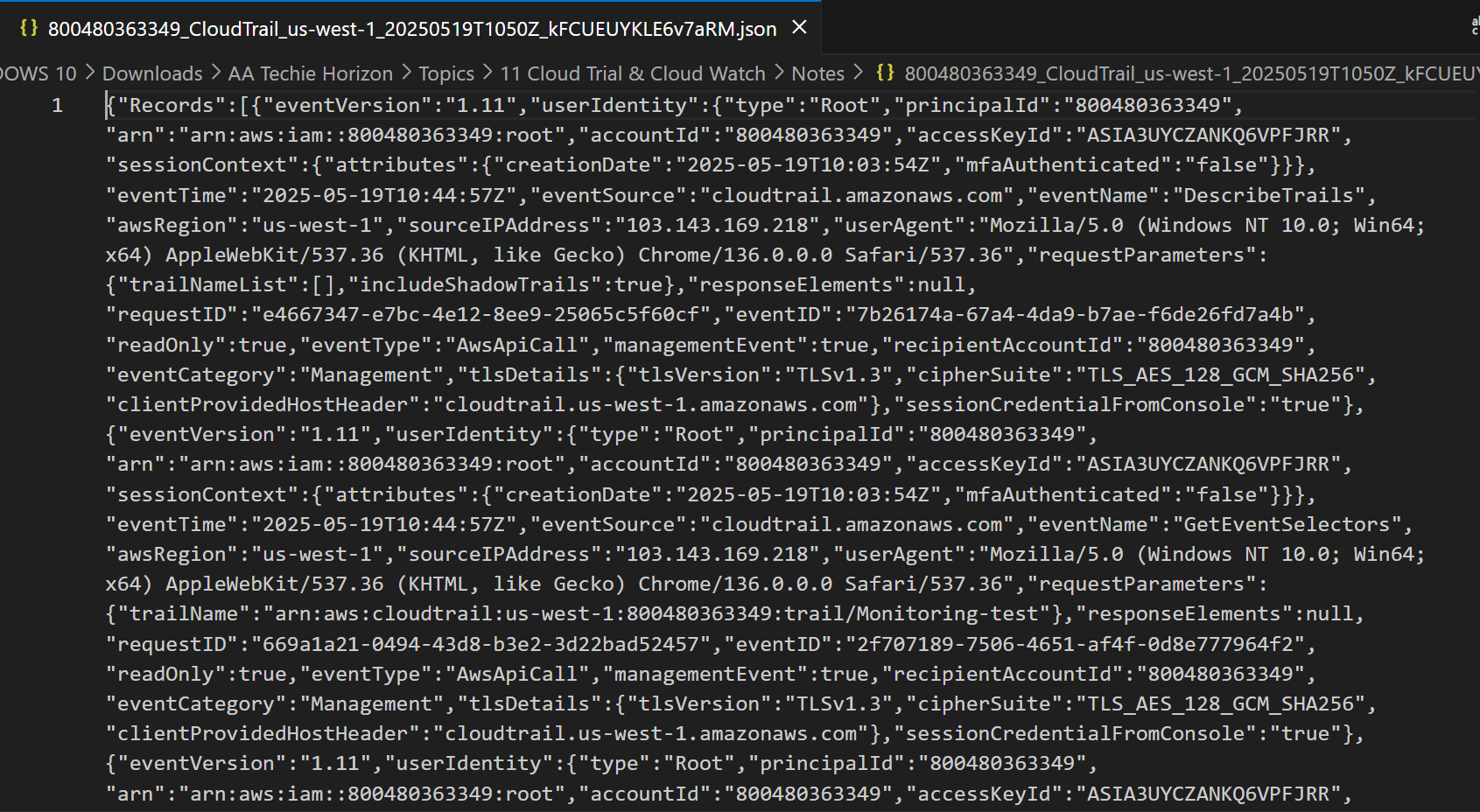
**Step 4: Verify S3 Bucket Logs**

1. Navigate to the S3 dashboard.

2. Select the bucket created for CloudTrail logs.



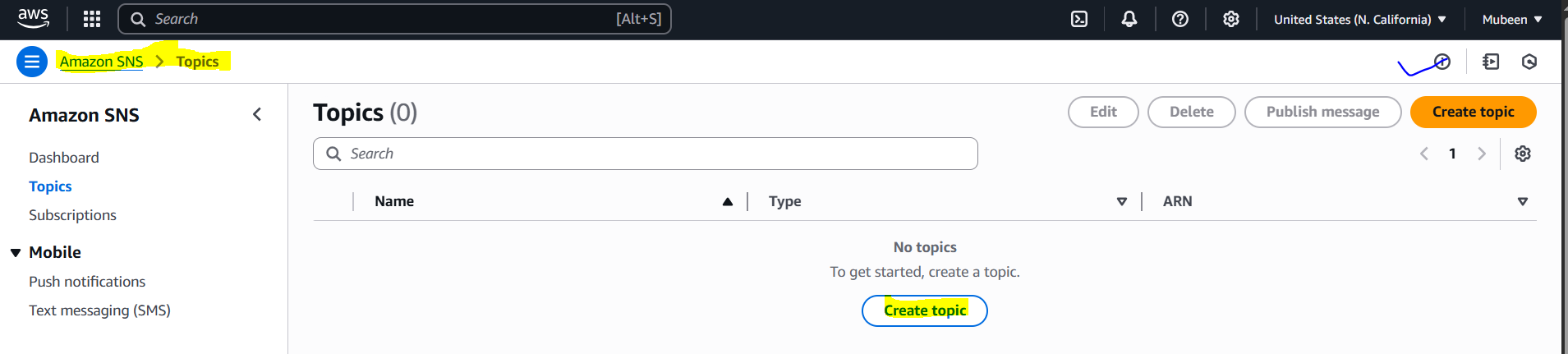
3. Verify that log files are being generated and stored in the S3 bucket.



**2) Enable SNS for cloudtrial to send alert on email.   
  
Step 1: Create an SNS Topic**

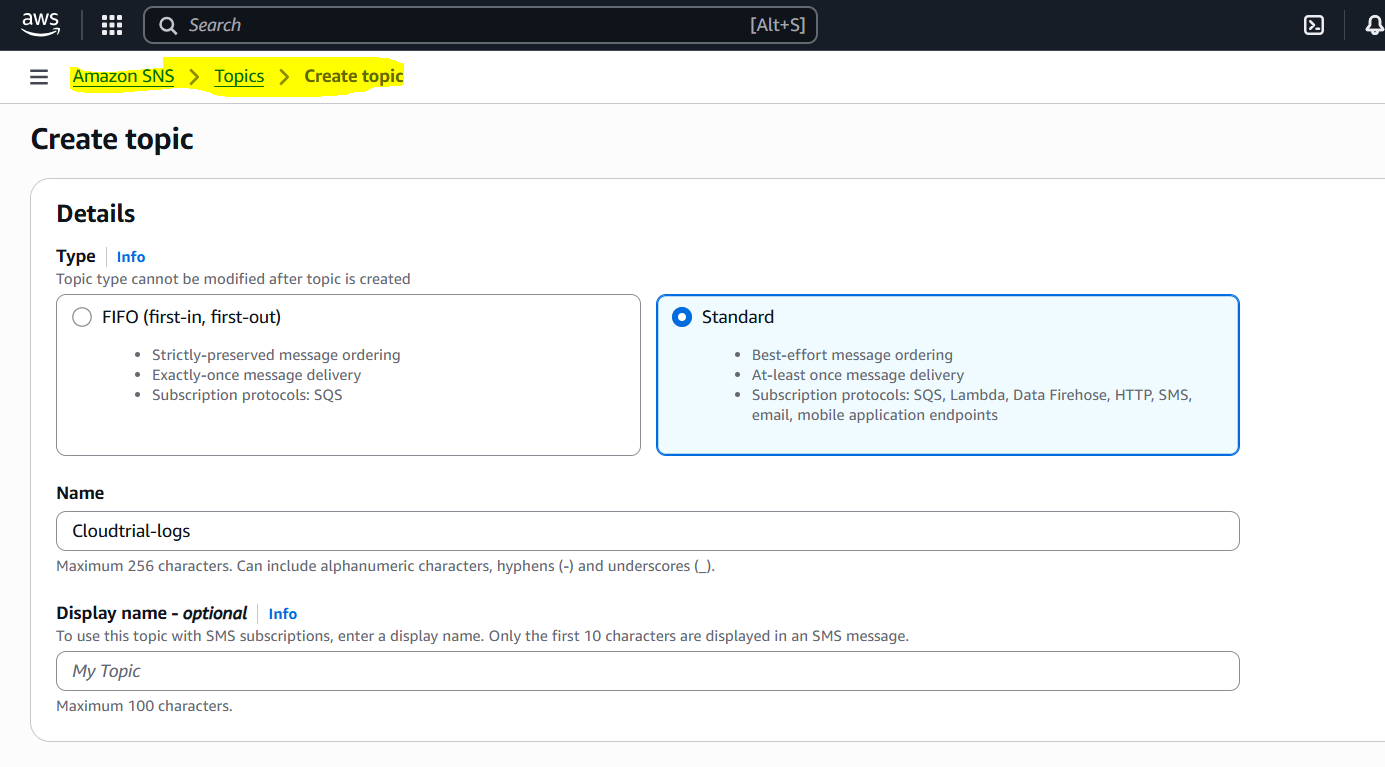
1. Navigate to the SNS dashboard.

2. Click "Create topic".

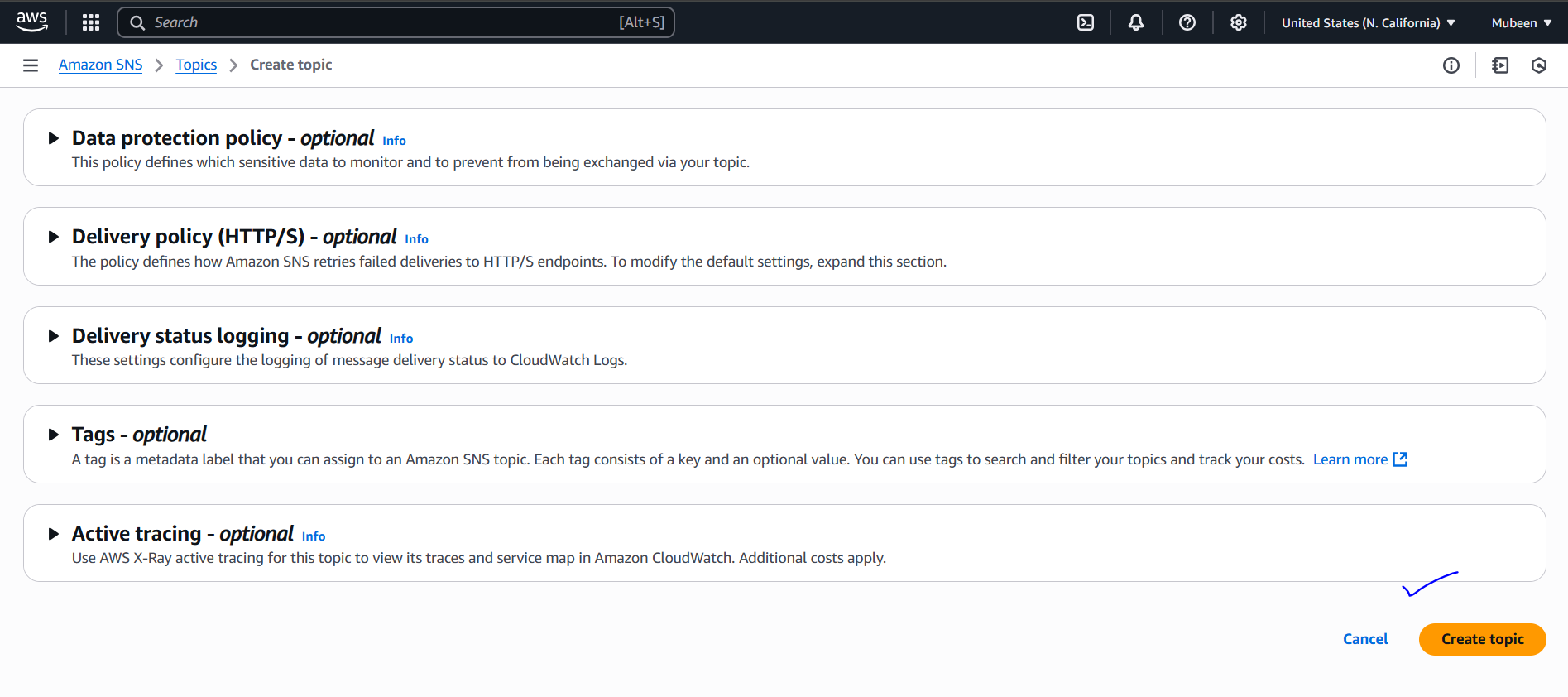


3. Choose "Standard" topic type.

4. Enter a topic name.



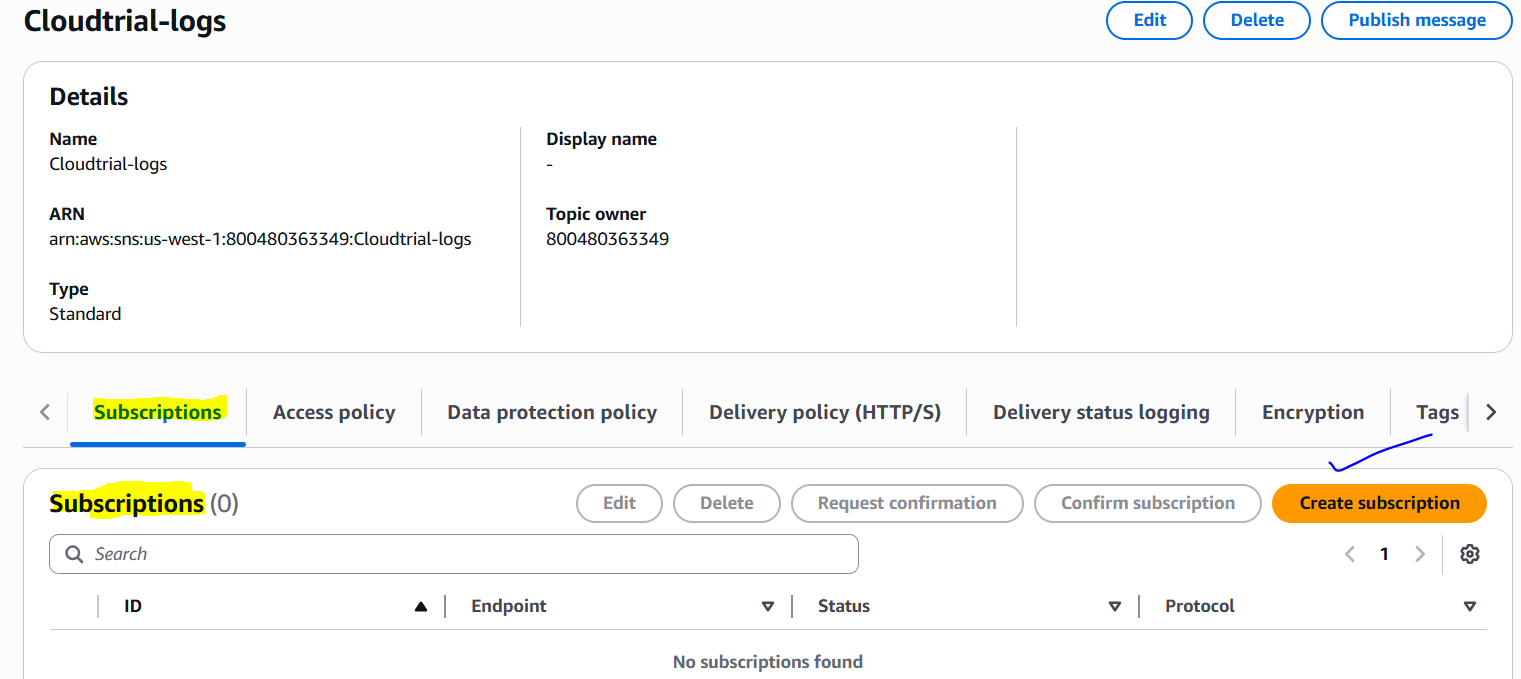
5. Click "Create topic".



