

1:- Create EC2 Instance

The screenshots show the process of creating an EC2 instance on the AWS Management Console.

Step 1: Selecting the AMI

The first screenshot shows the 'Amazon Machine Image (AMI)' selection step. A box highlights the 'Amazon Linux' option, which is selected. Other options like macOS, Ubuntu, Windows, Red Hat, SUSE Linux, and Deep Learning are shown. The summary panel indicates 1 instance, using the Amazon Linux 2023 kernel-6.1 AMI (ami-0f7b02bb6a0e14062), with a t2.micro instance type, and a New security group. The 'Launch instance' button is highlighted.

Step 2: Selecting the Instance Type

The second screenshot shows the 'Instance type' selection step. The t2.micro instance type is selected. The summary panel remains the same, showing 1 instance using the same AMI and security group.

Step 3: Network Settings

The third screenshot shows the 'Network settings' step. It includes fields for Network (vpc-0376ebe6043cd8004), Subnet (No preference), Auto-assign public IP (Enable checked), and Firewall (security groups). Under Firewall, the 'Create security group' button is selected. The summary panel remains consistent.

2:- Create ALB Load Balancer

The screenshot shows the 'Create Application Load Balancer' wizard. In the 'VPC' step, a VPC is selected: 'vpc-0376be6043cd8004'. The 'IP pools' step is shown next, with the 'Use IPAM pool for public IPv4 addresses' checkbox checked. The 'Availability Zones and subnets' step follows, showing 'eu-west-2a (euw2-az2)' selected. The final step, 'Subnet', lists a single subnet: 'subnet-067a0503e6a0b68f' (IPv4 subnet CIDR: 172.31.16.0/20). The bottom of the screen includes standard AWS navigation links like CloudShell and Feedback.

The screenshot shows the 'Load balancers' list page. A single load balancer named 'ELB5xxErrors' is listed, showing it is active, application type, Internet-facing, IPv4, and associated with VPC 'vpc-0376be6043cd8004'. The left sidebar shows navigation for Network & Security, Load Balancing, and Auto Scaling. The bottom of the screen includes standard AWS navigation links like CloudShell and Feedback.

3 :- Create SNS

The screenshot shows the 'Topics' list page for an SNS topic named 'adish-invoice-topic'. The topic details are: Name: adish-invoice-topic, ARN: arn:aws:sns:eu-west-2:975050024946:adish-invoice-topic, and Type: Standard. Two subscriptions are listed under 'Subscriptions'. The bottom of the screen includes standard AWS navigation links like CloudShell and Feedback.

4:- Create IAM Role

The screenshot shows the AWS IAM Roles page. The left sidebar has sections for Dashboard, Access management (User groups, Users, Roles, Policies, Identity providers, Account settings, Root access management), and Access reports. The main area is titled 'Permissions policies (7)'. It includes a search bar, a filter by type dropdown set to 'All types', and a table with columns for Policy name, Type, and Attached entities. The policies listed are: AmazonCloudWatchEvidentlyRole, AmazonEC2ContainerRegistryRole, AmazonEC2FullAccess, AmazonS3ObjectLambdaExecutionRole, AmazonSNSFullAccess, and AWSLambdaBasicExecutionRole.

5:- Create Lambda Function

The screenshot shows the 'Create function' wizard. Step 1: Basic information. It offers three options: 'Author from scratch' (selected), 'Use a blueprint', and 'Container image'. The 'Author from scratch' section contains a text input field with the value 'NotifyWhenELB5xxErrorsSpikeUsingAWSLambda'. The 'Runtime' section shows 'Python 3.13' selected. The 'Architecture' section shows 'lambda@edge' selected. The bottom navigation bar includes CloudShell and Feedback.

The screenshot shows the 'Create function' wizard. Step 2: Permissions info. It states that Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. It includes sections for 'Change default execution role' (with 'Use an existing role' selected), 'Execution role' (with 'Create a new role with basic Lambda permissions' selected), and 'Existing role' (with 'Jigar-Role-ELB-AutoScale-Notifications' selected). The bottom navigation bar includes CloudShell and Feedback.

