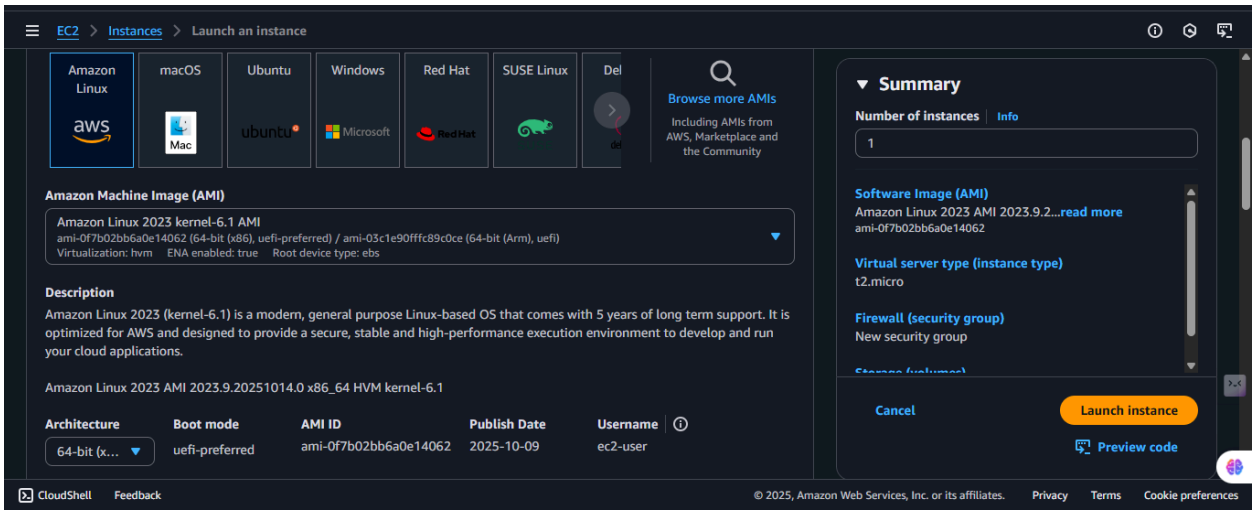
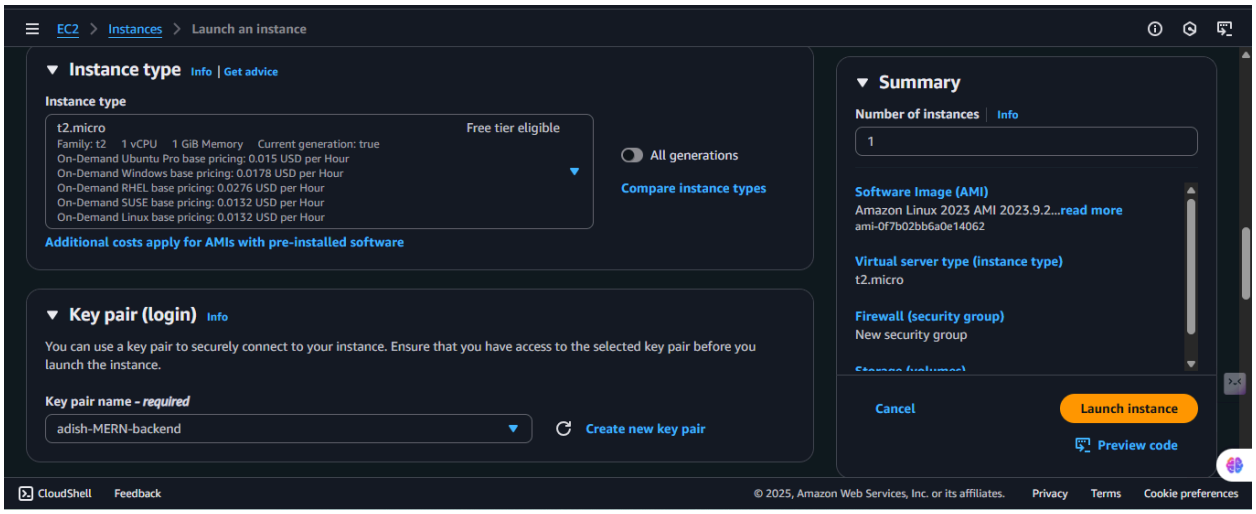
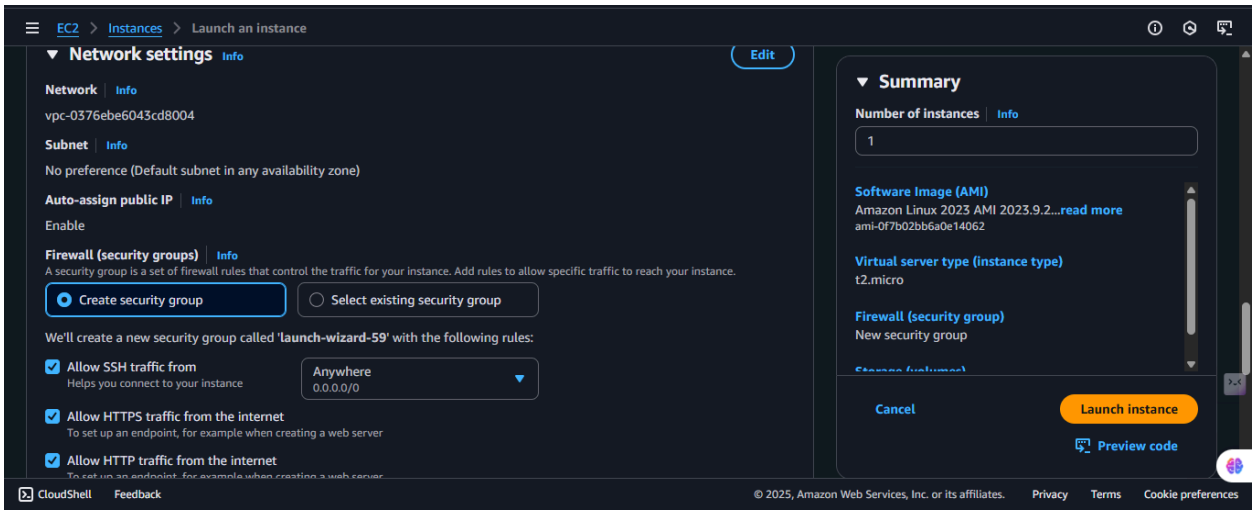
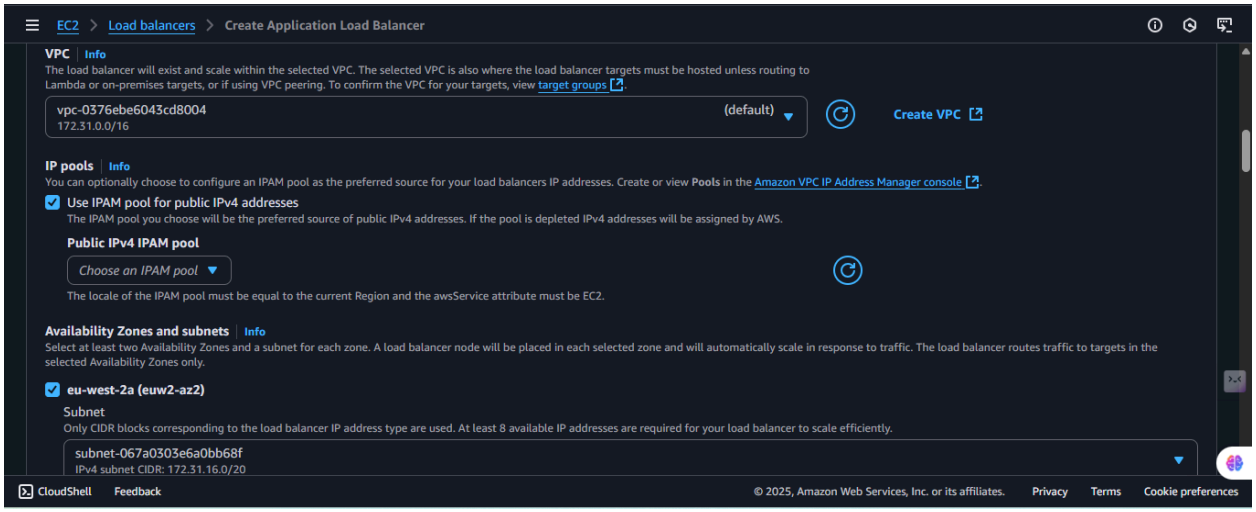
1:- Create EC2 Instance

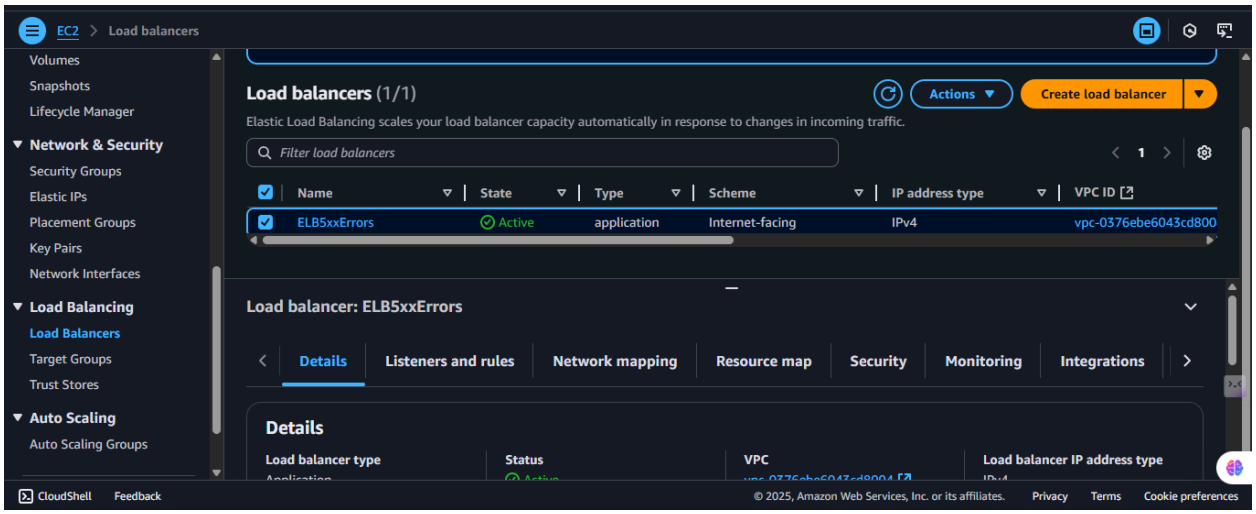




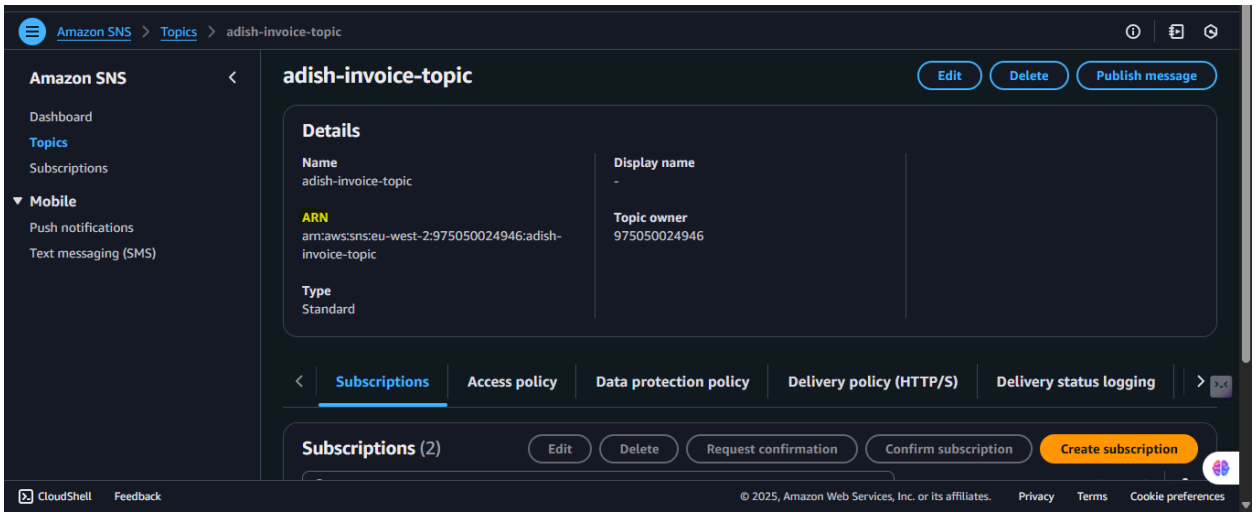


2:- Create ALB Load Balancer

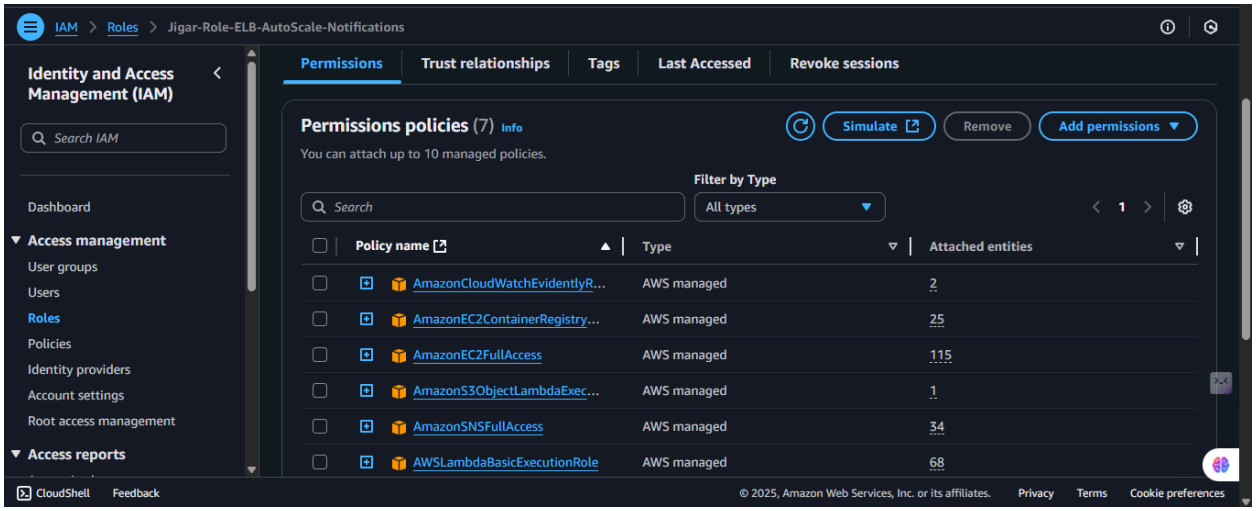




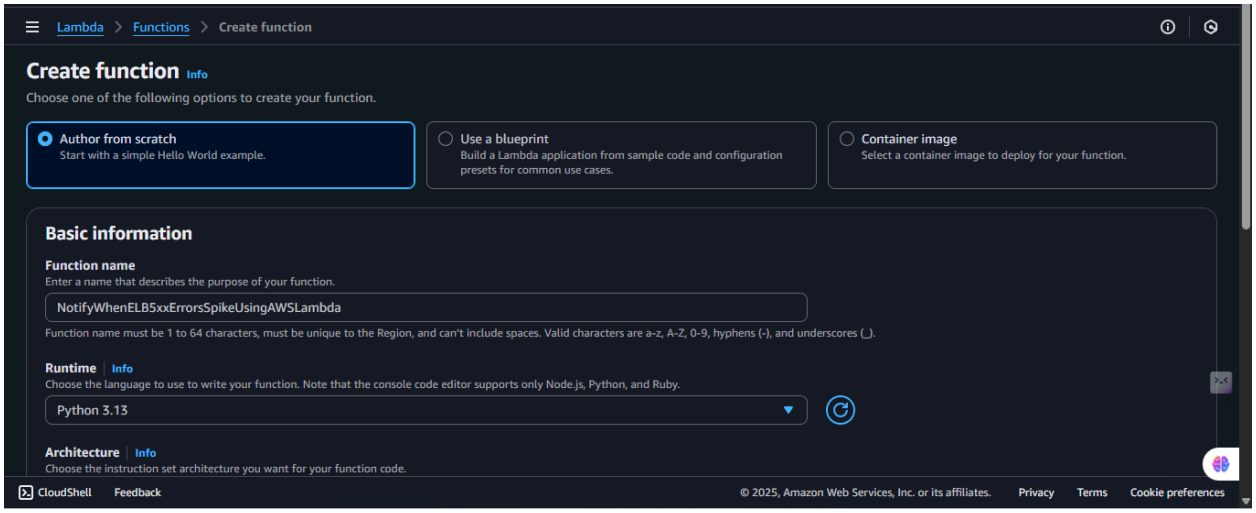
3 :- Create SNS

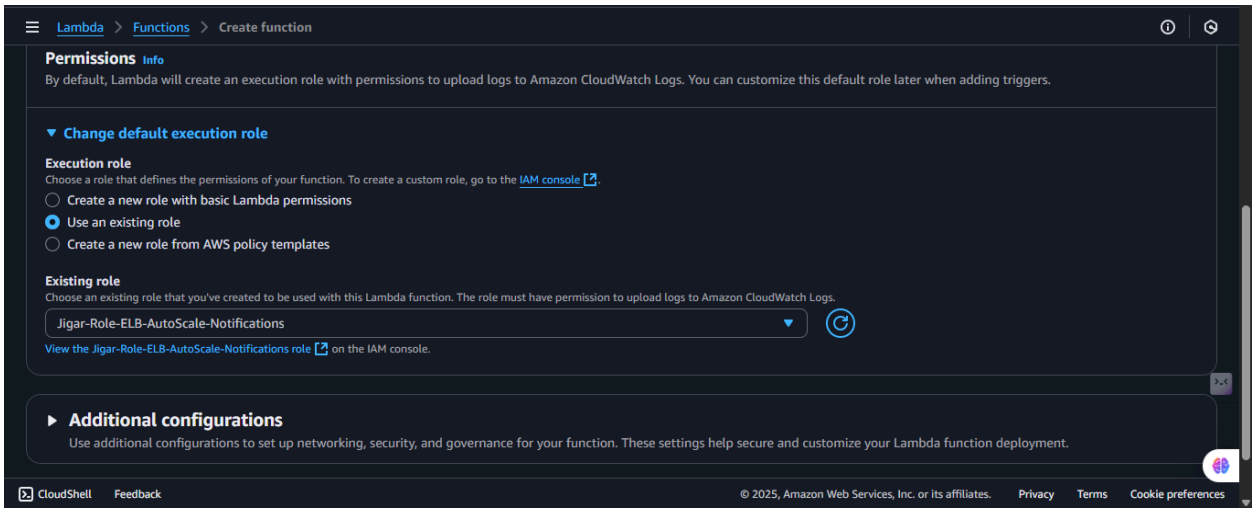


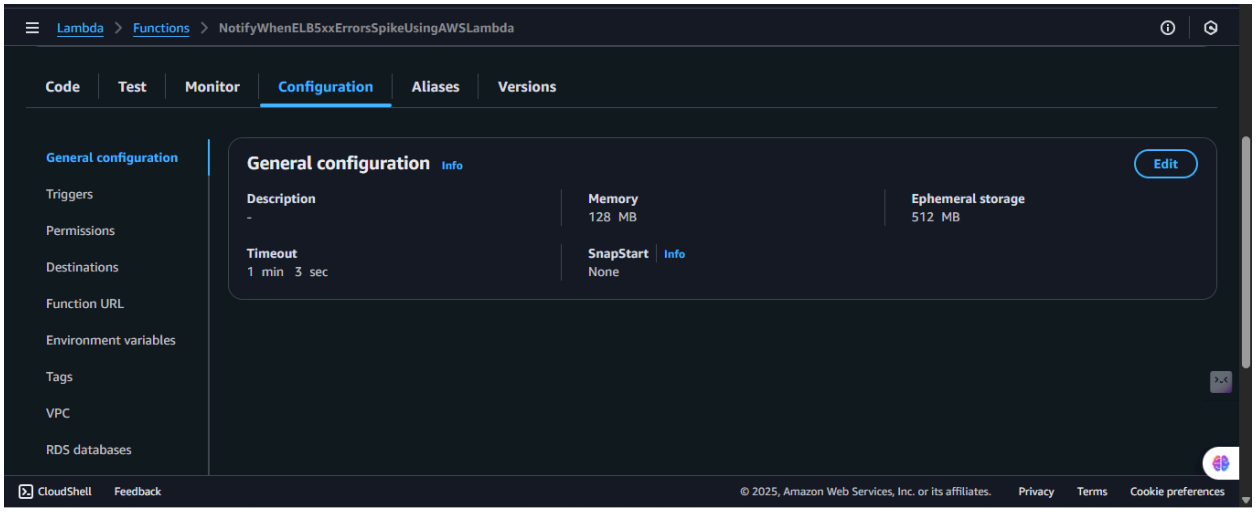
4:- Create IAM Role

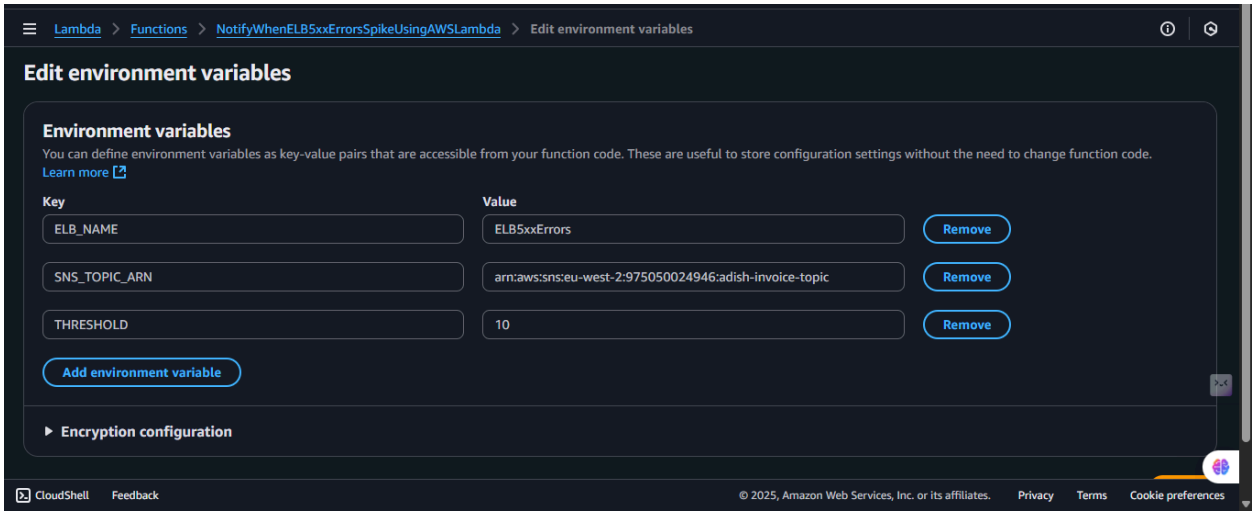


5:- Create Lambda Function



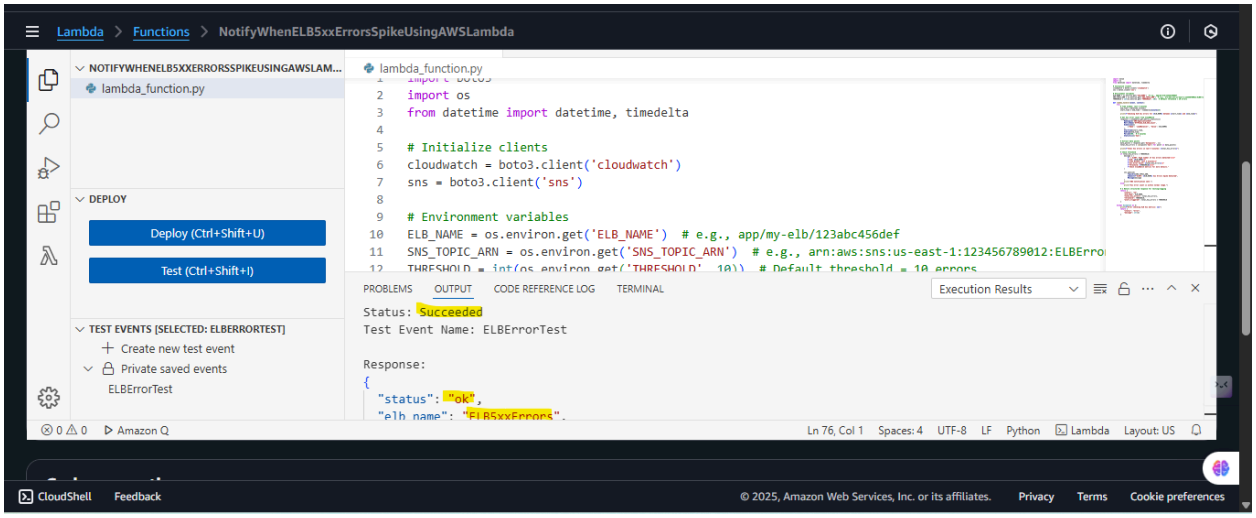






Environment Variables

Variable Name	Example Value	Description
`ELB_NAME`	`app/my-elb/123abc456def`	Your Load Balancer name (from AWS console)
`SNS_TOPIC_ARN`	`arn:aws:sns:us-east-1:123456789012:ELBErrorAlerts`	SNS Topic ARN
`THRESHOLD`	`10`	5xx error threshold for triggering alert




```
import boto3
import os
from datetime import datetime, timedelta

# Initialize clients
cloudwatch = boto3.client('cloudwatch')
sns = boto3.client('sns')

# Environment variables
ELB_NAME = os.environ.get('ELB_NAME')  # e.g., app/my-elb/123abc456def
SNS_TOPIC_ARN = os.environ.get('SNS_TOPIC_ARN')  # e.g.,
arn:aws:sns:us-east-1:123456789012:ELBErrorAlerts
THRESHOLD = int(os.environ.get('THRESHOLD', 10))  # Default threshold = 10
errors

def lambda_handler(event, context):
    try:
        # Time window: last 5 minutes
        end_time = datetime.utcnow()
        start_time = end_time - timedelta(minutes=5)

        print(f"Checking ELB 5xx errors for {ELB_NAME} between
{start_time} and {end_time}")

        # Get 5xx error count from CloudWatch
        response = cloudwatch.get_metric_statistics(
            Namespace='AWS/ApplicationELB',
```

```
        MetricName='HTTPCode_ELB_5XX_Count',
        Dimensions=[
            {'Name': 'LoadBalancer', 'Value': ELB_NAME}
        ],
        StartTime=start_time,
        EndTime=end_time,
        Period=300,  # 5 minutes
        Statistics=['Sum']
    )

    # Extract data points
    data_points = response.get('Datapoints', [])
    total_5xx_errors = sum(point['Sum'] for point in data_points)

    print(f"Total 5xx errors in last 5 minutes: {total_5xx_errors}")

    # Check threshold
    if total_5xx_errors > THRESHOLD:
        message = (
            f"⚠️ ALERT: High number of 5xx errors detected!\n\n"
            f"ELB: {ELB_NAME}\n"
            f"Time Window: Last 5 minutes\n"
            f"5xx Error Count: {total_5xx_errors}\n"
            f"Threshold: {THRESHOLD}\n\n"
            f"Check CloudWatch metrics for more details."
        )

        sns.publish(
            TopicArn=SNS_TOPIC_ARN,
            Subject=f"ALERT: {ELB_NAME} 5xx Errors Spike Detected",
            Message=message
        )
        print("SNS notification sent.")
    else:
        print("5xx error count is within normal range.")

    # ✅ Return structured response for testing/logging
    return {
        "status": "ok",
        "elb_name": ELB_NAME,
```

```
            "total_5xx_errors": total_5xx_errors,
            "threshold": THRESHOLD,
            "alert_triggered": total_5xx_errors > THRESHOLD
        }

    except Exception as e:
        print(f"Error checking ELB 5xx metrics: {e}")
        return {
            "status": "error",
            "message": str(e)
        }
```