**Step 2: Create an SNS Subscription**

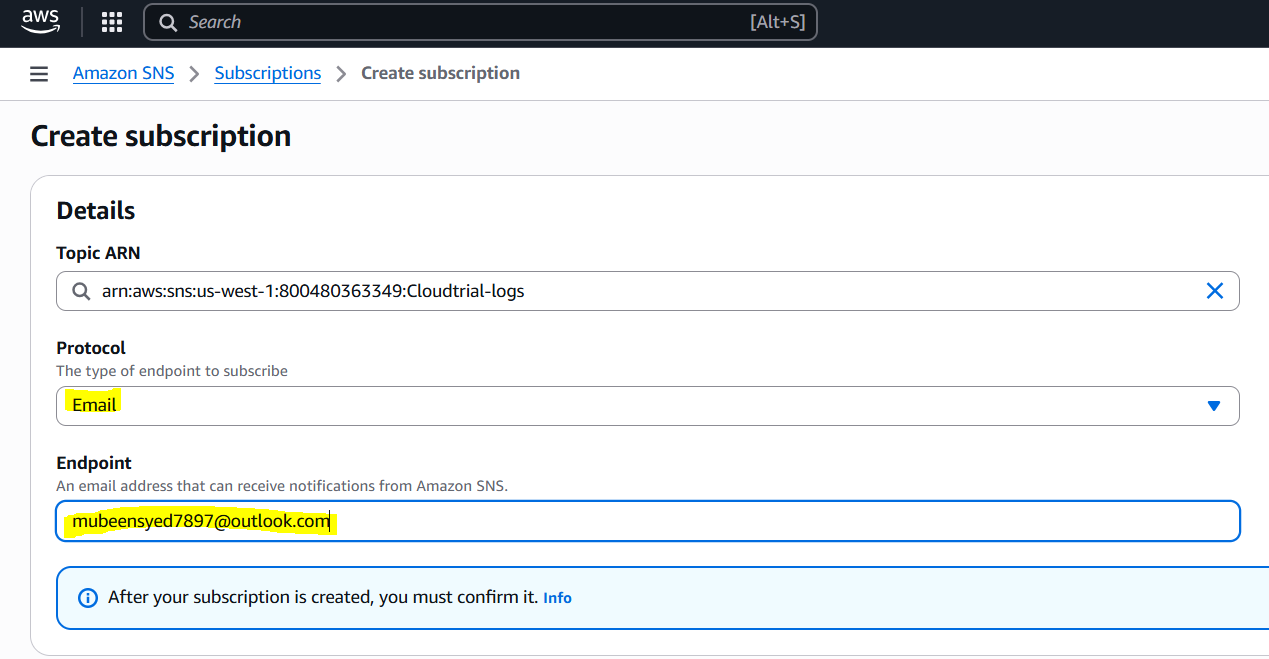
1. Select the SNS topic created in Step 1.

2. Click "Create subscription".

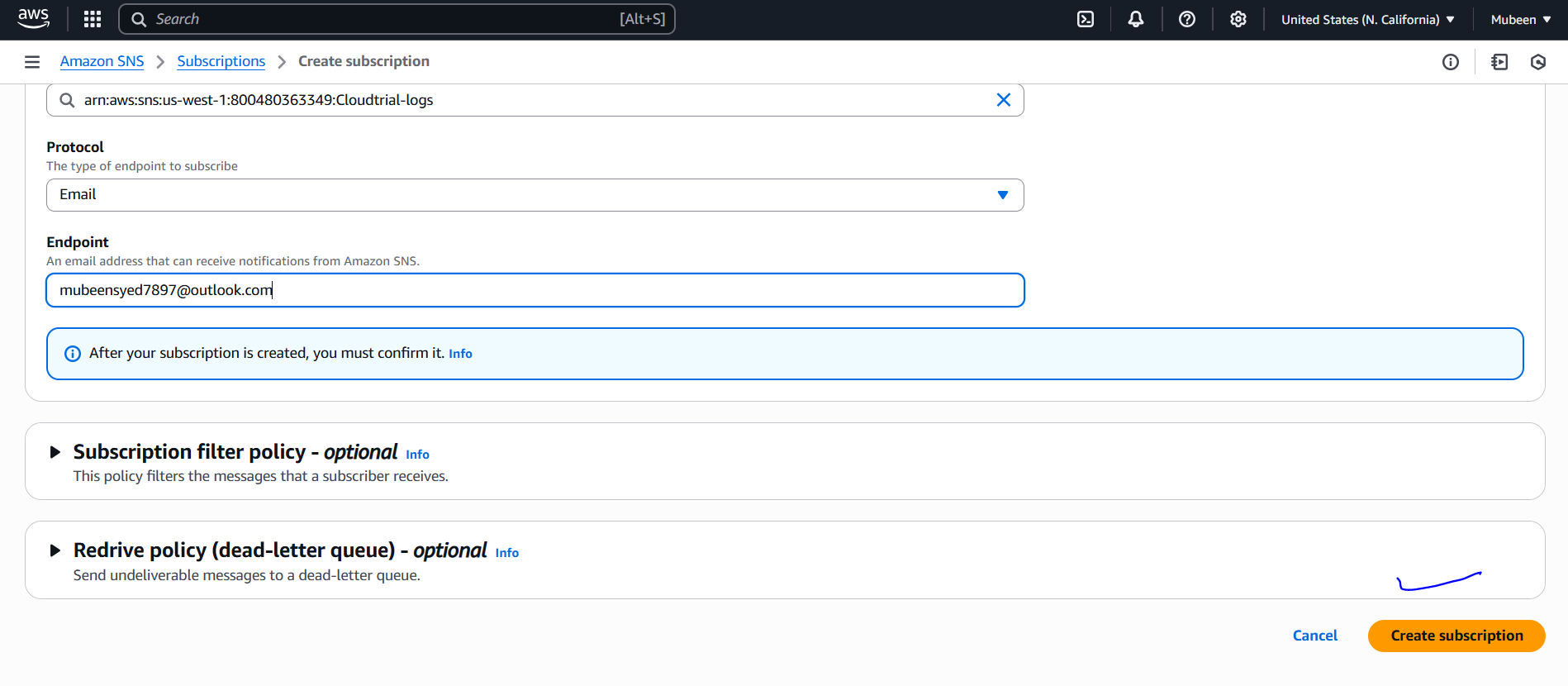


3. Choose "Email" as the protocol.

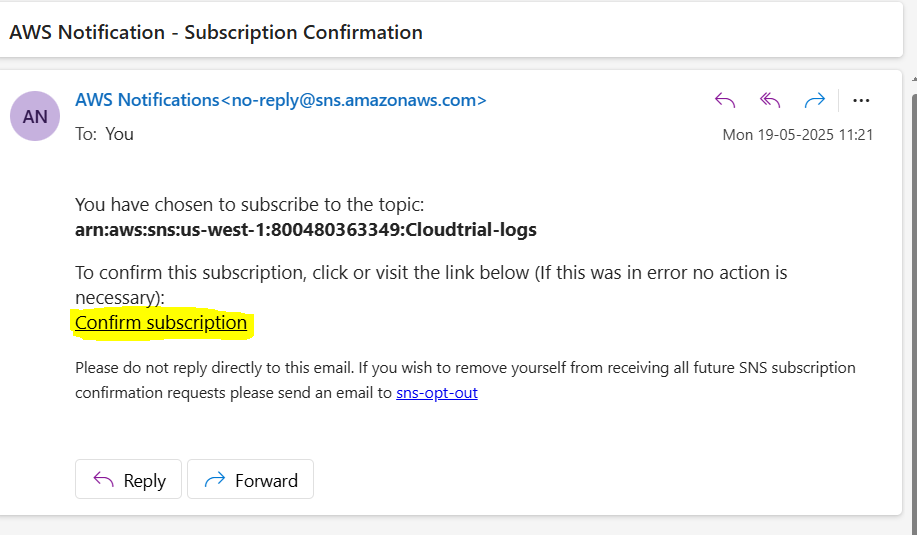
4. Enter your email address.

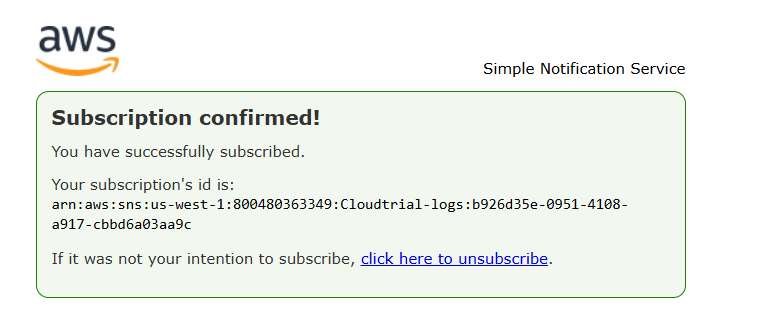


5. Click "Create subscription".



6. Confirm your email subscription by clicking the link sent to your email.





**Step 3: Configure CloudTrail to Send Notifications to SNS**

1. Navigate to the CloudTrail dashboard.

2. Select the trail.

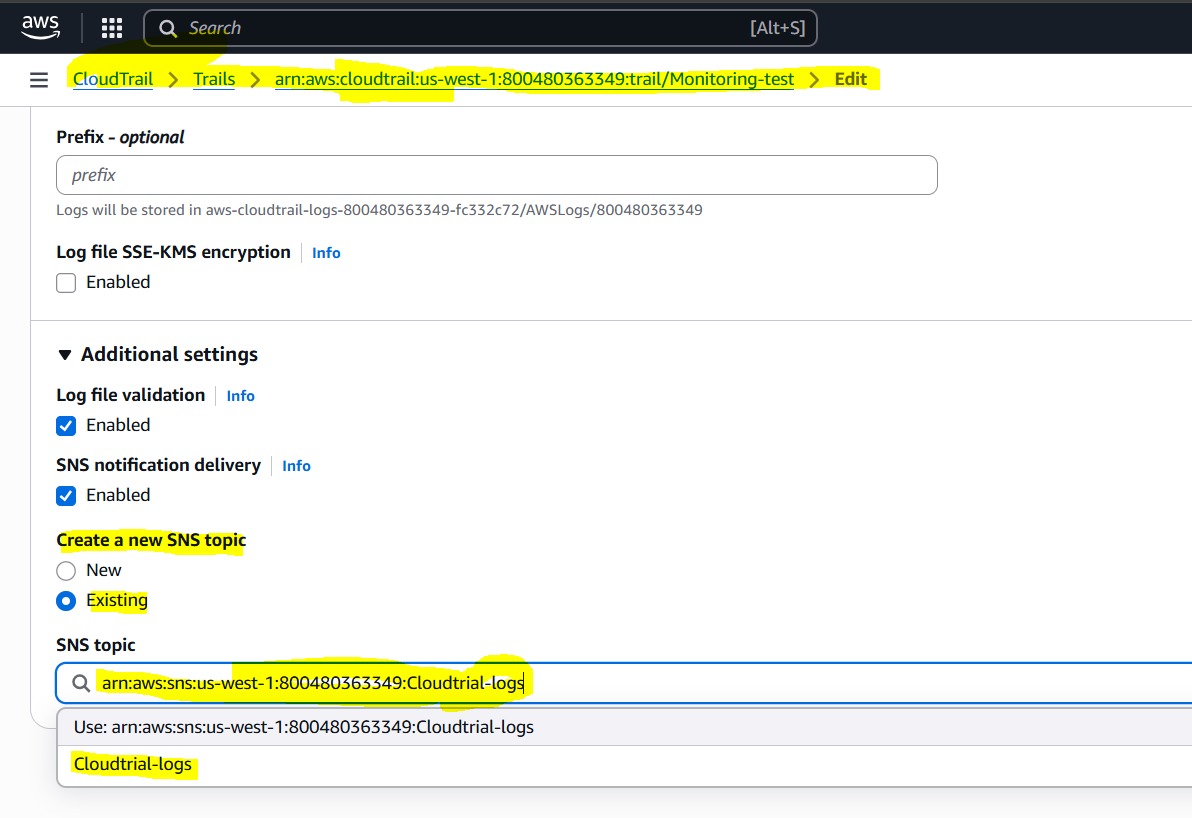
3. Click "Edit trail".

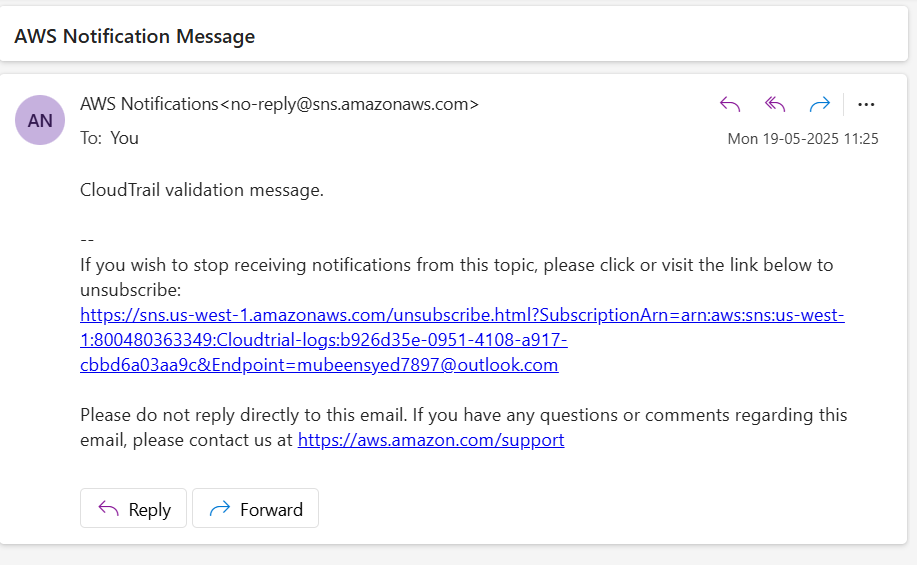
4. Scroll down to the "SNS notifications" section.

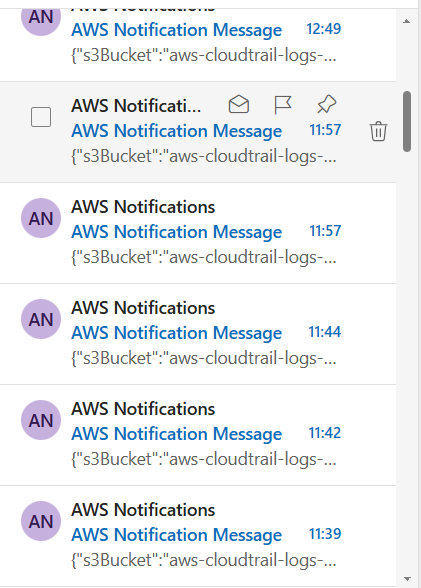
5. Select the SNS topic created in Step 1.

6. Choose the events you want to trigger notifications for.

7. Click "Save".







**3) Configure cloud watch monitoring and record the cpu utilization and other metrics of ec2.   
  
Step 1: Navigate to CloudWatch Dashboard**

1. Log in to the AWS Management Console.

2. Navigate to the CloudWatch dashboard.



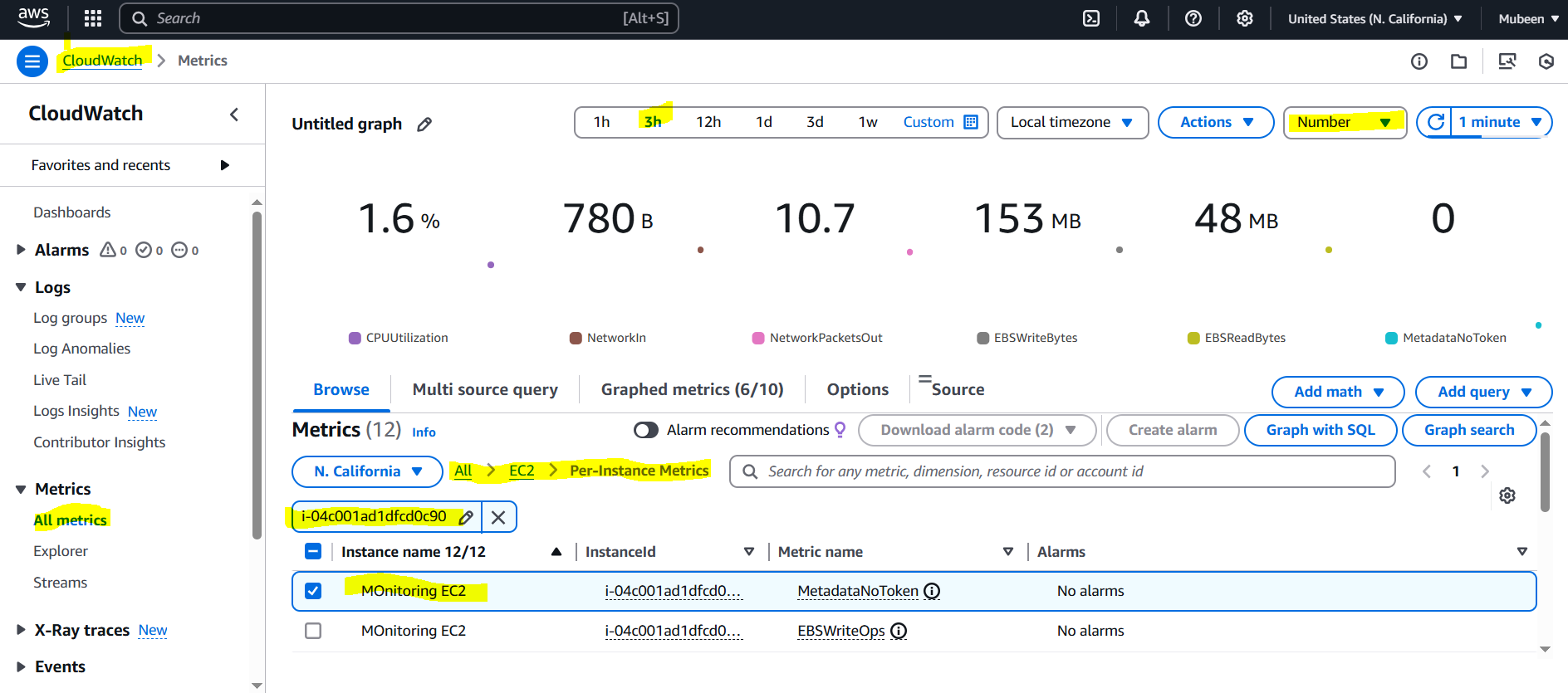
**Step 2: Select EC2 Metrics**

1. In the CloudWatch dashboard, click "Metrics".

2. In the search bar, type "EC2" and press Enter

3.. In the search bar, type "Instance ID" and press Enter

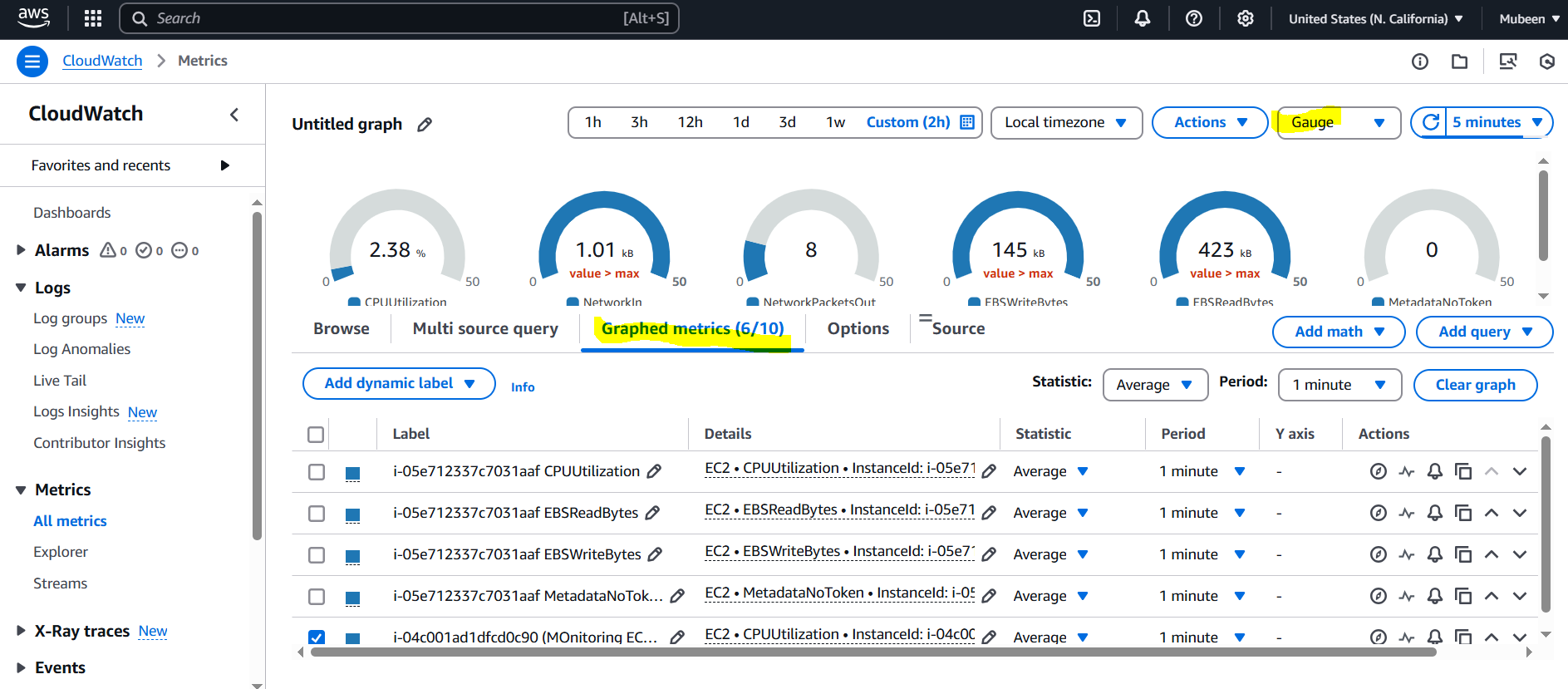
3. Select the metric you want to view.



**Step 3: View EC2 Metrics**

1. Choose the metric you want to view.

2. Use the time range and period settings to customize the graph.



**4) Create one alarm to send alert to email if the cpu utilization is more than 70 percent.   
  
Step 1: Navigate to CloudWatch**

1. Log in to the AWS Management Console.

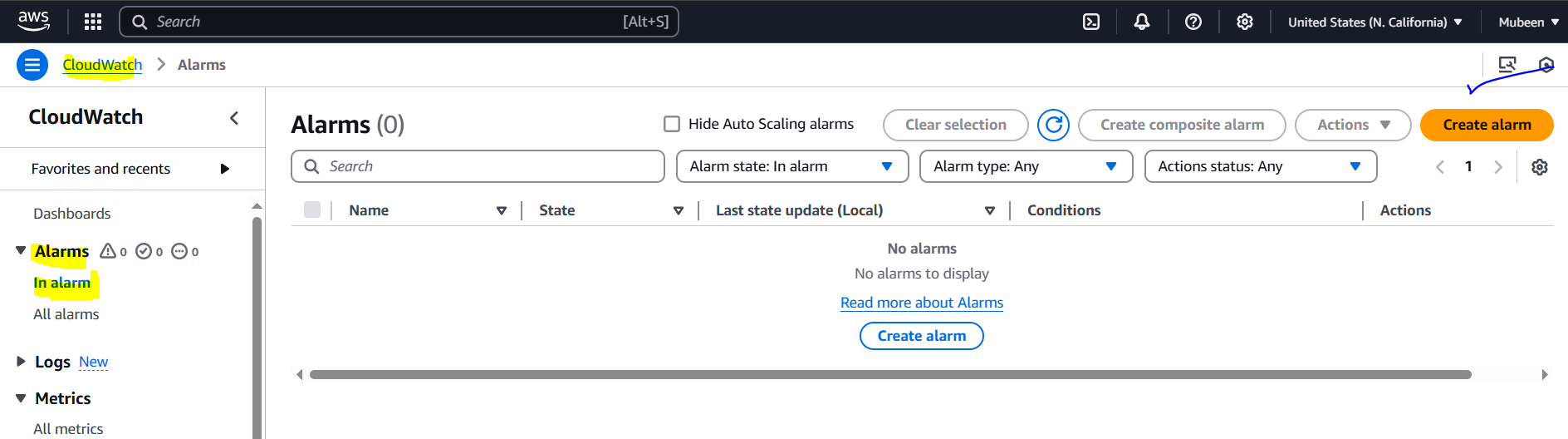
2. Navigate to the CloudWatch dashboard.



**Step 2: Create Alarm**

1. Click on "Alarms" in the left sidebar.

2. Click on "Create Alarm".



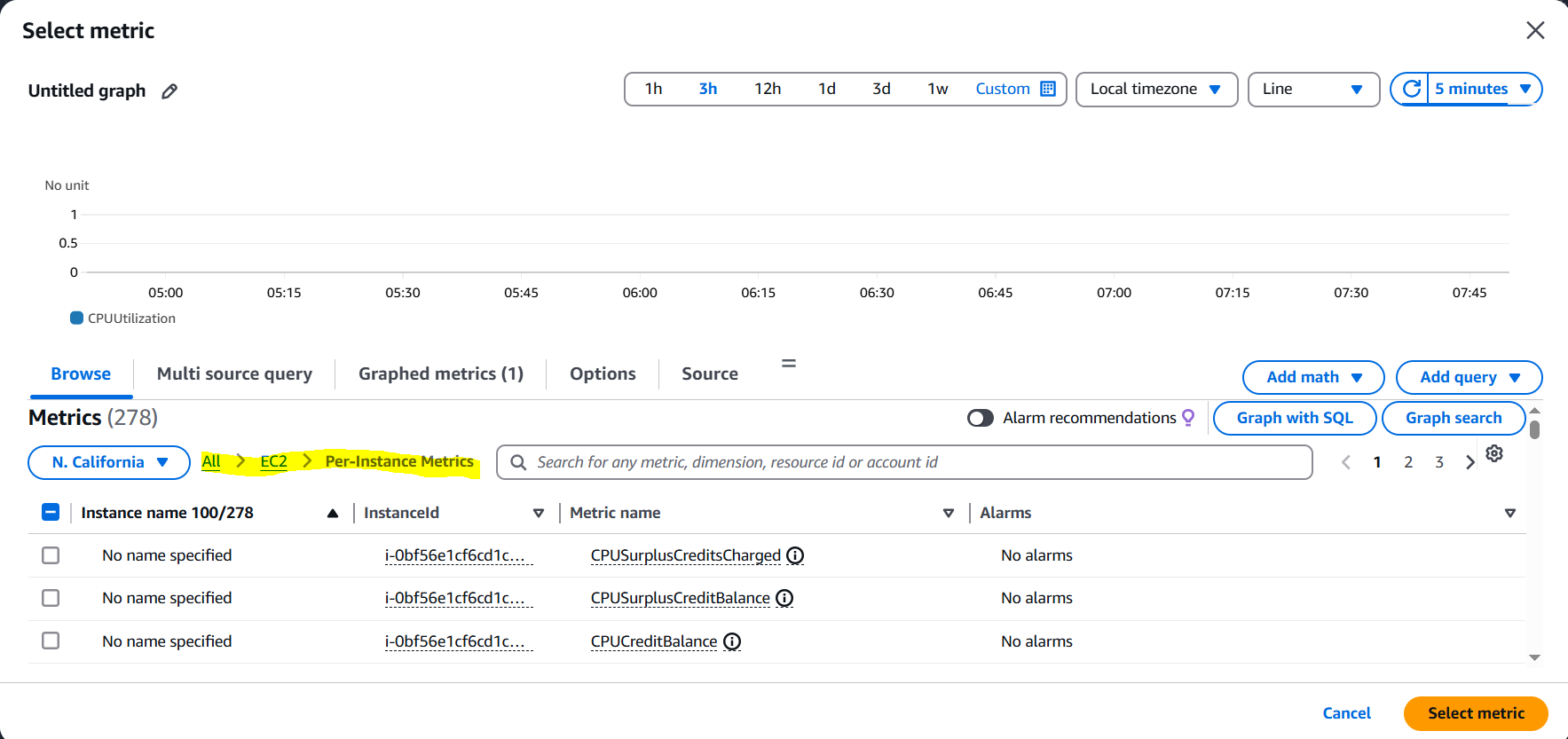
**Step 3: Select Metric**

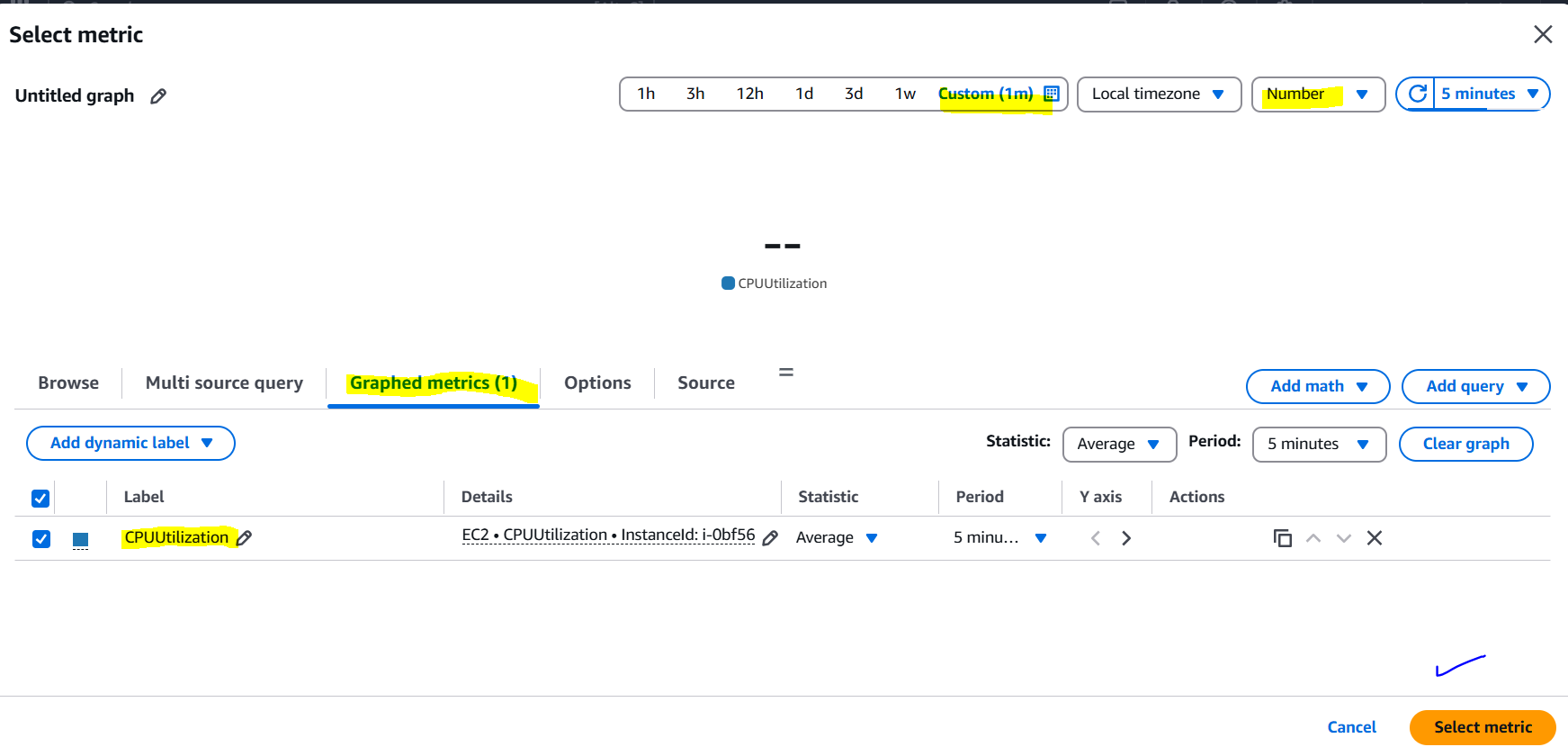
1. Click on "Select metric".

2. Choose "EC2" and then select "Per-Instance Metrics".

3. Select the instance you want to monitor.

4. Choose the "CPUUtilization" metric.

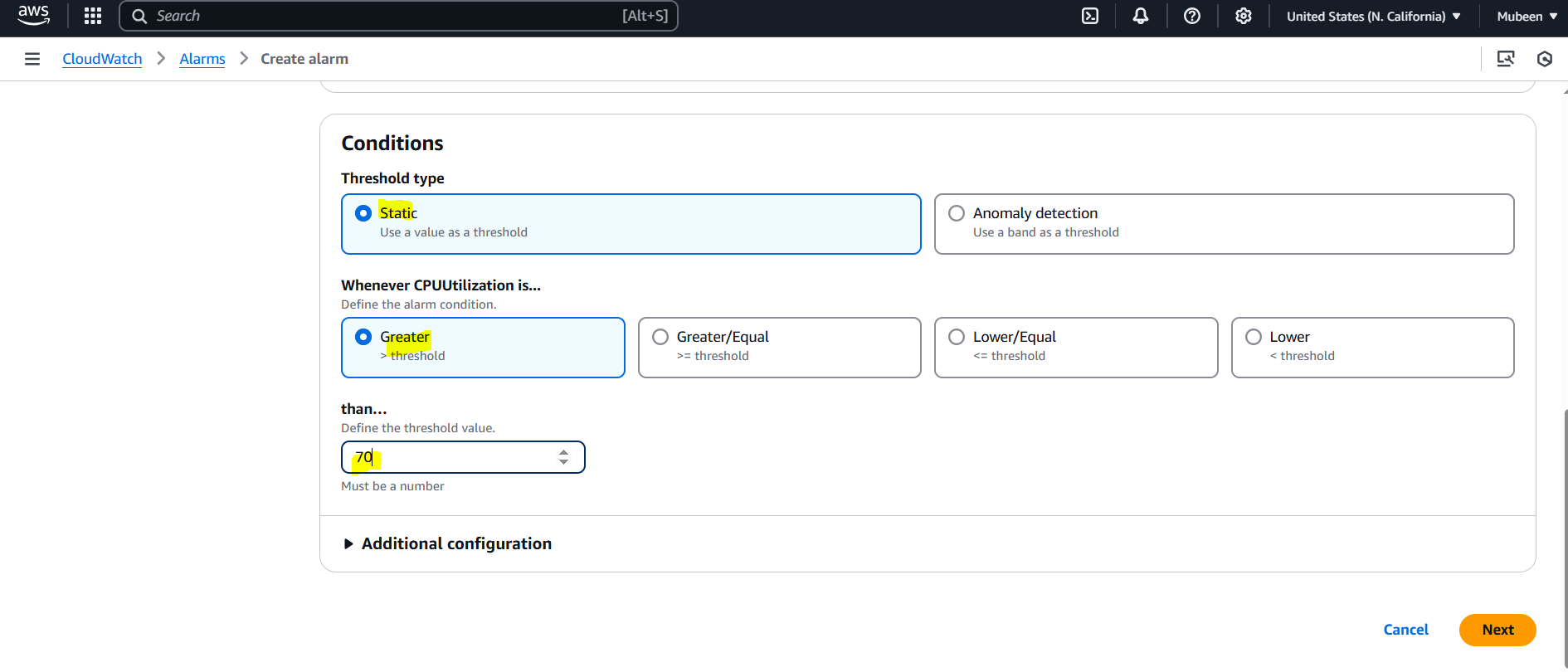




**Step 4: Configure Alarm**

1. Set the threshold to "Greater than" and enter "70" in the value field.

2. Set the period to the desired time frame.

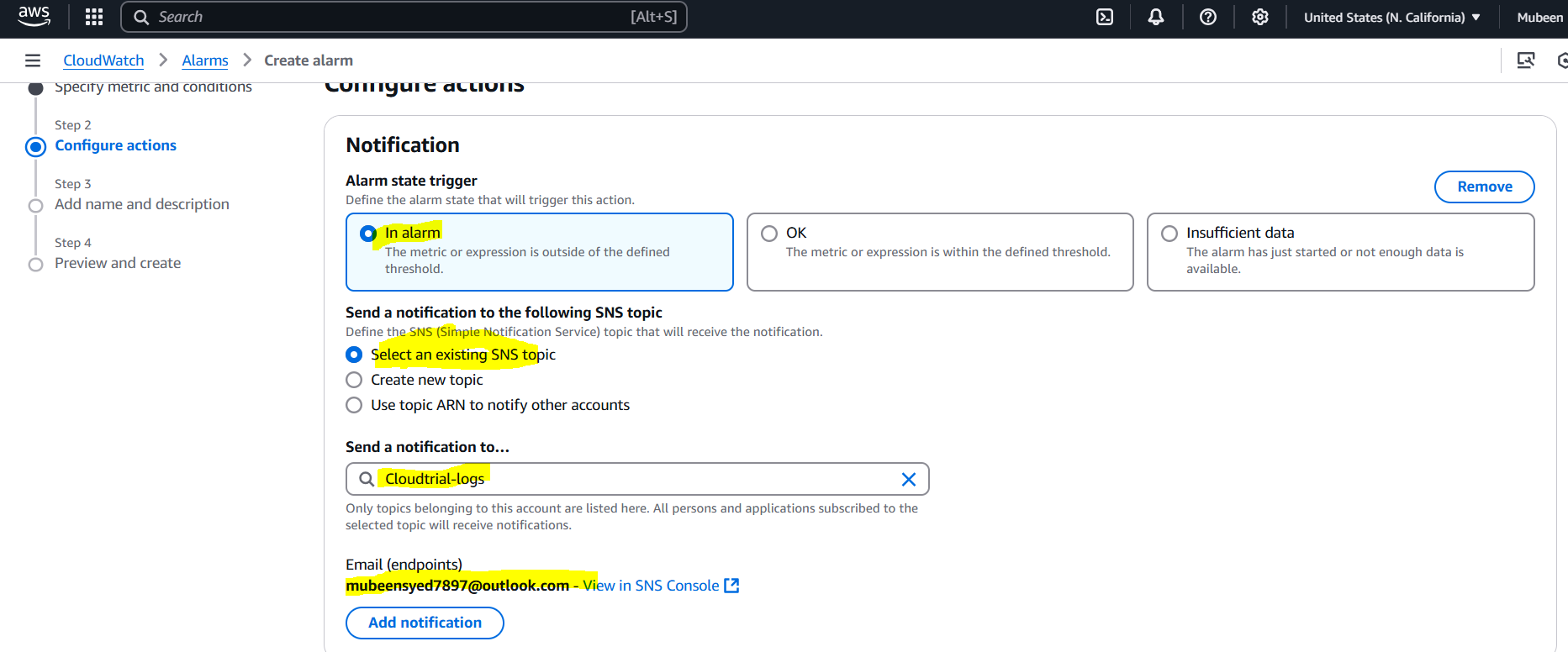


**Step 5: Configure Actions**

1. Click on "Actions".

2. Select "SNS topic" and then choose "Create a new topic" or select an existing one.

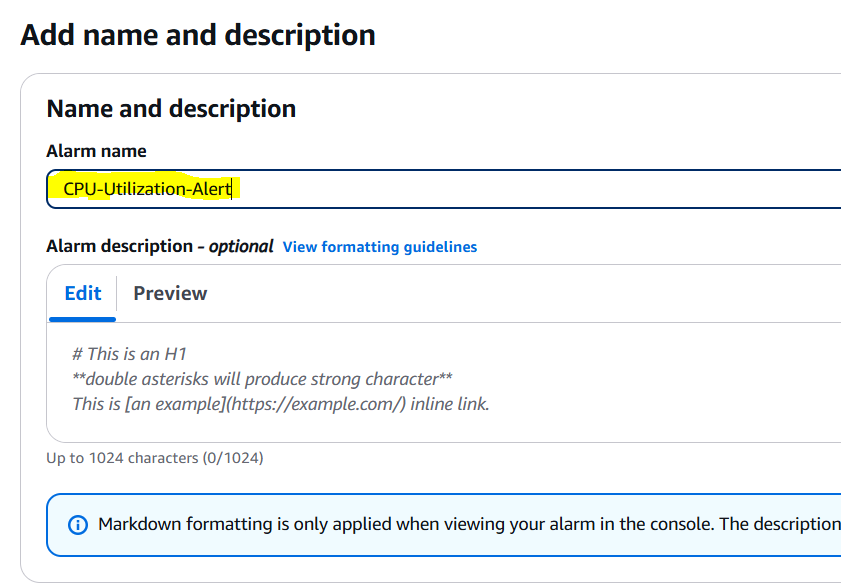
3. Enter the email address that will receive the notifications.

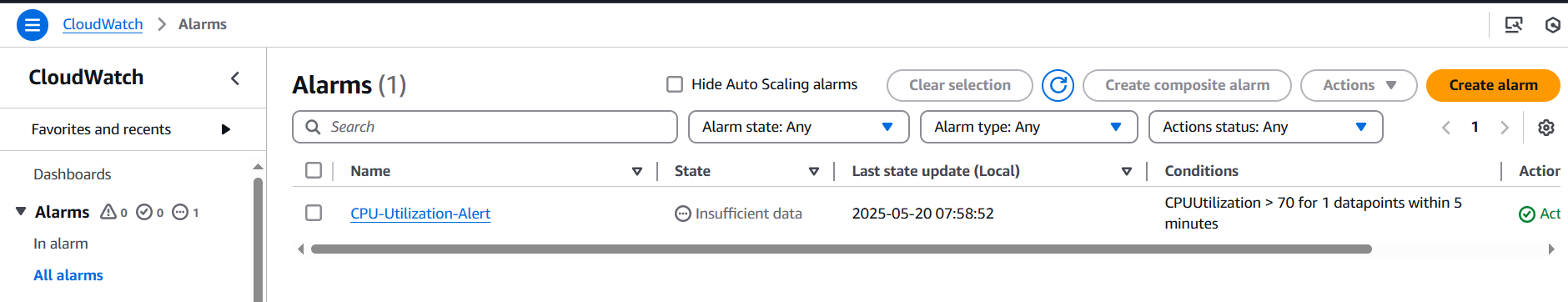


**Step 6: Create Alarm**

1. Give the alarm a name and description.

2. Click "Create Alarm".

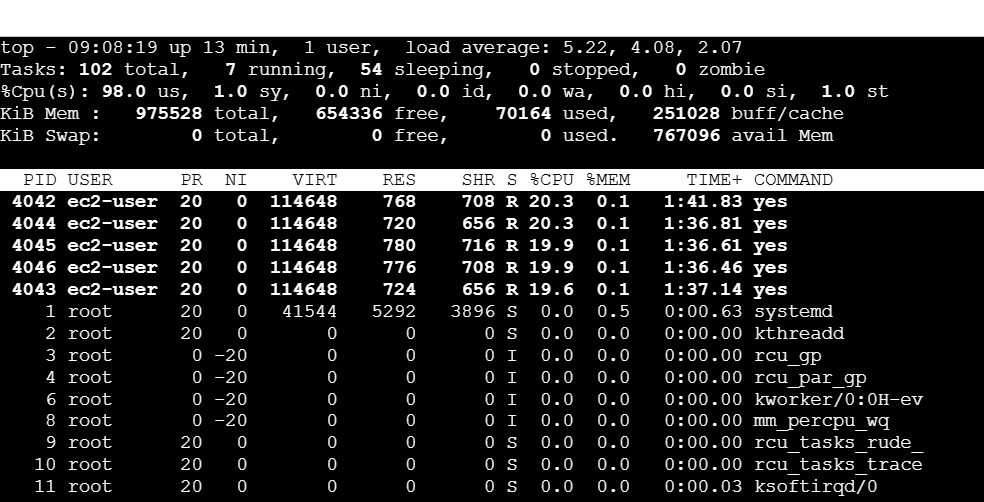




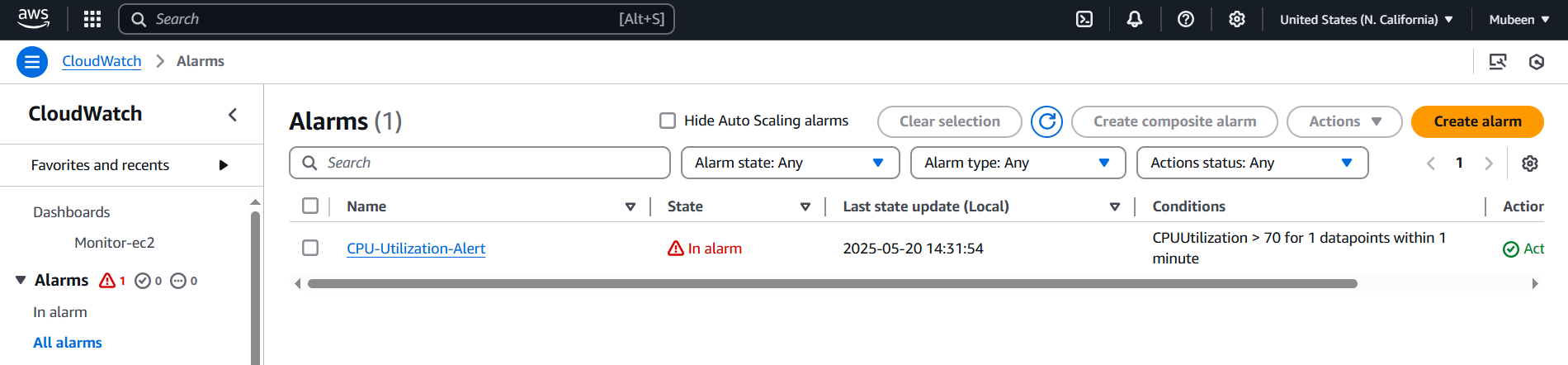
**Step 7: Testing Alarm**

1. Increasing load on cpu for testing and checking for mail.

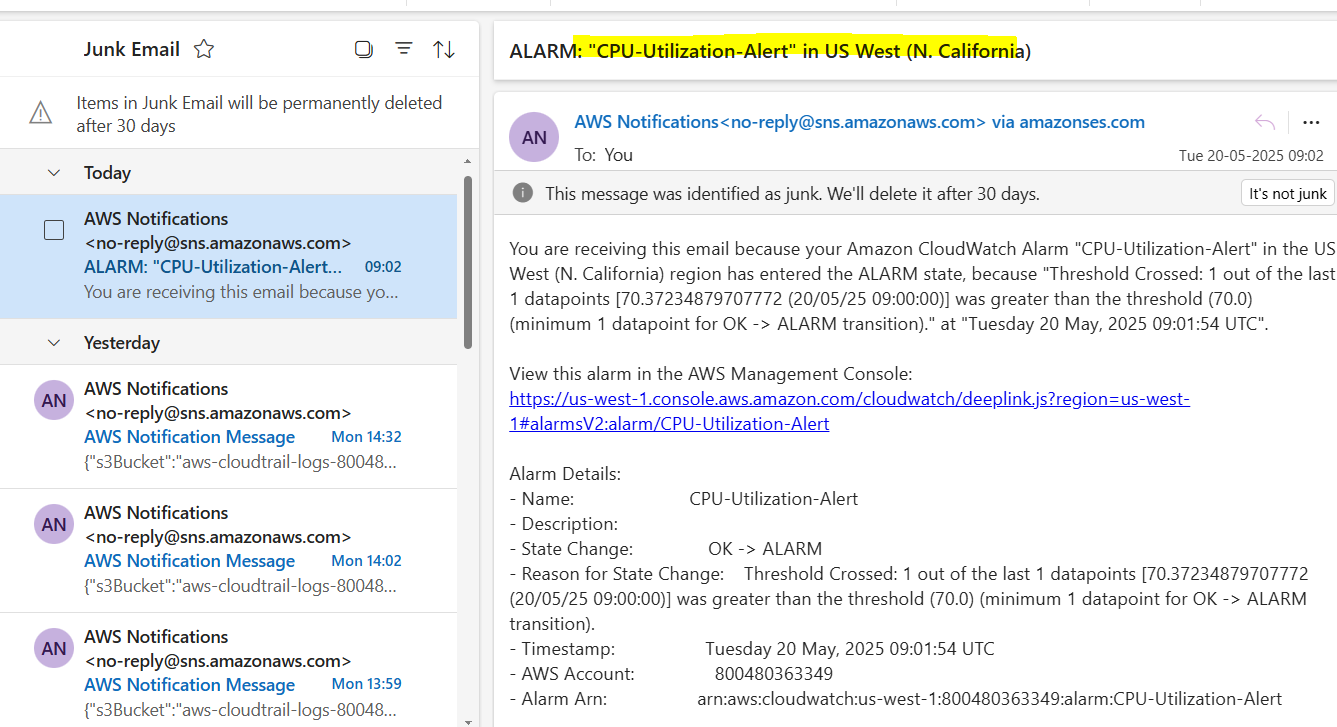




Alarm alert on console.



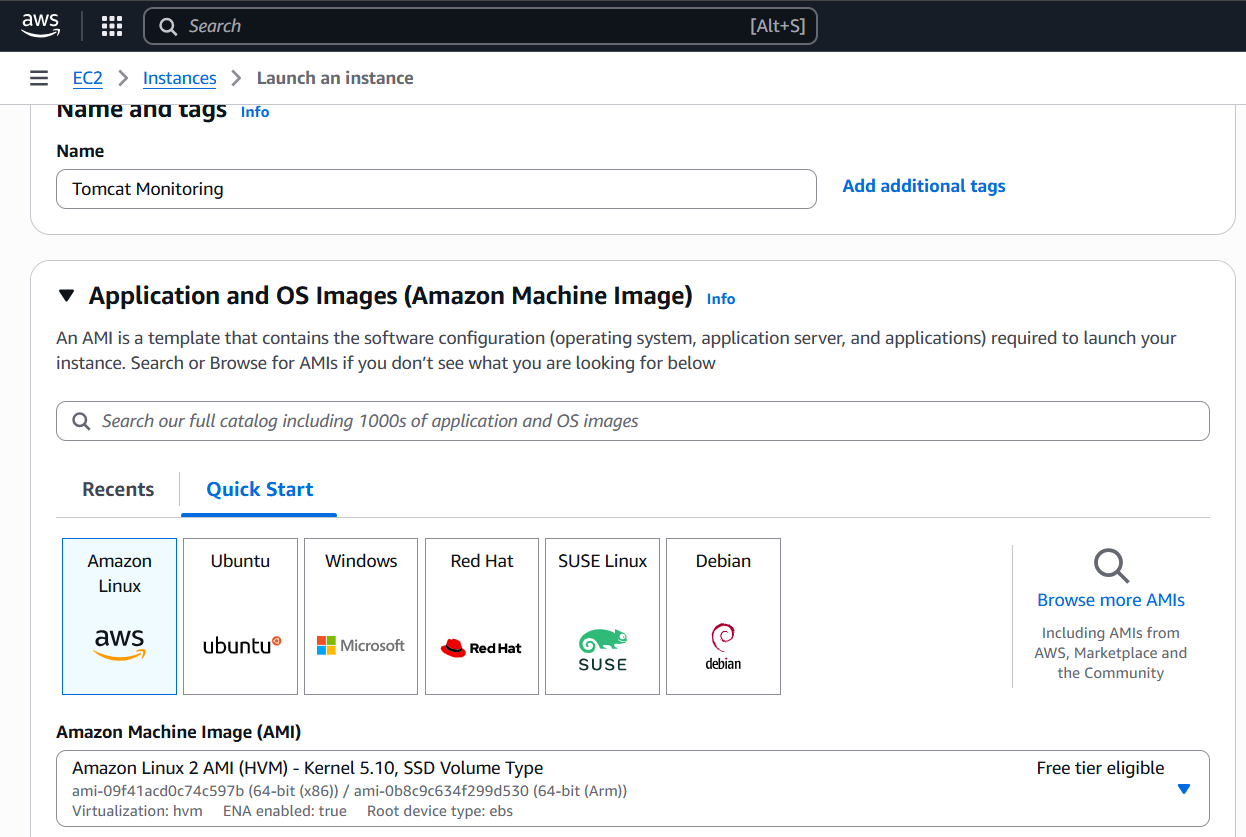
Alert on mail.



**5) Create Dashboard and monitor tomcat service whether it is running or not and send the alert.   
  
Step 1: Launch EC2 Instance**

1. Launch an EC2 instance with Amazon Linux or Ubuntu.

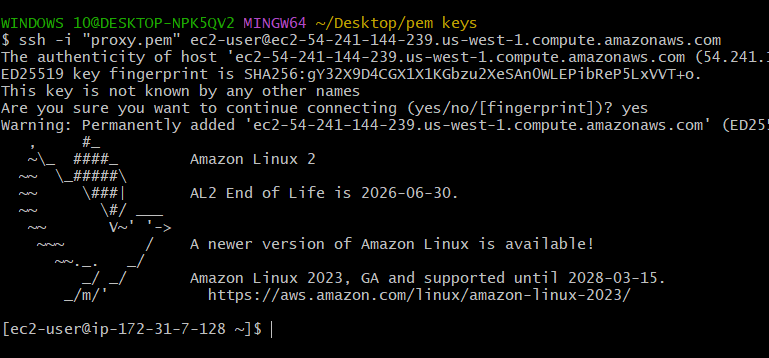
2. Configure security group to allow HTTP (port 8080) access.





**Step 2: Install Tomcat**

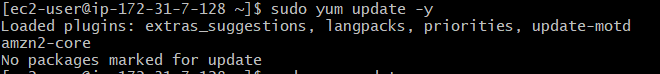
1. Connect to your EC2 instance using SSH.



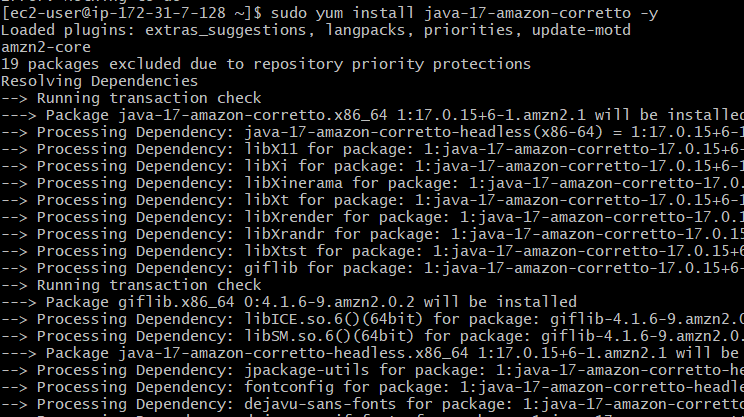
2. Install Tomcat using the following commands:

2. Install Java and Tomcat using the following commands:

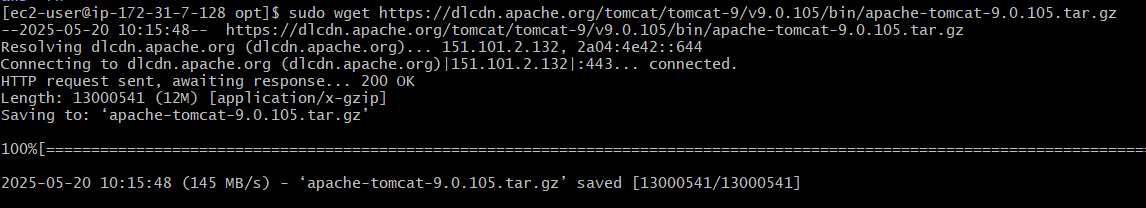
sudo yum update -y



sudo yum install java-17-amazon-corretto -y



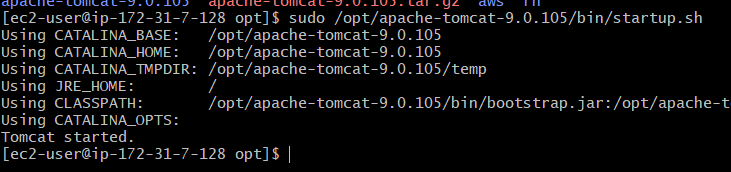
wget <https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.105/bin/apache-tomcat-9.0.105.tar.gz>



sudo tar -xvf apache-tomcat-9.0.105.tar.gz



sudo /opt/apache-tomcat-9.0.105/bin/startup.sh





**Step 3: Create Script**

1. Create a script that checks the Tomcat service status and pushes the custom metric:

bash

#!/bin/bash

# Get Instance ID from metadata

INSTANCE\_ID=$(curl -s http://169.254.169.254/latest/meta-data/i-0340a9ace3eafcab4)

# Check if Tomcat is running

if pgrep -f tomcat > /dev/null; then

STATUS=1

else

STATUS=0

fi

# Send the metric to CloudWatch

aws cloudwatch put-metric-data \

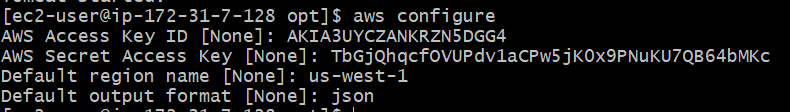
--namespace "Tomcat" \

--metric-name "TomcatServiceStatus" \

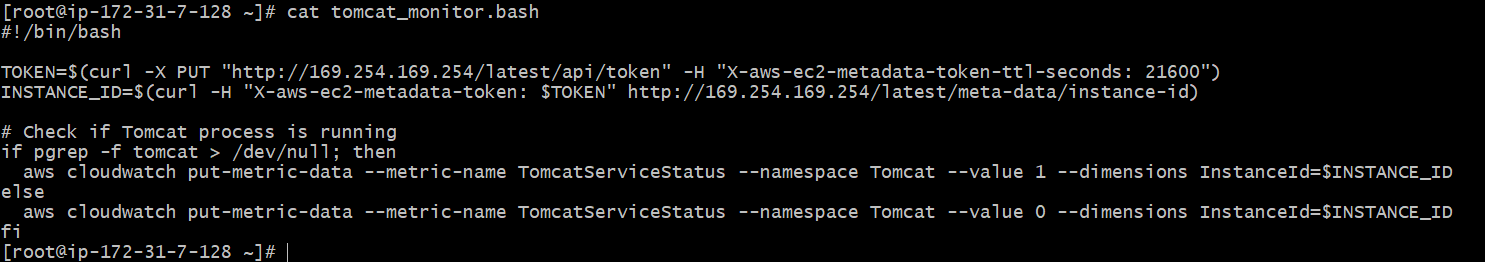
--value "$STATUS" \

--dimensions InstanceId="$INSTANCE\_ID"

2. After configuring the **AWS CLI**, you'll need to:



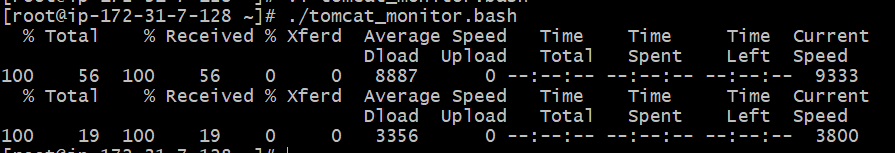
* Save the script: Save the script to a file (tomcat\_monitor.bash) on your EC2 instance.



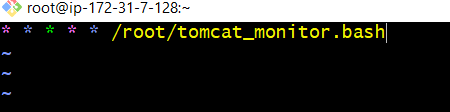
* Make the script executable: Run the command chmod +x tomcat\_monitor.bash to make the script executable.



* Run the script: Run the script manually by executing ./tomcat\_monitor.bash to test it.



* Schedule the script: Schedule the script to run at regular intervals using cron.

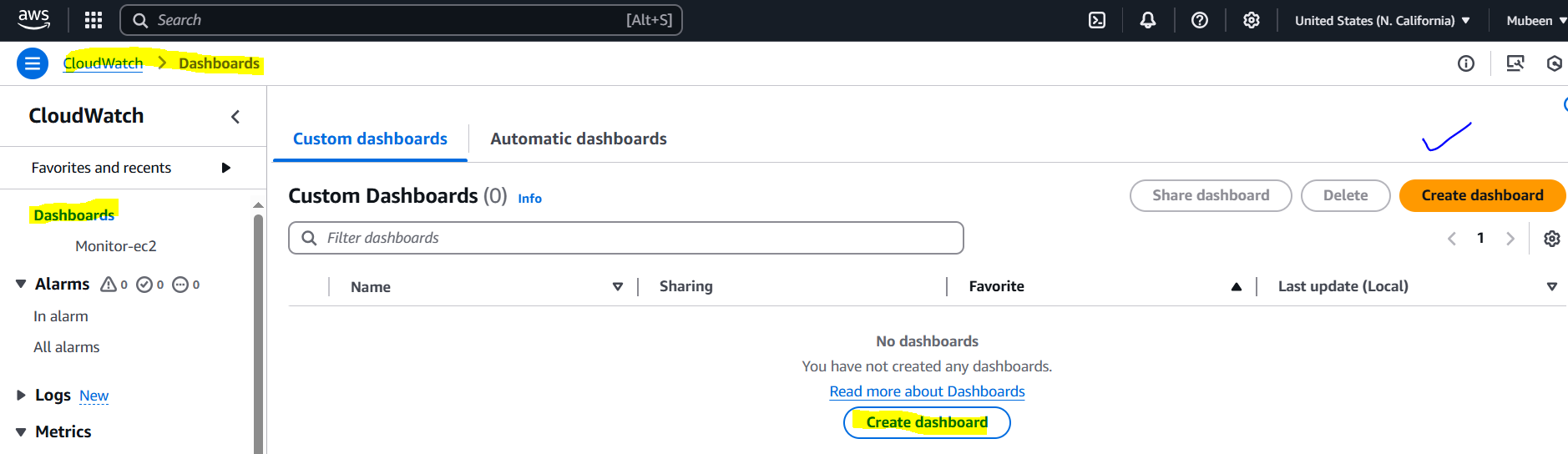




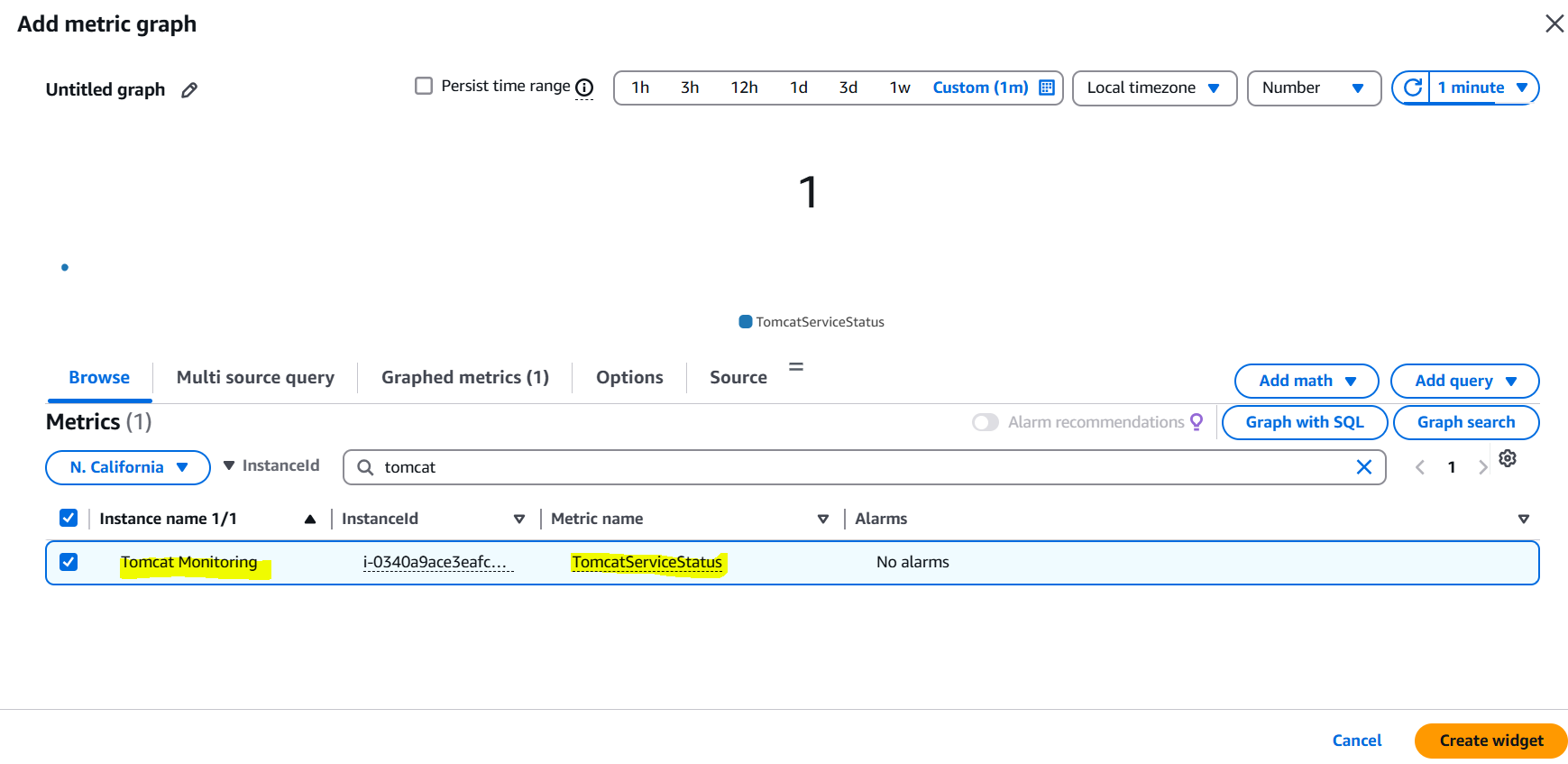
**Step 4: Create CloudWatch Dashboard**

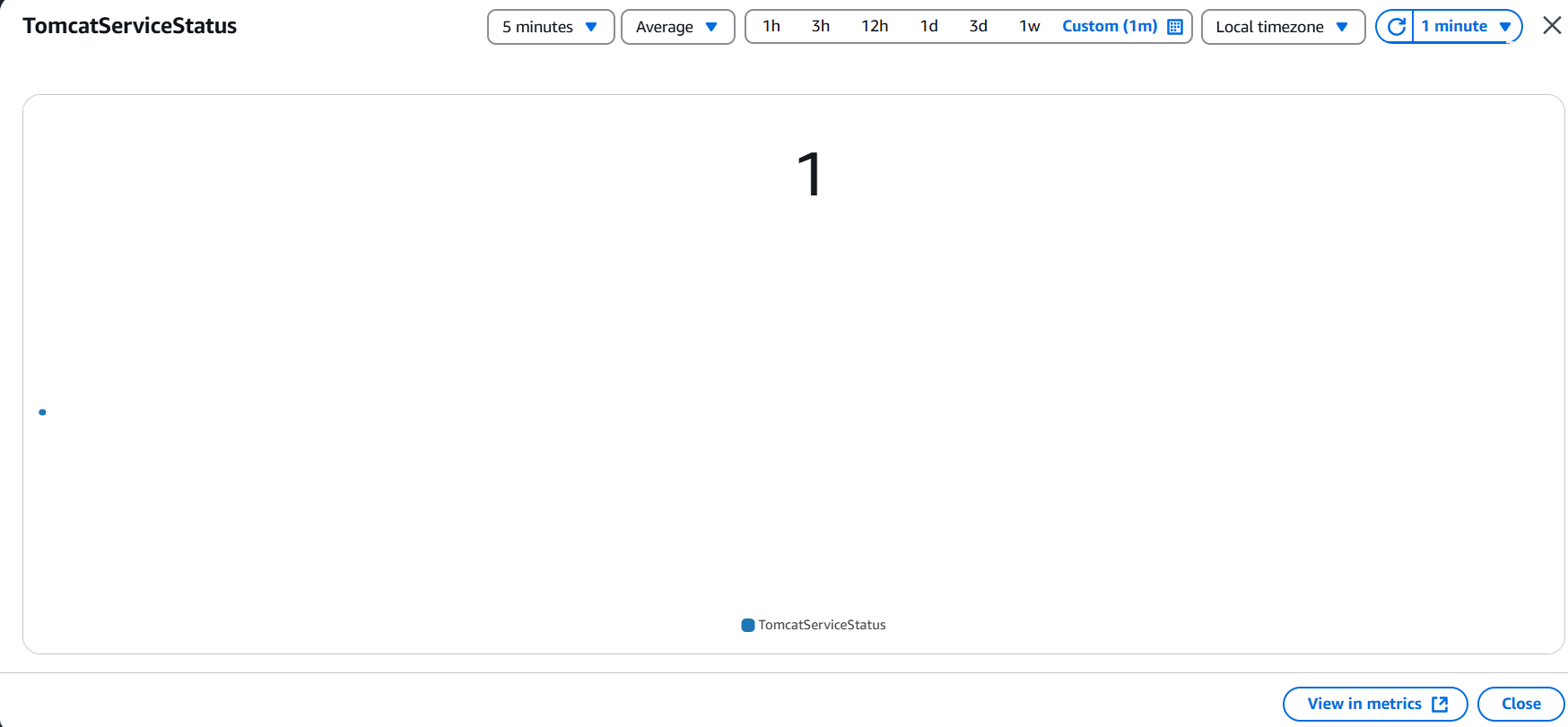
1. Go to CloudWatch console.

2. Click on "Dashboards" and then "Create dashboard".



3. Choose a dashboard type and add a widget with the custom metric "TomcatServiceStatus".

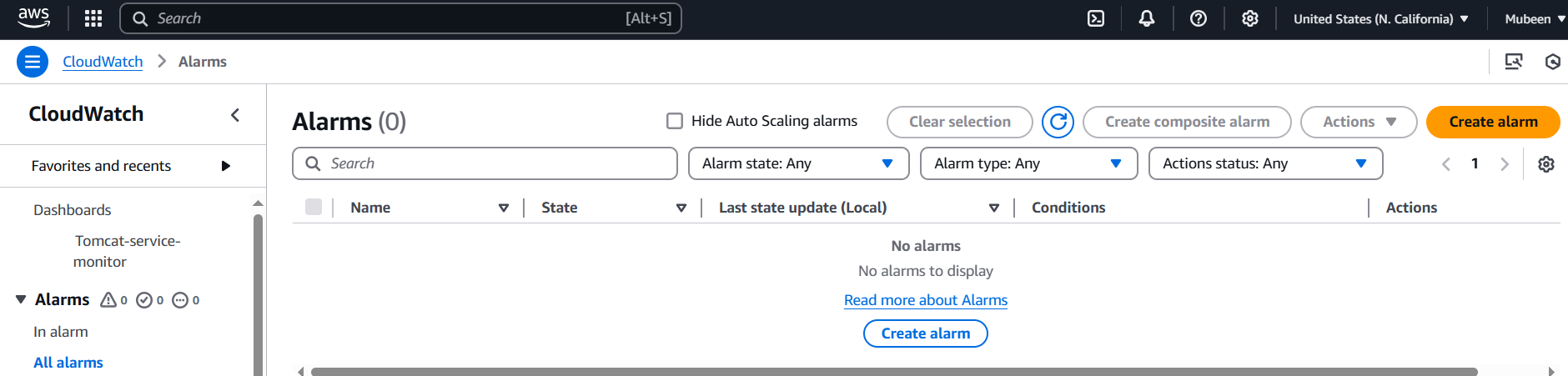




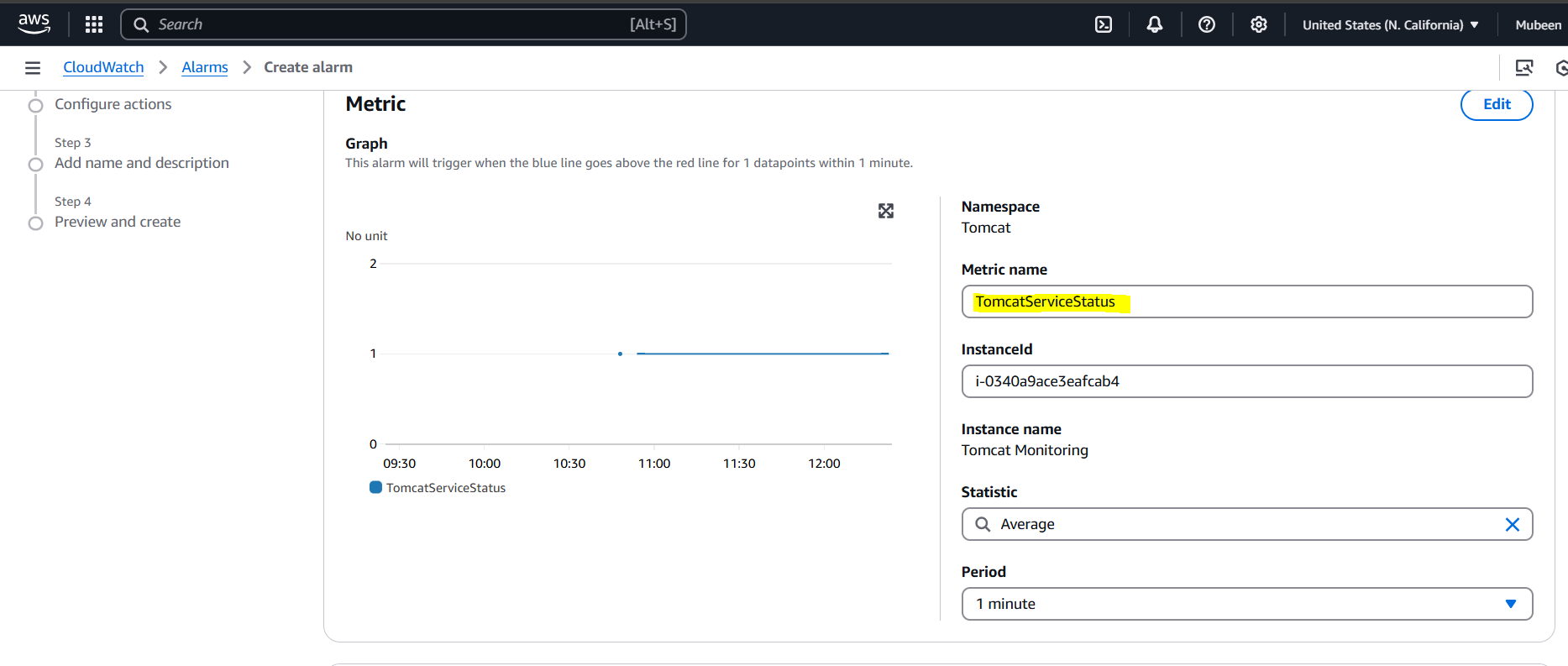
**Step 5: Create CloudWatch Alarms**

1. Go to CloudWatch console and navigate to Alarms.

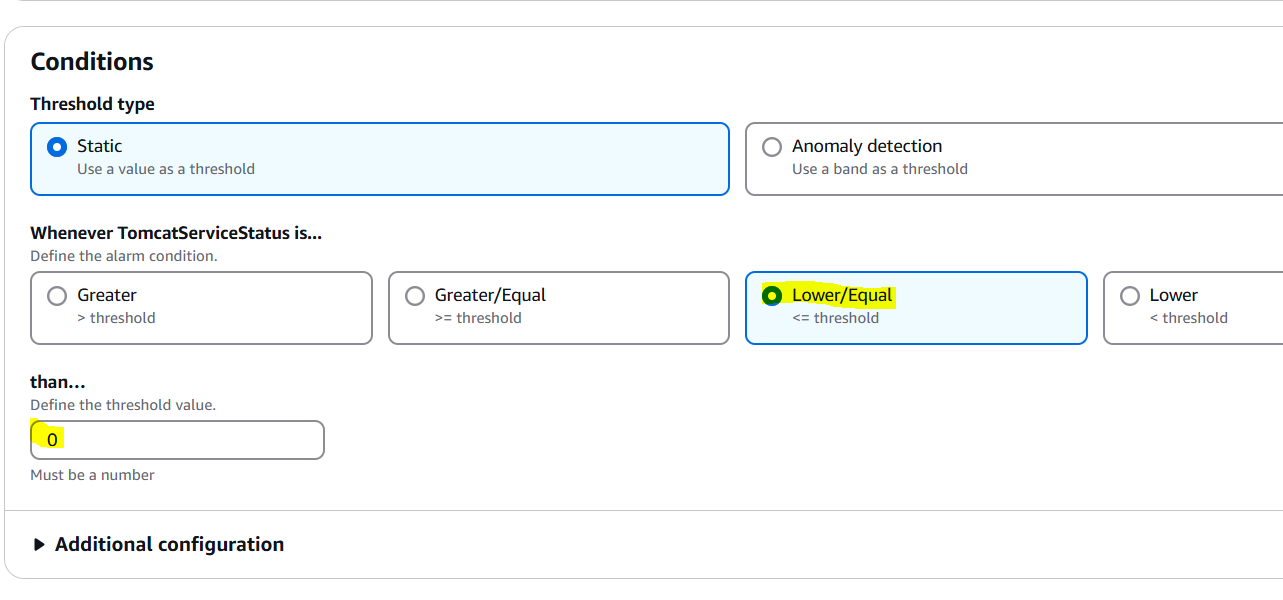
2. Click "Create alarm".



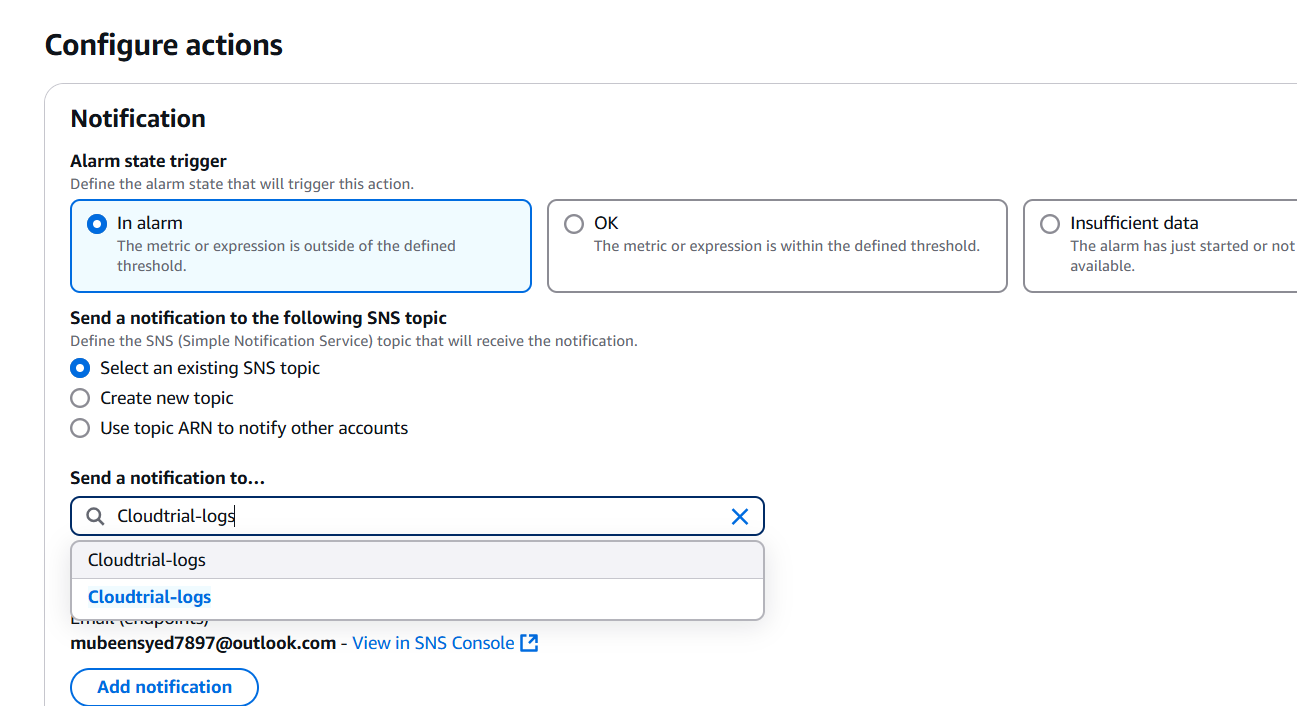
3. Select the TomcatServiceStatus metric.



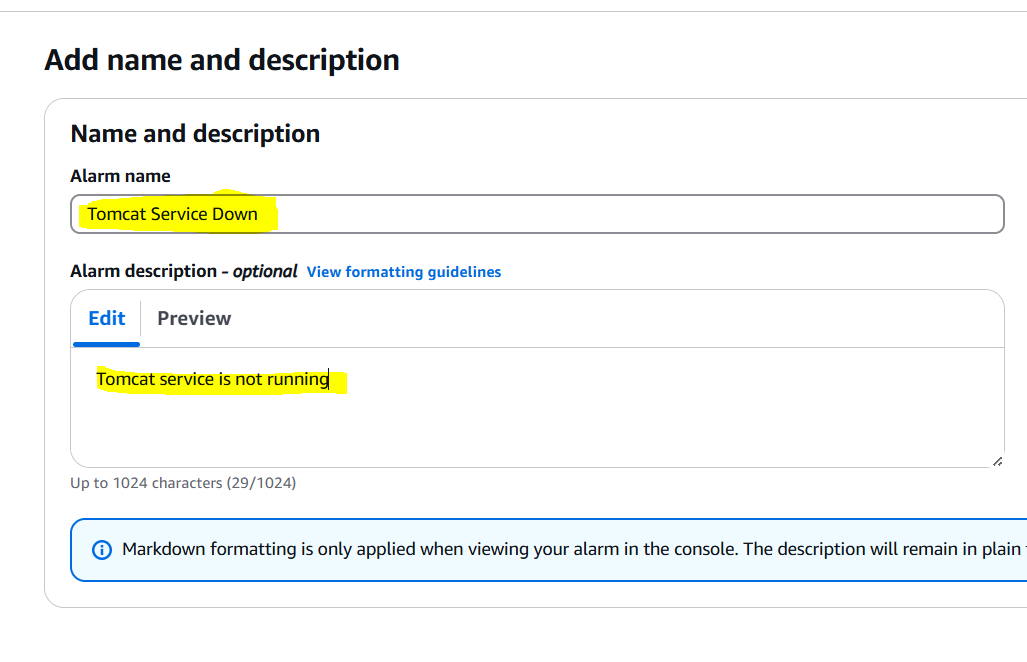
4. Set the threshold to <= 0.



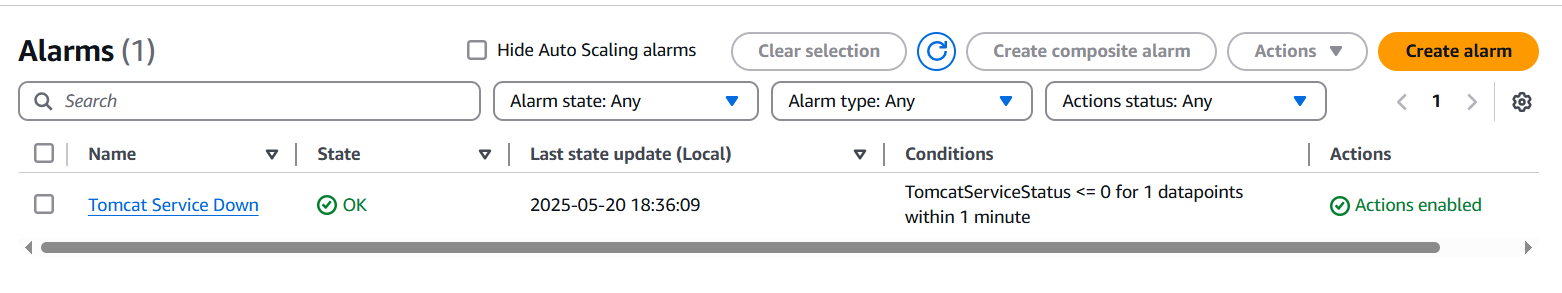
5. Configure the actions (e.g., send SNS notification).

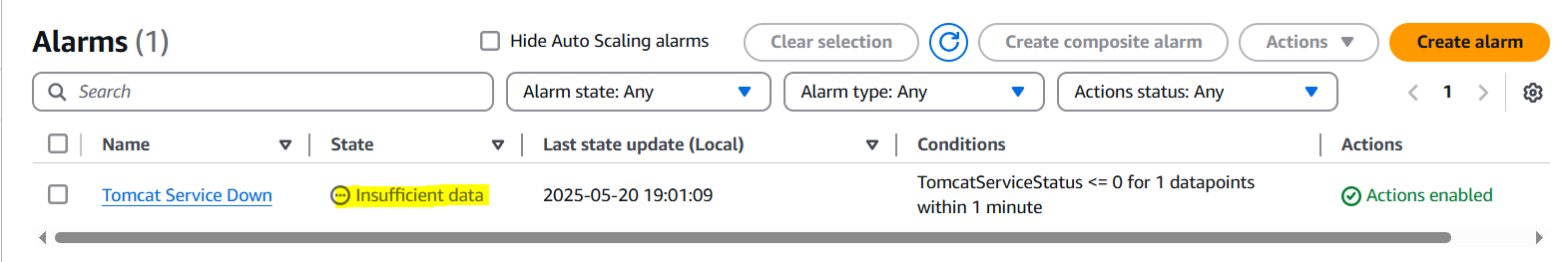


6. Set the alarm name to "Tomcat Service Down".

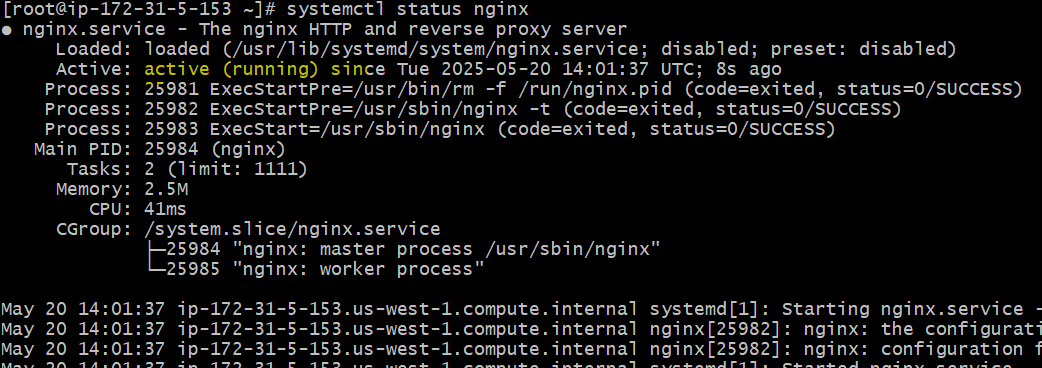


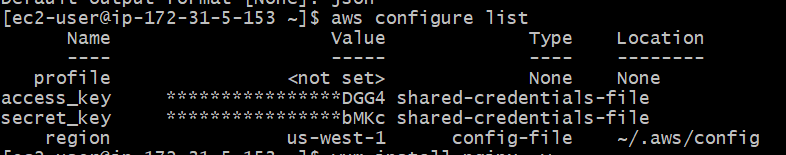
7. Create the alarm.



If instance is terminated.  


**6) Create Dashboard and monitor nginx service to send the alert if nginx is not running.**

We have a existing nginx service and aws cli is already configured.  
  




**Step 1: Create a Script to Monitor Nginx**

Create a script (e.g., nginx\_monitor.sh) to check the Nginx service status and push metric data to CloudWatch:

#!/bin/bash

# Check Nginx service status

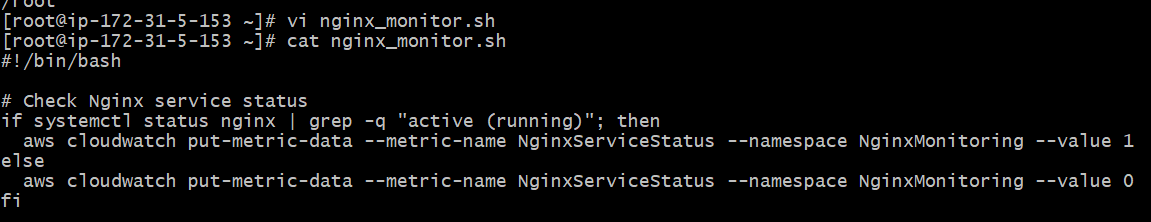
if systemctl status nginx | grep -q "active (running)"; then

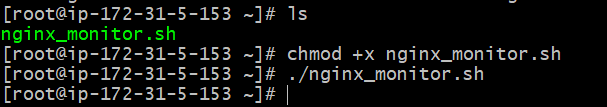
aws cloudwatch put-metric-data --metric-name NginxServiceStatus --namespace NginxMonitoring --value 1

else

aws cloudwatch put-metric-data --metric-name NginxServiceStatus --namespace NginxMonitoring --value 0

fi





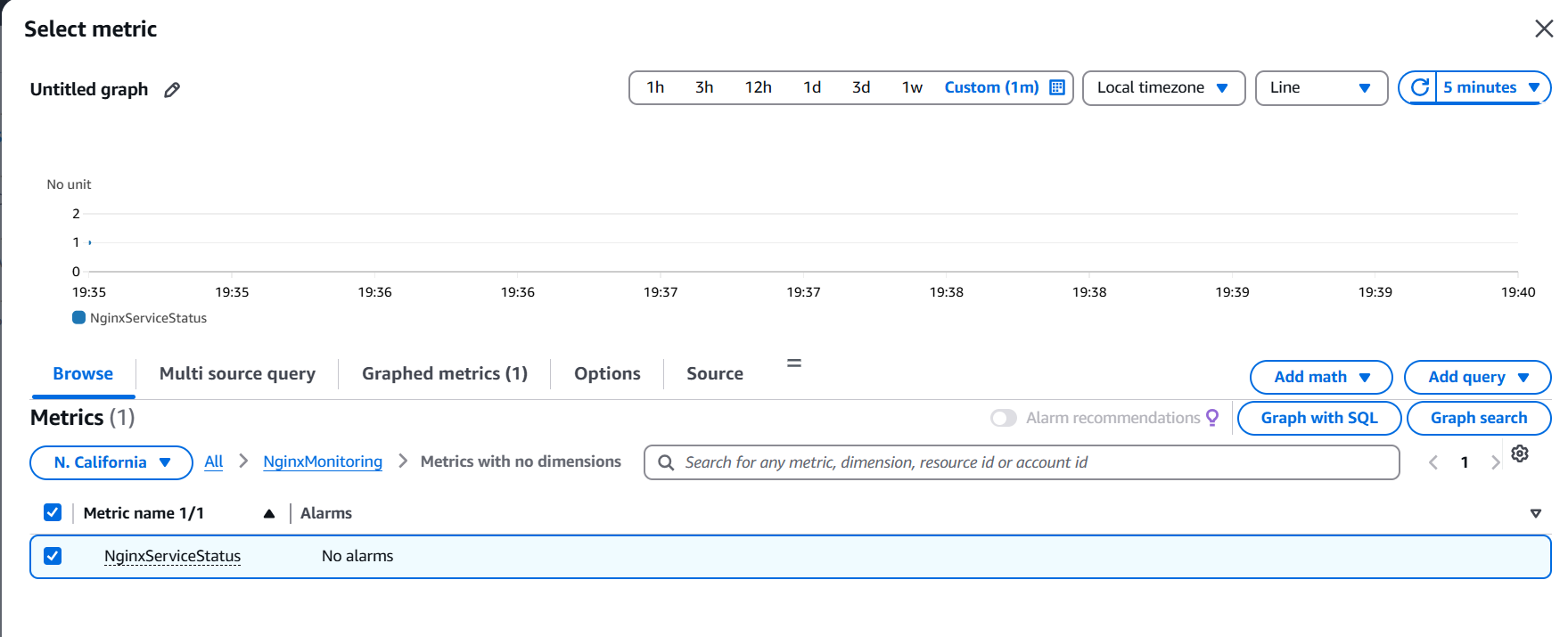
**Step 2: Create a CloudWatch Alarm**

1. Go to the CloudWatch console.

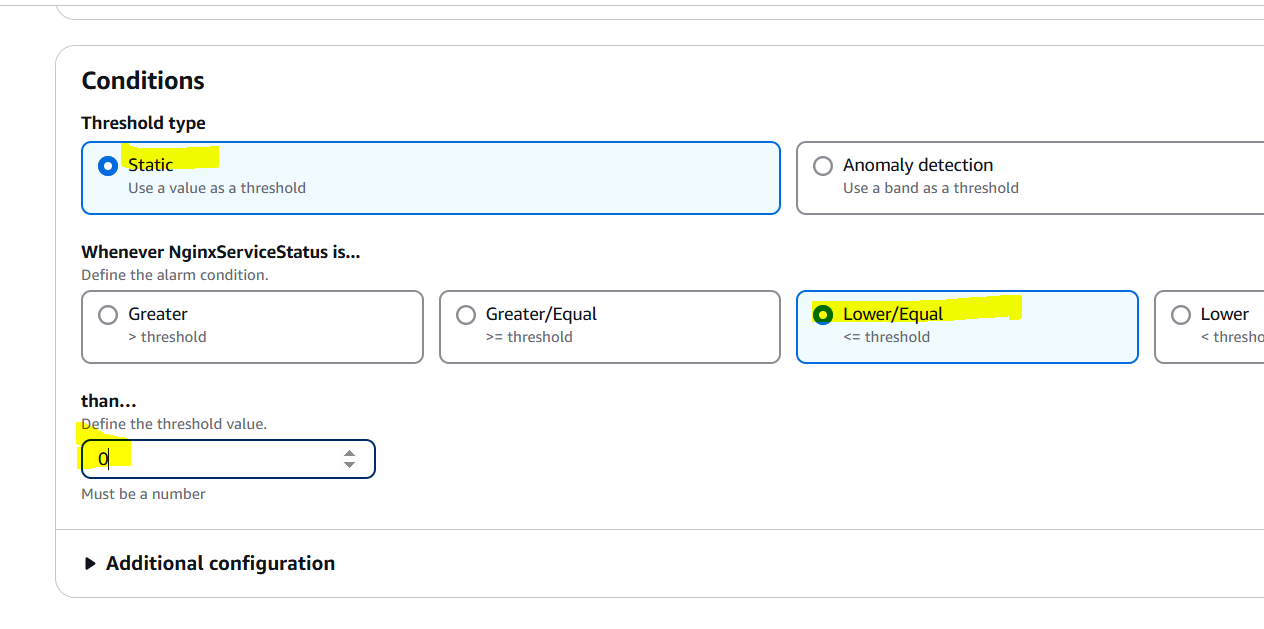
2. Navigate to Alarms.

3. Click Create alarm.

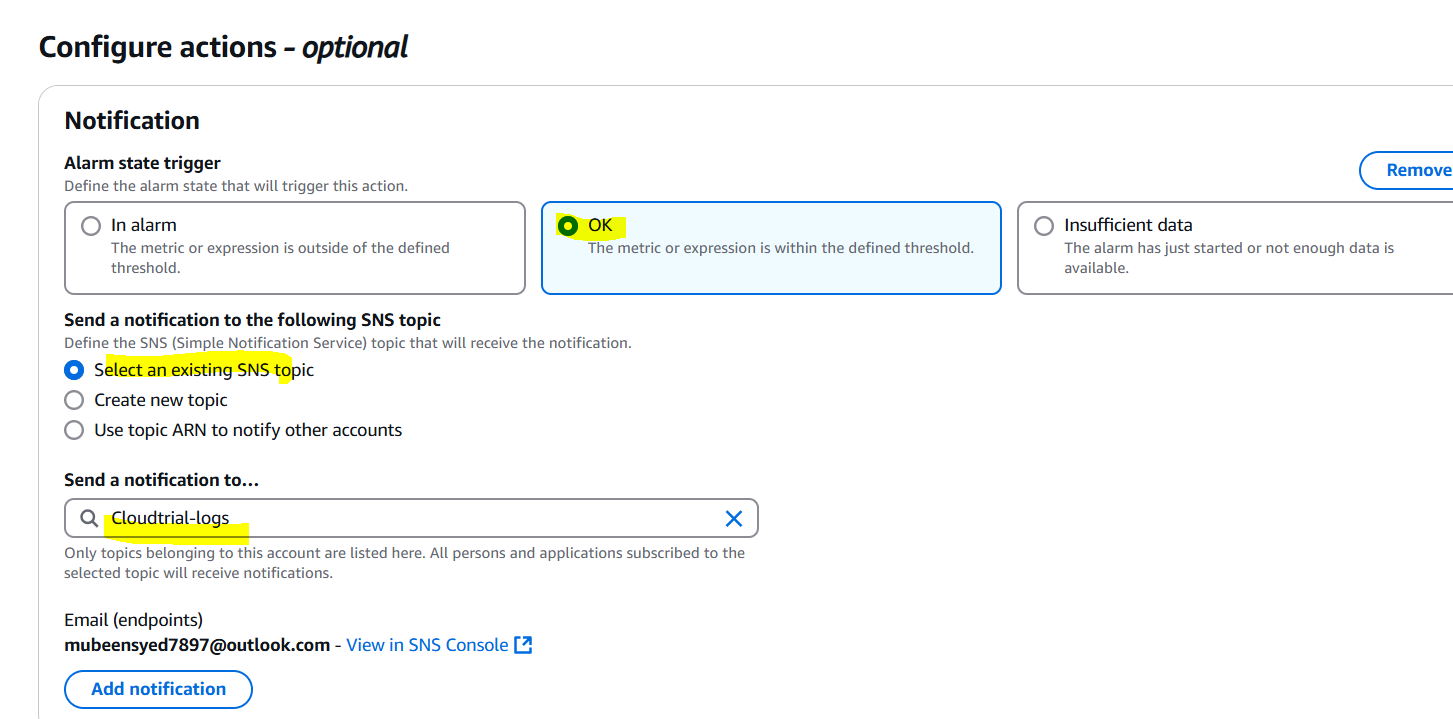
4. Select the NginxServiceStatus metric.

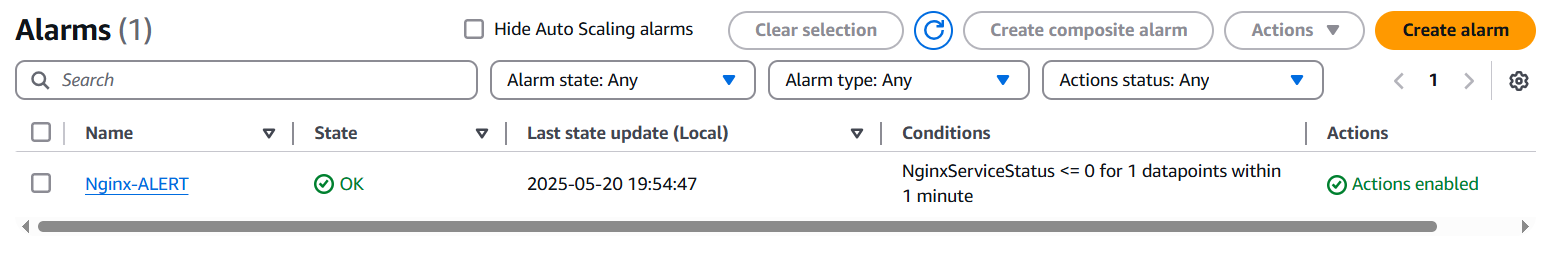


5. Set the threshold to <= 0.



6. Set the actions (e.g., send an SNS notification).





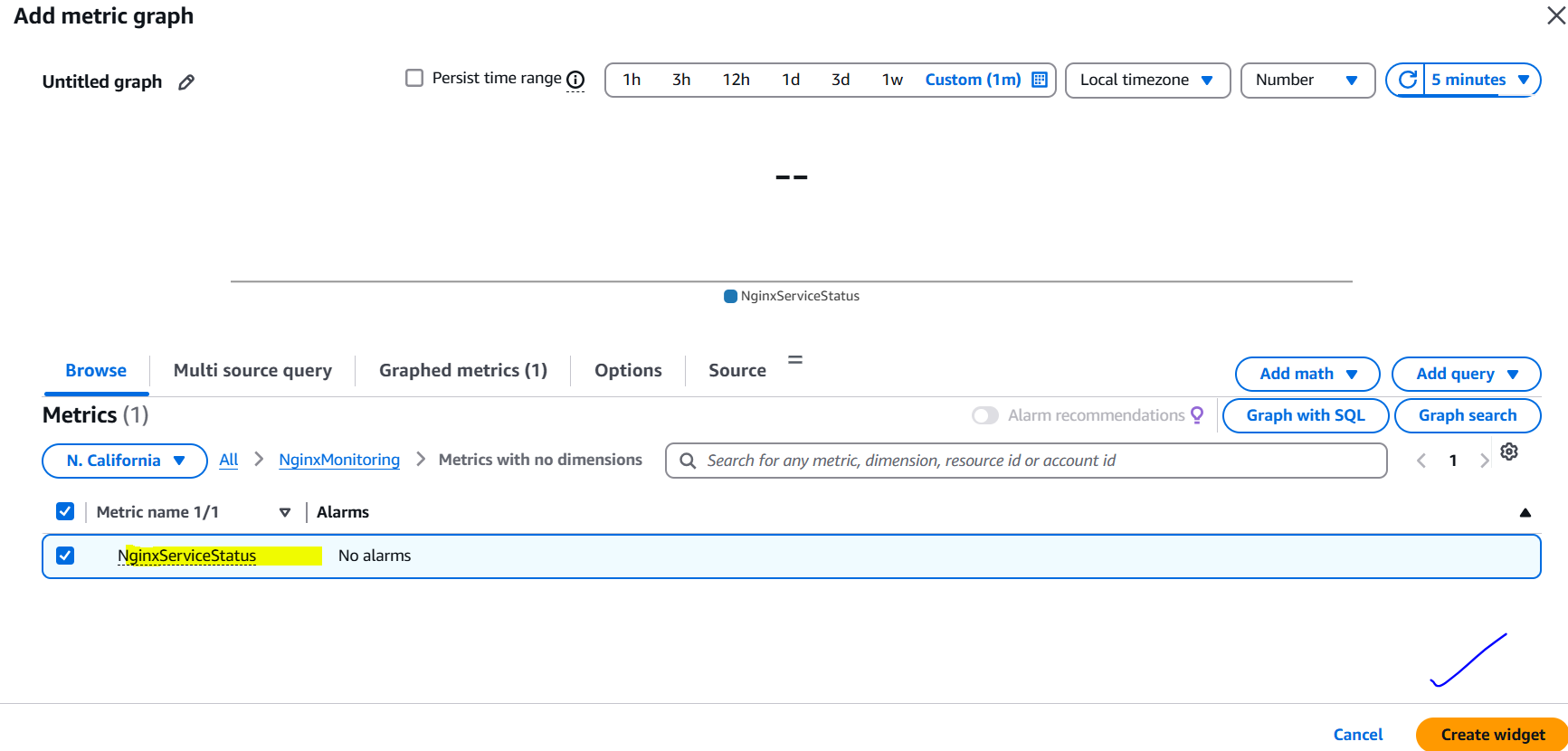
**Step 3: Create a CloudWatch Dashboard**

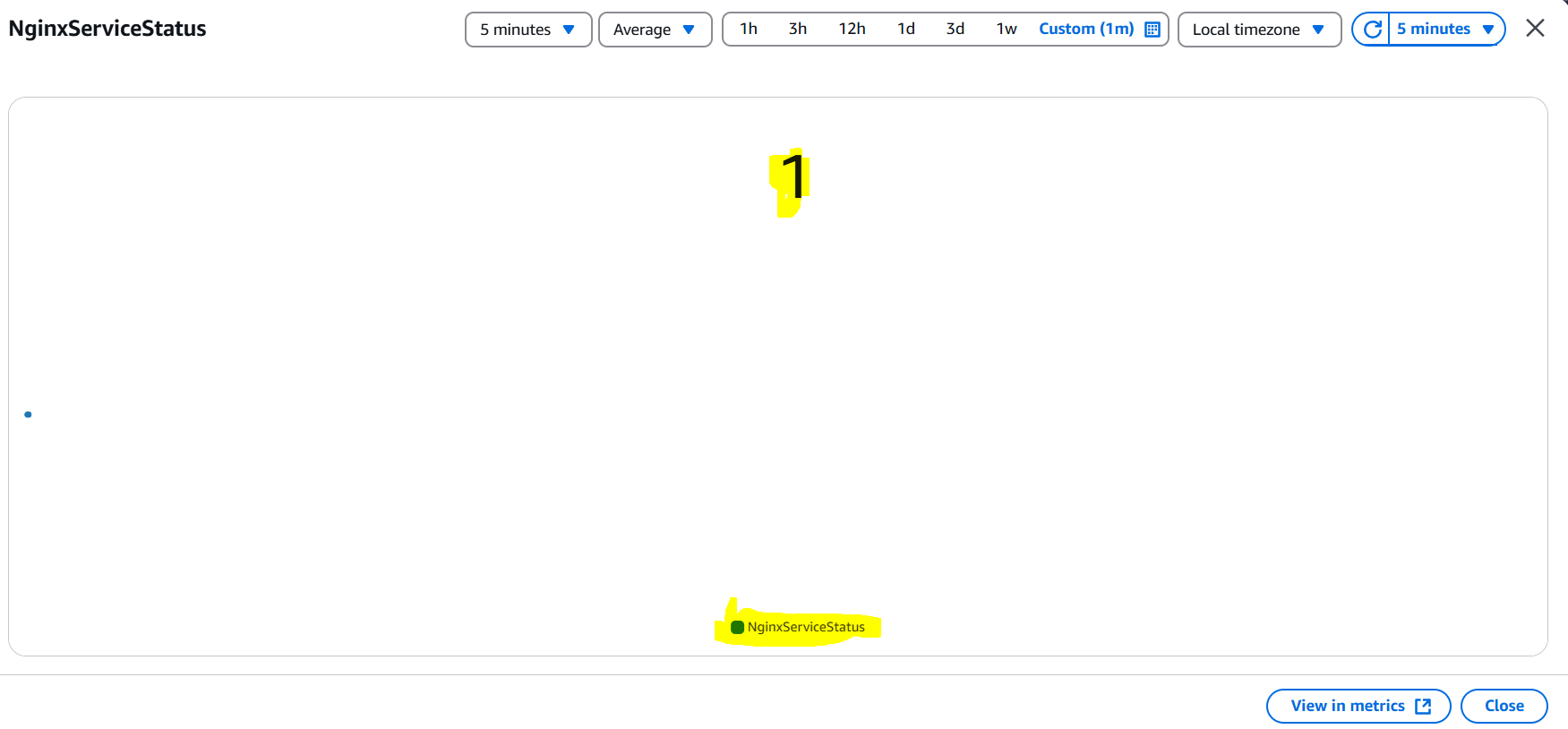
1. Go to the CloudWatch console.

2. Navigate to Dashboards.

3. Click Create dashboard.

4. Add widgets to display the NginxServiceStatus metric and alarm.

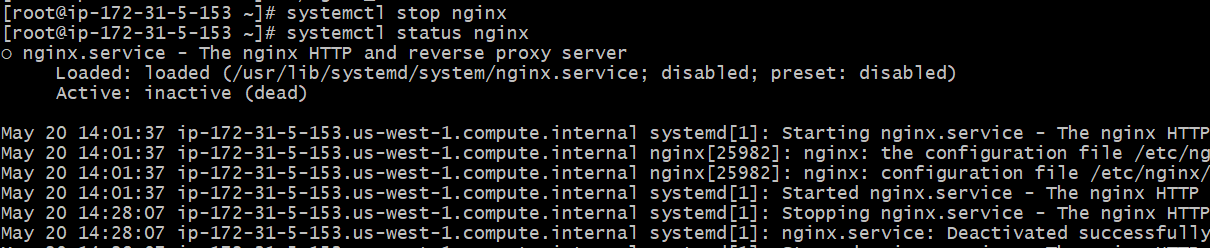




**Testing the Setup**

Stop the Nginx Service

sudo systemctl stop nginx, then run the script again. We can also run cronjob by using **```\* \* \* \* \* /root/nginx\_monitor.sh```** So there will be no need of running script gain and again it will run every minute.





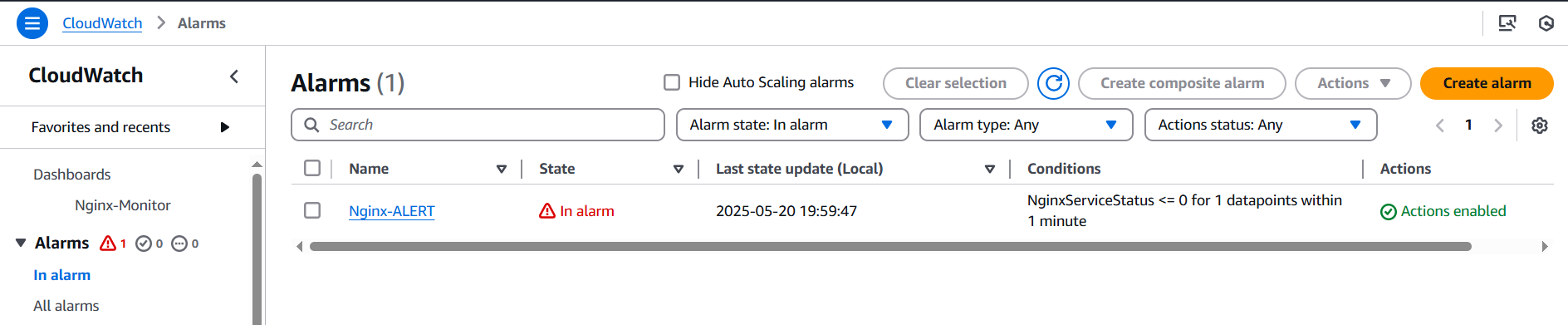
**Verify Alarm Trigger**

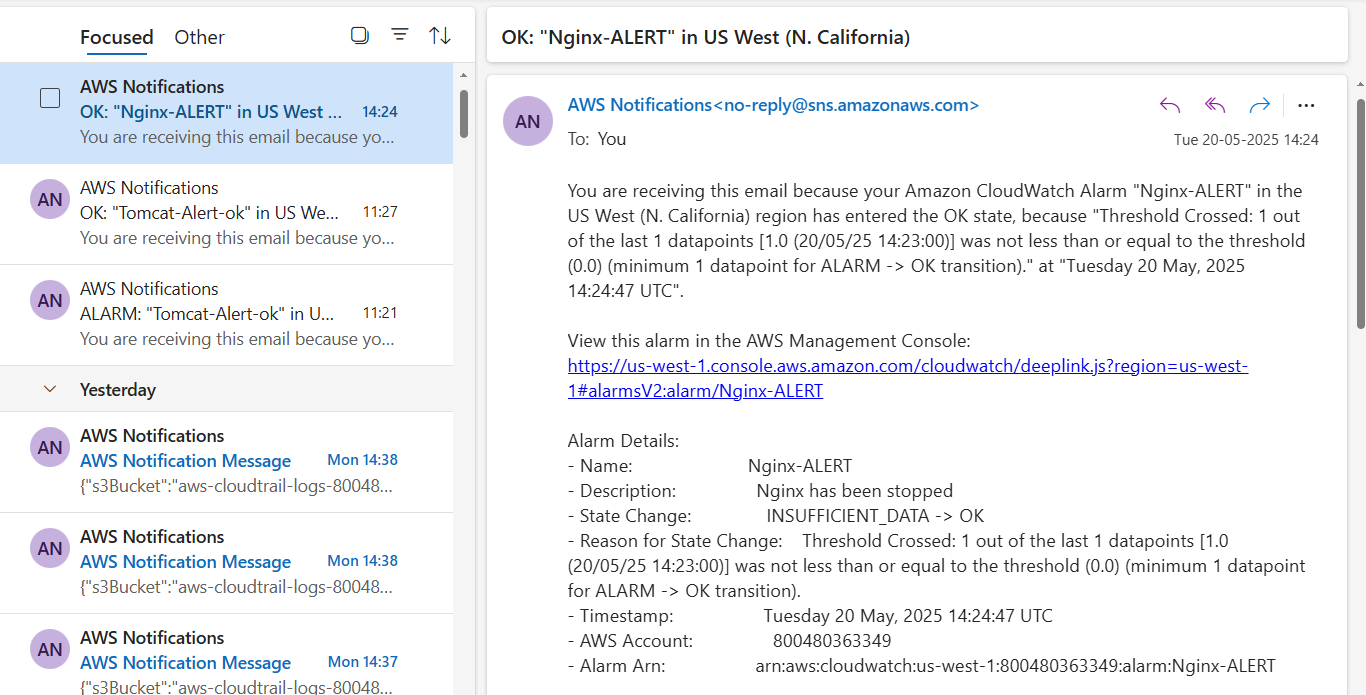
1. Wait for 1-2 minutes for the alarm to trigger.

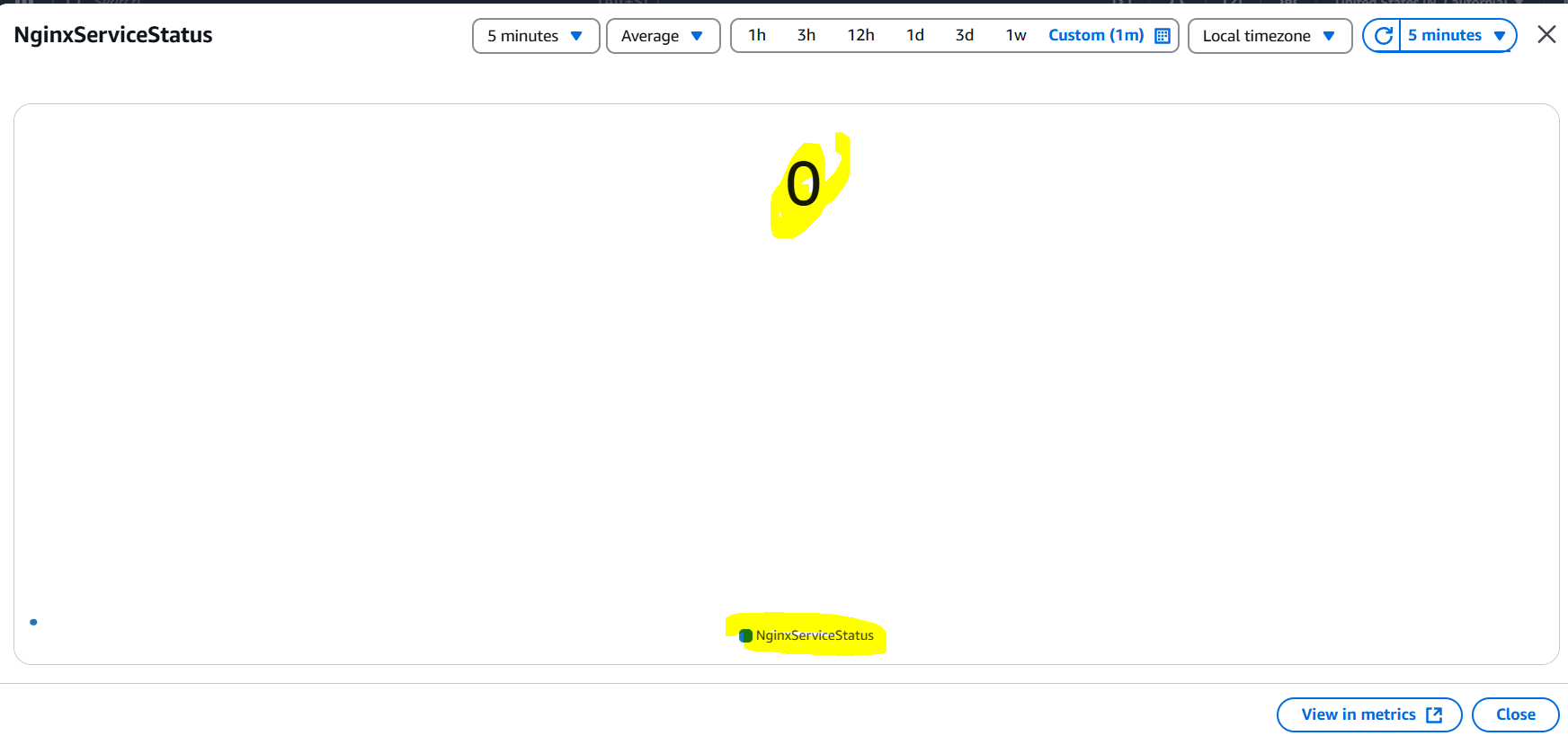
2. Go to the CloudWatch console.

3. Navigate to Alarms.

4. Verify that the alarm has transitioned to In Alarm state.







**The-End**