Lambda > Functions > NotifyWhenELB5xxErrorsSpikeUsingAWSLambda

Code | Test | Monitor | Configuration | Aliases | Versions

General configuration

Description: -

Memory: 128 MB

Ephemeral storage: 512 MB

Timeout: 1 min 3 sec

SnapStart: None

Edit

Triggers
Permissions
Destinations
Function URL
Environment variables
Tags
VPC
RDS databases

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Lambda > Functions > NotifyWhenELB5xxErrorsSpikeUsingAWSLambda > Edit environment variables

Edit environment variables

Environment variables

You can define environment variables as key-value pairs that are accessible from your function code. These are useful to store configuration settings without the need to change function code.

Learn more [Edit](#)

Key	Value	Action
ELB_NAME	ELB5xxErrors	Remove
SNS_TOPIC_ARN	arn:aws:sns:eu-west-2:975050024946:adish-invoice-topic	Remove
THRESHOLD	10	Remove

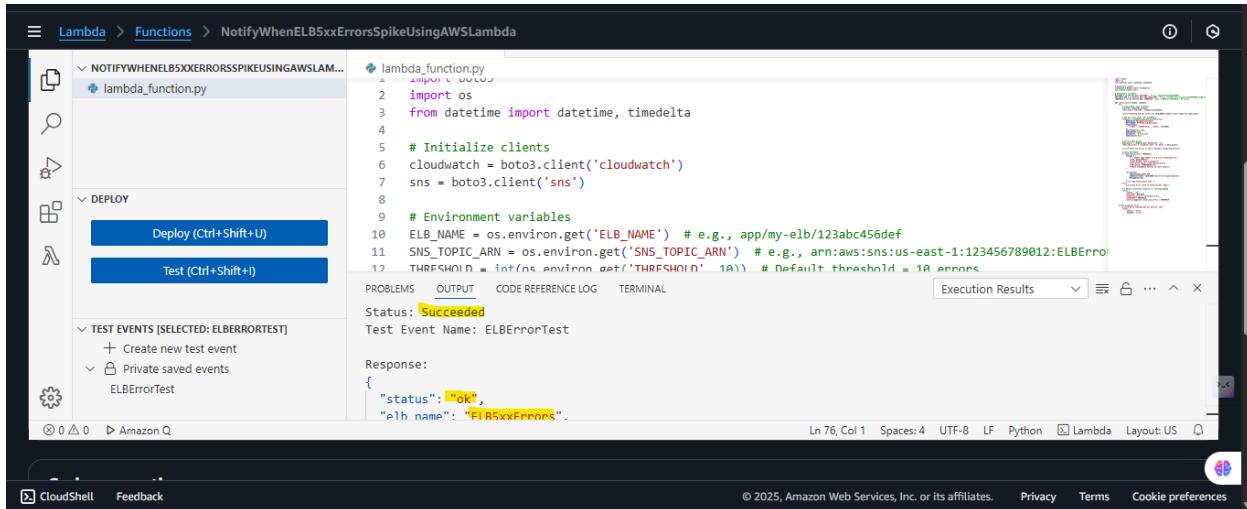
Add environment variable

Encryption configuration

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Environment Variables

Variable Name	Example Value	Description
ELB_NAME	app/my-elb/123abc456def	Your Load Balancer name (from AWS console)
SNS_TOPIC_A	arn:aws:sns:us-east-1:1234567890	SNS Topic ARN
RN	12:ELBErrorAlerts	
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)
}

```

The screenshot shows the AWS CloudWatch Log Groups interface. The left sidebar navigation includes CloudWatch, Favorites and recents, Dashboards, Alarms, Logs (selected), Log Anomalies, Live Tail, Logs Insights, Contributor Insights, Metrics, and Application Signals (APM). The main content area is titled 'Log events' and displays log entries. A search bar at the top says 'Filter events - press enter to search'. Below it are time range controls (1m, 1h, UTC timezone). The log entries are timestamped and show messages related to ELB 5xx errors:

- Timestamp: 2025-10-21T10:05:24.022Z, Message: Checking ELB 5xx errors for ELB5xxErrors between 2025-10-21 10:00:23.753967 and 2025-10-21 10:05:23.753967
- Timestamp: 2025-10-21T10:05:24.022Z, Message: Total 5xx errors in last 5 minutes: 0
- Timestamp: 2025-10-21T10:05:24.022Z, Message: Total 5xx errors in last 5 minutes: 0
- Timestamp: 2025-10-21T10:05:24.022Z, Message: 5xx error count is within normal range.
- Timestamp: 2025-10-21T10:05:24.022Z, Message: 5xx error count is within normal range.

At the bottom of the page, there are links for CloudShell, Feedback, and a footer with copyright information: © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences.