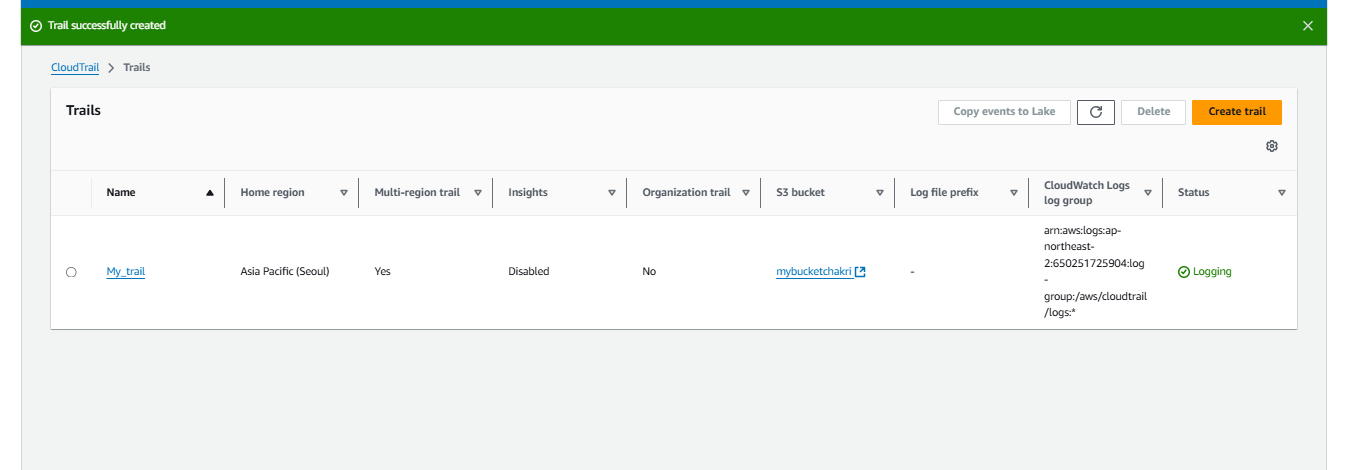
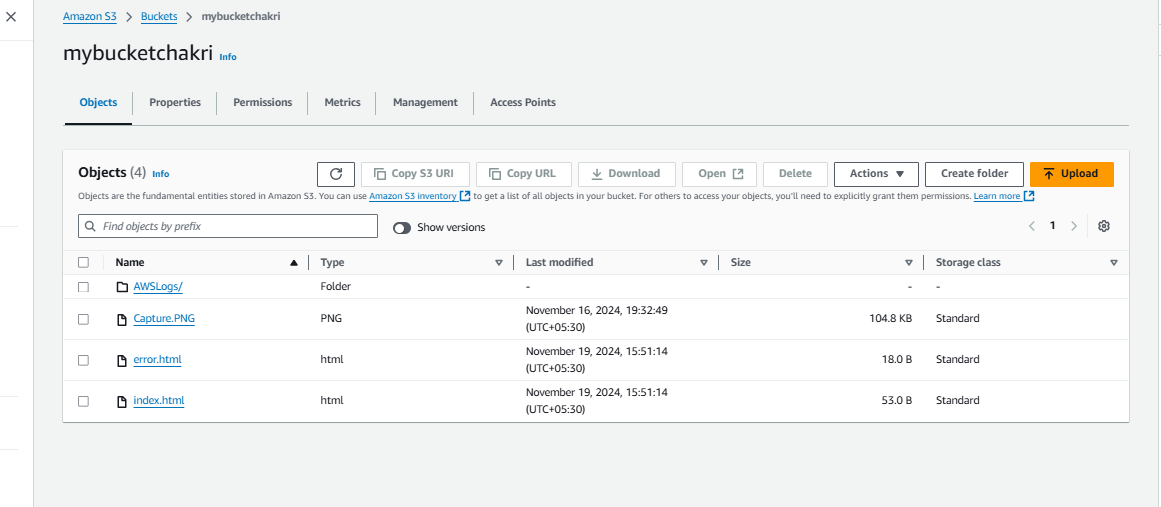
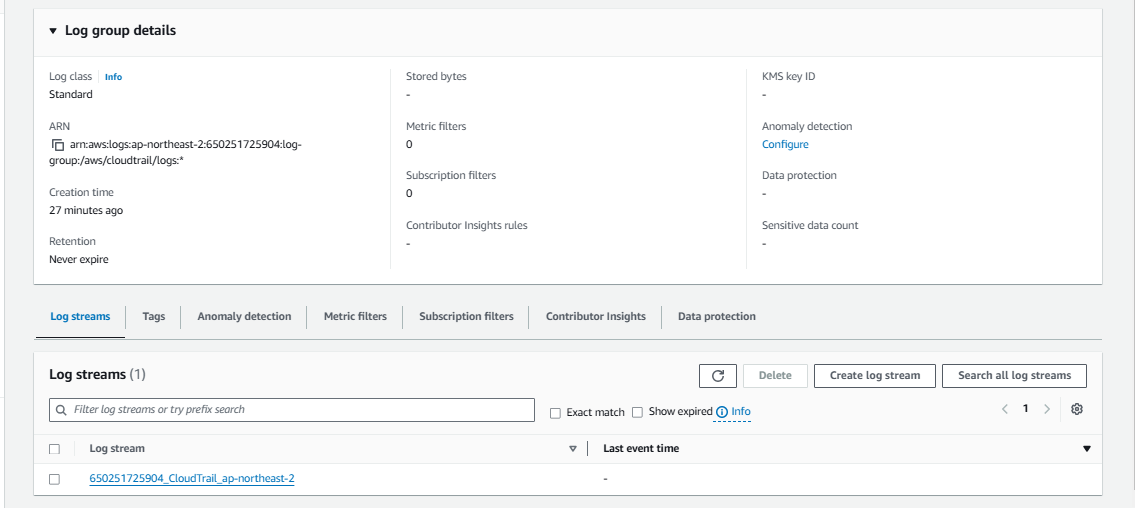
Cloud Watch And Cloud Trail

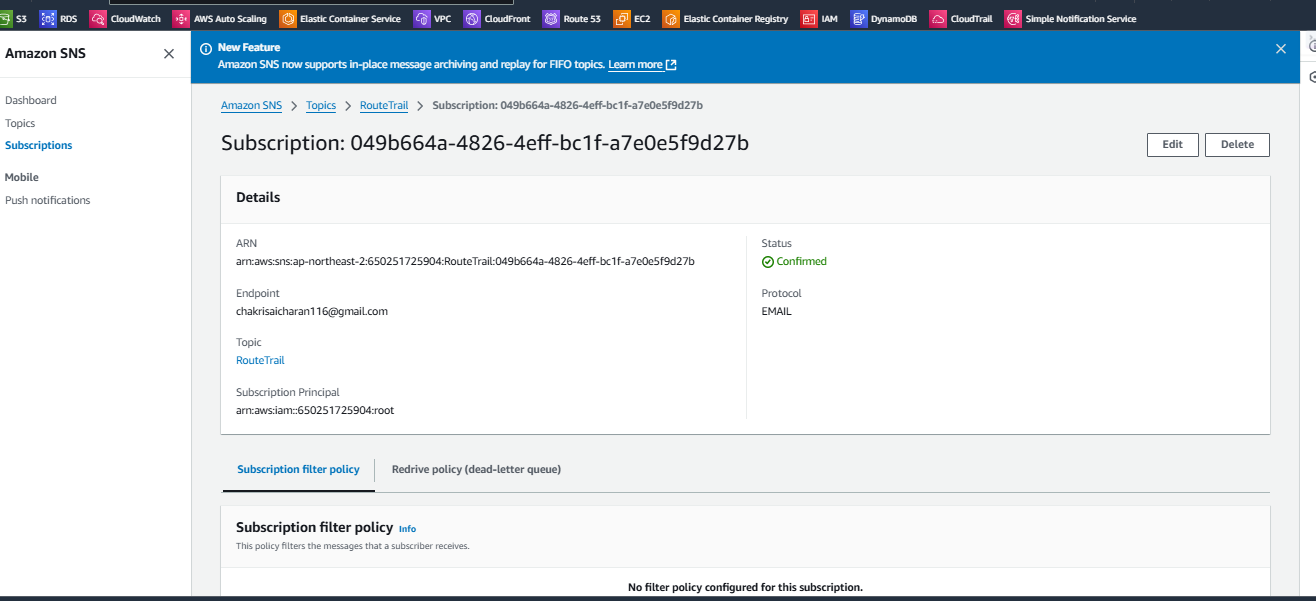
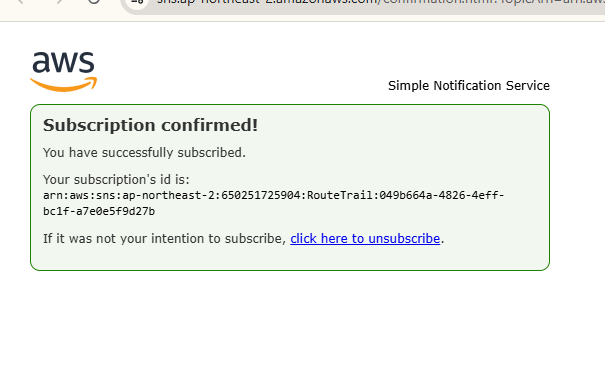
1. Enable cloudtrail monitoring and store the events in s3 and cloudwatch log events.



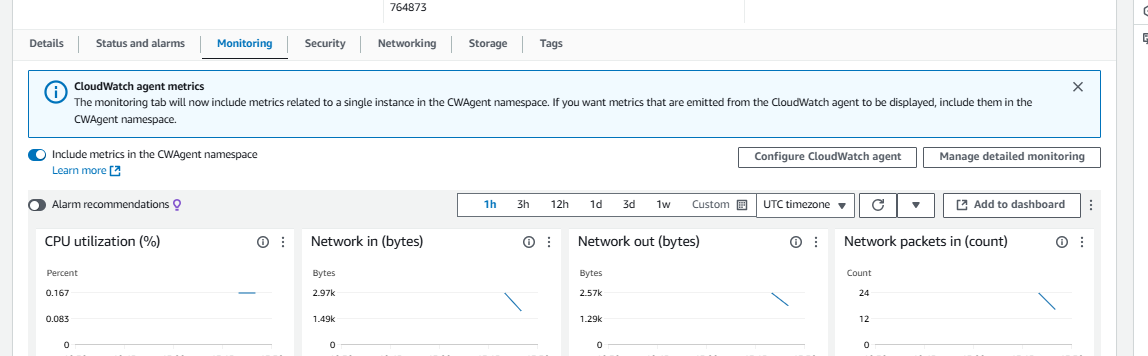


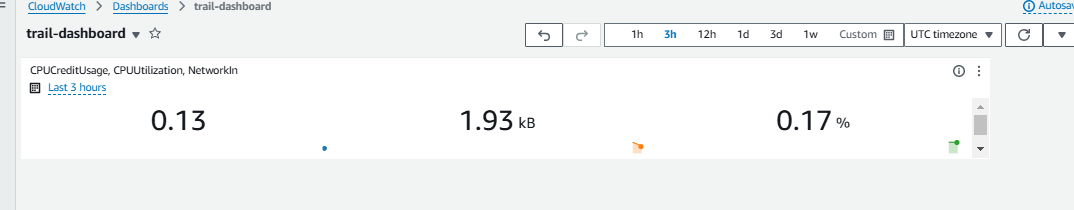


1. Enable SNS for cloudtrial to send alert on email.

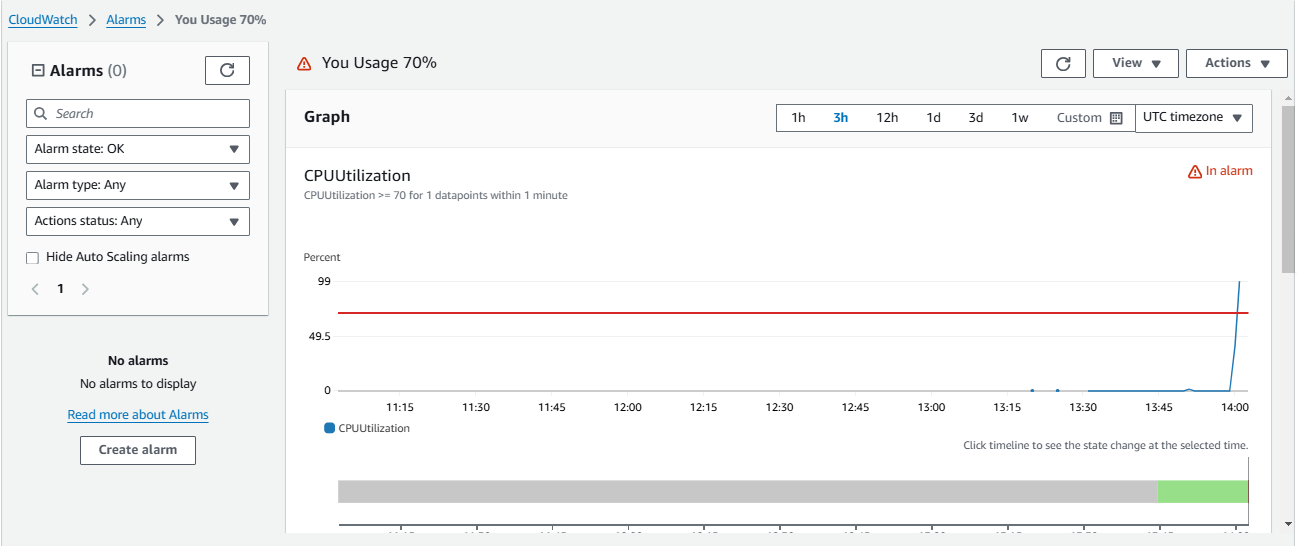
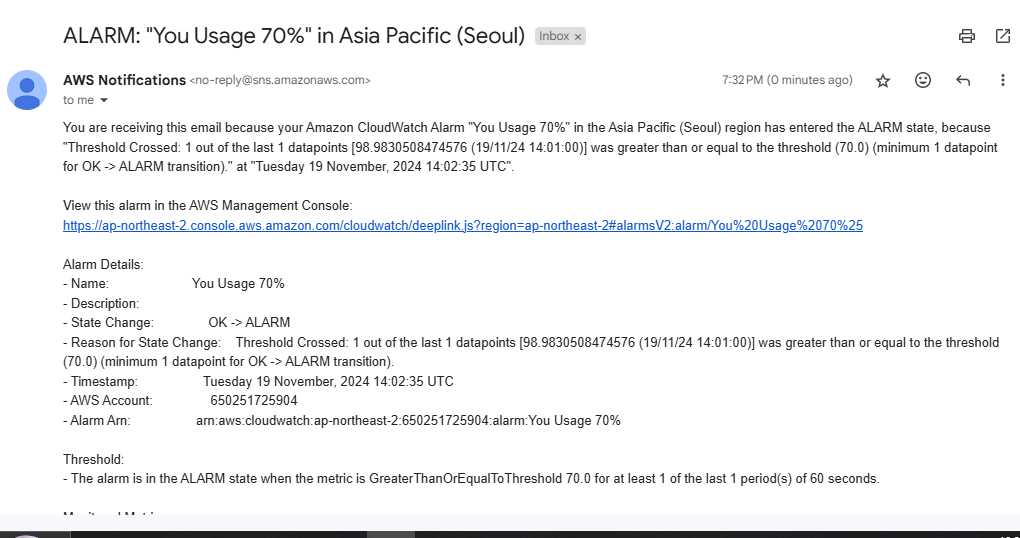


1. Configure cloud watch monitoring and record the cpu utilization and other metrics of ec2.





1. Create one alarm to send alert to email if the cpu utilization is more than 70 percent.



1. Create Dashboard and monitor tomcat service wether it is running or not and send the alert.

**login to instance and install java and tomcat**

**wirte a bash script**

#!/bin/bash

# Variables

DASHBOARD\_NAME="TomcatMonitoringDashboard"

ALARM\_NAME="TomcatServiceAlarm"

TOPIC\_NAME="TomcatServiceAlert"

TOPIC\_EMAIL="syedfs97@outlook.com"

INSTANCE\_ID="i-0f97671fe115ecf24" # Replace with your EC2 instance ID

# Create CloudWatch Dashboard

aws cloudwatch put-dashboard --dashboard-name $DASHBOARD\_NAME --dashboard-body '{

"widgets": [

{

"type": "metric",

"x": 0,

"y": 0,

"width": 24,

"height": 6,

"properties": {

"metrics": [

[ "AWS/EC2", "CPUUtilization", "InstanceId", "'$INSTANCE\_ID'" ]

],

"period": 300,

"stat": "Average",

"region": "us-east-1",

"title": "EC2 CPU Utilization"

}

},

{

"type": "metric",

"x": 0,

"y": 6,

"width": 24,

"height": 6,

"properties": {

"metrics": [

[ "AWS/EC2", "StatusCheckFailed", "InstanceId", "'$INSTANCE\_ID'" ]

],

"period": 300,

"stat": "Average",

"region": "us-east-1",

"title": "EC2 Status Check Failed"

}

}

]

}'

# Create SNS Topic

TOPIC\_ARN=$(aws sns create-topic --name $TOPIC\_NAME --query 'TopicArn' --output text)

# Subscribe to SNS Topic

aws sns subscribe --topic-arn $TOPIC\_ARN --protocol email --notification-endpoint $TOPIC\_EMAIL

# Create CloudWatch Alarm

aws cloudwatch put-metric-alarm --alarm-name $ALARM\_NAME --metric-name StatusCheckFailed --namespace AWS/EC2 --statistic Average --period 300 --threshold 1 --comparison-operator GreaterThanOrEqualToThreshold --dimensions Name=InstanceId,Value=$INSTANCE\_ID --evaluation-periods 1 --alarm-actions $TOPIC\_ARN --insufficient-data-actions $TOPIC\_ARN --ok-actions $TOPIC\_ARN

echo "CloudWatch Dashboard and Alarm setup complete. Check your email for subscription confirmation."

1. Create Dashboard and monitor nginx service to send the alert if nginx is not running.

**login to instance and install niginx**

**wirte a bash script**

#!/bin/bash

# Variables

DASHBOARD\_NAME="NginxMonitoringDashboard"

ALARM\_NAME="NginxServiceAlarm"

TOPIC\_NAME="NginxServiceAlert"

TOPIC\_EMAIL="syedfs97@outlook.com"

INSTANCE\_ID="i-0f97671fe115ecf24" # Replace with your EC2 instance ID

# Create CloudWatch Dashboard

aws cloudwatch put-dashboard --dashboard-name $DASHBOARD\_NAME --dashboard-body '{

"widgets": [

{

"type": "metric",

"x": 0,

"y": 0,

"width": 24,

"height": 6,

"properties": {

"metrics": [

[ "AWS/EC2", "CPUUtilization", "InstanceId", "'$INSTANCE\_ID'" ]

],

"period": 300,

"stat": "Average",

"region": "us-east-1",

"title": "EC2 CPU Utilization"

}

},

{

"type": "metric",

"x": 0,

"y": 6,

"width": 24,

"height": 6,

"properties": {

"metrics": [

[ "AWS/EC2", "StatusCheckFailed", "InstanceId", "'$INSTANCE\_ID'" ]

],

"period": 300,

"stat": "Average",

"region": "us-east-1",

"title": "EC2 Status Check Failed"

}

}

]

}'

# Create SNS Topic

TOPIC\_ARN=$(aws sns create-topic --name $TOPIC\_NAME --query 'TopicArn' --output text)

# Subscribe to SNS Topic

aws sns subscribe --topic-arn $TOPIC\_ARN --protocol email --notification-endpoint $TOPIC\_EMAIL

# Create CloudWatch Alarm for Nginx

aws cloudwatch put-metric-alarm --alarm-name $ALARM\_NAME --metric-name StatusCheckFailed --namespace AWS/EC2 --statistic Average --period 300 --threshold 1 --comparison-operator GreaterThanOrEqualToThreshold --dimensions Name=InstanceId,Value=$INSTANCE\_ID --evaluation-periods 1 --alarm-actions $TOPIC\_ARN --insufficient-data-actions $TOPIC\_ARN --ok-actions $TOPIC\_ARN

# Create a custom metric for Nginx status

cat <<EOL > nginx\_status.sh

#!/bin/bash

if systemctl is-active --quiet nginx; then

aws cloudwatch put-metric-data --metric-name NginxStatus --namespace CustomMetrics --value 1 --dimensions InstanceId=$INSTANCE\_ID

else

aws cloudwatch put-metric-data --metric-name NginxStatus --namespace CustomMetrics --value 0 --dimensions InstanceId=$INSTANCE\_ID

fi

EOL

chmod +x nginx\_status.sh

# Schedule the script to run every minute using cron

(crontab -l 2>/dev/null; echo "\* \* \* \* \* /path/to/nginx\_status.sh") | crontab –