

## Project Name: Automated EC2 Health Check with AWS Lambda

### Objective:

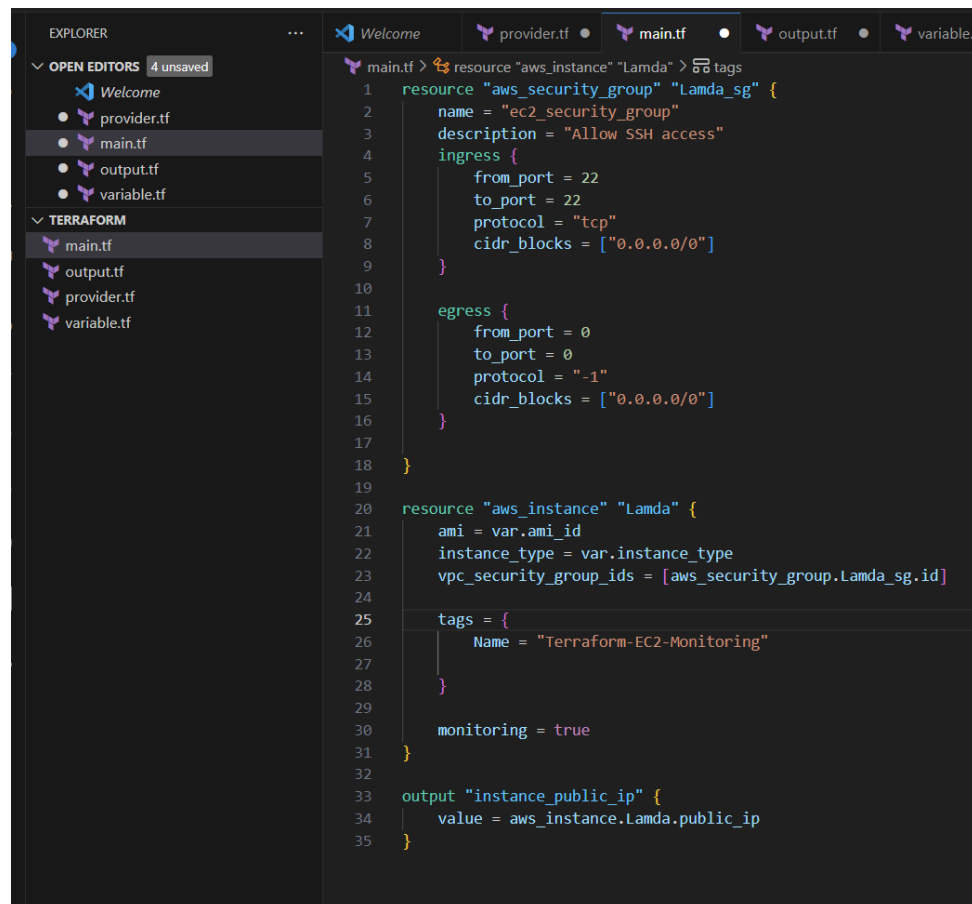
Build a serverless solution using AWS Lambda to automatically monitor the health of EC2 instances and notify the team in case of issues via Amazon SNS (Simple Notification Service).

### Step1 (Create using Terraform)

- Launch one or more EC2 instances
- Ensure you have basic monitoring enabled (CloudWatch metrics)

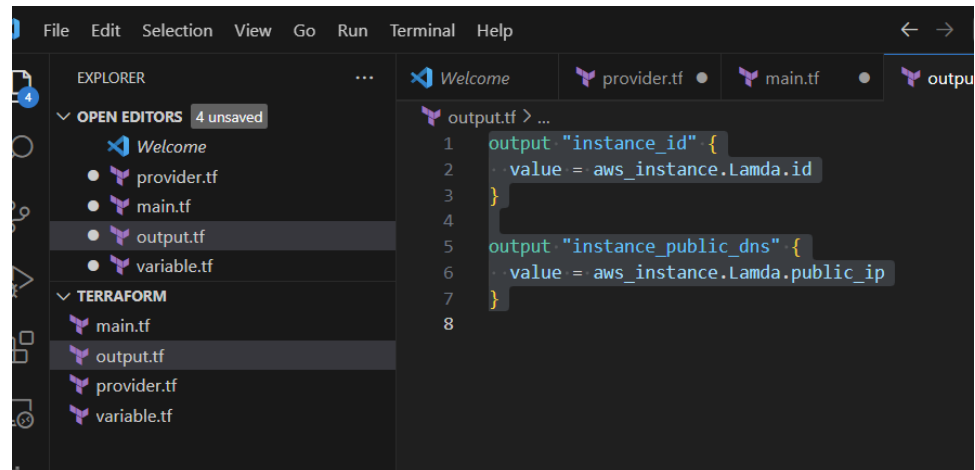
### Terraform Code

- Main. tf

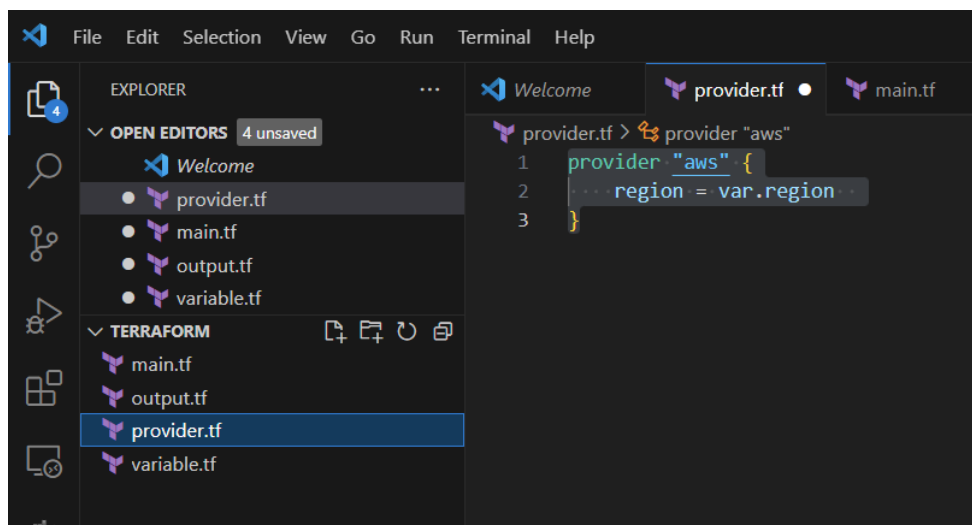


```
1 resource "aws_security_group" "Lamda_sg" {
2   name = "ec2_security_group"
3   description = "Allow SSH access"
4   ingress {
5     from_port = 22
6     to_port = 22
7     protocol = "tcp"
8     cidr_blocks = ["0.0.0.0/0"]
9   }
10
11   egress {
12     from_port = 0
13     to_port = 0
14     protocol = "-1"
15     cidr_blocks = ["0.0.0.0/0"]
16   }
17 }
18
19
20 resource "aws_instance" "Lamda" {
21   ami = var.ami_id
22   instance_type = var.instance_type
23   vpc_security_group_ids = [aws_security_group.Lamda_sg.id]
24
25   tags = {
26     Name = "Terraform-EC2-Monitoring"
27   }
28
29   monitoring = true
30 }
31
32
33 output "instance_public_ip" {
34   value = aws_instance.Lamda.public_ip
35 }
```

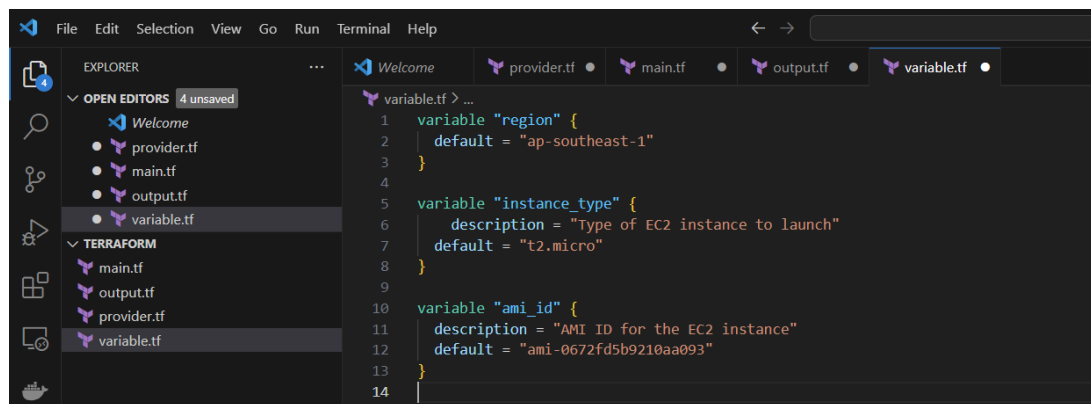
- Output.tf



- Provider.tf



- Variable.tf



## Terraform command

- aws configure
- terraform init
- terraform fmt
- terraform plan
- terraform apply

```
ubuntu@ip-172-31-35-90: ~/terraform/ec2
ubuntu@ip-172-31-35-90:~/terraform/ec2$ ll
total 66120
drwxrwxr-x 4 ubuntu ubuntu    4096 Jan 27 11:03 ./
drwxrwxr-x 3 ubuntu ubuntu    4096 Jan 27 09:01 ../
drwxr-xr-x 3 ubuntu ubuntu    4096 Jan 27 10:31 .terraform/
-rw-r--r-- 1 ubuntu ubuntu    1377 Jan 27 10:31 .terraform.lock.hcl
drwxr-xr-x 3 ubuntu ubuntu    4096 Jan 24 19:15 aws/
-rw-rw-r-- 1 ubuntu ubuntu 67651287 Jan 27 09:08 awscliv2.zip
-rw-rw-r-- 1 ubuntu ubuntu    671 Jan 27 10:53 main.tf
-rw-rw-r-- 1 ubuntu ubuntu    131 Jan 27 10:23 output.tf
-rw-rw-r-- 1 ubuntu ubuntu    41 Jan 27 10:39 provider.tf
-rw-rw-r-- 1 ubuntu ubuntu   7154 Jan 27 10:54 terraform.tfstate
-rw-rw-r-- 1 ubuntu ubuntu    181 Jan 27 10:53 terraform.tfstate.backup
-rw-rw-r-- 1 ubuntu ubuntu    268 Jan 27 10:46 variable.tf
ubuntu@ip-172-31-35-90:~/terraform/ec2$
```

<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
<input type="checkbox"/>	Terraform	i-0001d71a36c0a352	Running	t2.micro	2/2 checks passed	View alarms +	ap-southeast-1b	ec2-47.129.152.152.ap...	47.129.152.152	-
<input checked="" type="checkbox"/>		i-038fee49b159f414d	Terminated	t2.micro	-	View alarms +	ap-southeast-1a	-	-	-
<input type="checkbox"/>	Terraform-EC2-Monitoring	i-0064a3e530a0002db	Running	t2.micro	Initializing	View alarms +	ap-southeast-1a	ec2-18.142.252.200.ap...	18.142.252.200	-

## Step2 (Create AWS Lamda Function)

- Create AWS Lamda Function

AWS

🔍

[Alt+S]

📄

🔔

👤

⚙️

Asia Pacific (Singapore)

sattitva

☰

[Lambda](#) > [Functions](#) > Create function

🔍

🌐

## Create function info

Choose one of the following options to create your function.

☒ **Author from scratch**  
Start with a simple Hello World example.

☐ **Use a blueprint**  
Build a Lambda application from sample code and configuration presets for common use cases.

☐ **Container image**  
Select a container image to deploy for your function.

### Basic information

**Function name** info  
Enter a name that describes the purpose of your function.

Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (\_).

**Runtime** info  
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

🔻 🔄

**Architecture** info  
Choose the instruction set architecture you want for your function code.

☒ **x86\_64**  
☐ arm64

## Permissions [Info](#)

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

### ▼ Change default execution role

#### Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

- ☒ Create a new role with basic Lambda permissions
- ☐ Use an existing role
- ☐ Create a new role from AWS policy templates

① Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.

Lambda will create an execution role named MyDemoLambda-role-m8l09lfd, with permission to upload logs to Amazon CloudWatch Logs.

## Step3 (Create IAM Role for Lamda)

- AmazonEC2ReadOnlyAccess (to get EC2 instance statuses).
- AmazonSNSFullAccess (to publish notifications to SNS).

Step 2

Add permissions

Step 3

Name, review, and create

### Trusted entity type

☒ **AWS service**  
Allow AWS services like EC2, Lambda, or others to perform actions in this account.

☐ **AWS account**  
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

☐ **Web identity**  
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.

☐ **SAML 2.0 federation**  
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.

☐ **Custom trust policy**  
Create a custom trust policy to enable others to perform actions in this account.

### Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

**Service or use case**

Lambda

Choose a use case for the specified service.

**Use case**

☒ **Lambda**  
Allows Lambda functions to call AWS services on your behalf.

Cancel

Next

Step 1

Select trusted entity

**Step 2**

Add permissions

Step 3

Name, review, and create

## Add permissions [Info](#)

Permissions policies (2/1030) [Info](#)

Choose one or more policies to attach to your new role.

AmazonSNSFullAccess

Filter by Type

All types

1 match

<input checked="" type="checkbox"/>	Policy name <a href="#">?</a>	Type	Description
<input checked="" type="checkbox"/>	AmazonSNSFullAccess	AWS managed	Provides full access to Amazon SNS via...

► Set permissions boundary - optional

Cancel

Previous

Next

[IAM](#) > [Roles](#) > Create role

**Step 1: Select trusted entities** EDIT

**Trust policy**

```

1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": [
7         "sts:AssumeRole"
8       ],
9       "Principal": {
10        "Service": [
11          "lambda.amazonaws.com"
12        ]
13      }
14    ]
15  }
16 }

```

**Step 2: Add permissions** EDIT

**Permissions policy summary**

Policy name	Type	Attached as
<a href="#">AmazonEC2ReadOnlyAccess</a>	AWS managed	Permissions policy
<a href="#">AmazonSNSFullAccess</a>	AWS managed	Permissions policy

**Step 3: Add tags**

**Add tags - optional** [info](#)

Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

[Add new tag](#)

You can add up to 50 more tags.

[Cancel](#) [Previous](#) [Create role](#)

## Step4 (Attached the Role to Lamda Function)

[Code](#) [Test](#) [Monitor](#) [Configuration](#) [Aliases](#) [Versions](#)

**General configuration**

Triggers

**Permissions**

Destinations

Function URL

Environment variables

Tags

VPC

RDS databases

**Execution role** EDIT [View role document](#)

**Role name**

[LambdaDemo-role-h53n9wok](#)

**Resource summary**

To view the resources and actions that your function has permission to access, choose a service.

[Amazon CloudWatch Logs](#)  
3 actions, 2 resources

**By action** **By resource**

Resource	Actions
----------	---------

**Ephemeral storage** [info](#)

You can configure up to 10 GB of ephemeral storage (/tmp) for your function. [View pricing](#)

512 MB

Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

**SnapStart** [info](#)

Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the [SnapStart compatibility considerations](#). For Python and .NET runtimes, [view pricing](#).

None

Supported runtimes: .NET 8 (C#/.NET Core), Java 11, Java 17, Java 21, Python 3.12, Python 3.13.

**Timeout**

0 min 3 sec

**Execution role**

Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

☒ Use an existing role

☐ Create a new role from AWS policy templates

**Existing role**

Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

[Lambda-EC2-SNS-Role](#)

[View the Lambda-EC2-SNS-Role role](#) on the IAM console.

[Cancel](#) [Save](#)

## Step5 (Write the Lamda Function Code)

The screenshot shows the AWS Lambda console interface. At the top, a green notification bar states "Successfully updated the function LambdaDemo." Below this, the "Code" tab is selected, displaying the code source for the "LambdaDemo" function. The code is a Python file named "lambda\_function.py" with the following content:

```
10
11 def lambda_handler(event, context):
12     # Retrieve the EC2 instance statuses
13     response = ec2.describe_instance_status(includeAllInstances=True)
14
15     unhealthy_instances = []
16
17     for instance in response['InstanceStatuses']:
18         instance_id = instance['InstanceId']
19         state = instance['InstanceState']['Name']
20         status = instance['InstanceState']['Status']
21
22     # Check if the instance is stopped or has an impaired status
23     if state != 'running' or status == 'impaired':
24         unhealthy_instances.append((instance_id, state, status))
25
26     # If there are unhealthy instances, send an SNS notification
27     if unhealthy_instances:
28         message = "Unhealthy EC2 Instances Detected:\n"
29         for instance in unhealthy_instances:
30             message += f"Instance ID: {instance[0]}, State: {instance[1]}, Status: {instance[2]}\n"
31
32     sns.publish(
33         TopicArn=SNSTOPIC_ARN,
34         Subject="Unhealthy EC2 Instances Alert",
35         Message=message
```

The left sidebar shows the "EXPLORER" view with the "LAMBDADEMO" function selected. The "DEPLOY (UNDEPLOYED CHANGES)" section is visible, showing a "Deploy (Ctrl+Shift+D)" button and a "Test (Ctrl+Shift+T)" button. The "TEST EVENTS (NONE SELECTED)" section is also visible, with a "Create new test event" button.

## Step6 (Create CloudWatch Events to Trigger the Lamda)

The screenshot shows the Amazon EventBridge console interface. The "Rules" tab is selected, and the "Create rule" wizard is in progress. The "Define schedule" step is active, showing the "Schedule pattern" section. The "Schedule pattern" section has two options: "A fine-grained schedule that runs at a specific time, such as 8:00 a.m. PST on the first Monday of every month." and "A schedule that runs at a regular rate, such as every 10 minutes." The second option is selected. The "Rate expression" section shows a "rate" expression with a value of "5" and a unit of "Minutes". The "Next" button is highlighted in orange.

Amazon EventBridge > Rules > Create rule

**Amazon EventBridge**

- Dashboard [New](#)
- ▼ **Developer resources**
  - Learn
  - Sandbox
  - Quick starts
- ▼ **Buses**
  - Event buses
  - [Rules](#)
  - Global endpoints
  - Archives
  - Replays
- ▼ **Pipes**
  - Pipes
- ▼ **Scheduler**
  - Schedules
  - Schedule groups
- ▼ **Integration**
  - Partner event sources

**Steps:**

- Select target(s)
- Step 4 - optional: Configure tags
- Step 5: Review and create

**Target 1**

**Target types**  
Select an EventBridge event bus, EventBridge API destination (SaaS partner), or another AWS service as a target.

☐ EventBridge event bus  
☐ EventBridge API destination  
☒ AWS service

**Select a target** [Info](#)  
Select target(s) to invoke when an event matches your event pattern or when schedule is triggered (limit of 5 targets per rule)

Lambda function

**Target location**

☒ Target in this account  
☐ Target in another AWS account

**Function**  
LambdaDemo

► **Configure version/alias**

► **Additional settings**

[Add another target](#)
[Cancel](#)
[Skip to Review and create](#)
[Previous](#)
[Next](#)

## Step7 (Set Up SNS for Notifications)

- Create topic
- Create subscription

Amazon SNS > Subscriptions > Create subscription

**Topic ARN**  
arn:aws:sns:ap-southeast-1:529088293565:EC2HealthCheckAlerts

**Protocol**  
The type of endpoint to subscribe  
Email

**Endpoint**  
An email address that can receive notifications from Amazon SNS.  
satthya\_57@hotmail.com

After your subscription is created, you must confirm it. [Info](#)

► **Subscription filter policy - optional** [Info](#)  
This policy filters the messages that a subscriber receives.

► **Redrive policy (dead-letter queue) - optional** [Info](#)  
Send undeliverable messages to a dead-letter queue.

[Cancel](#)
[Create subscription](#)

## Step8 (Add the SNS topic ARN in the Lambda function's environment variables)

Successfully updated the function **LambdaDemo**.

Code | Test | Monitor | **Configuration** | Aliases | Versions

General configuration  
Triggers  
Permissions  
Destinations  
Function URL  
**Environment variables**  
Tags

**Environment variables (1)** Edit

The environment variables below are encrypted at rest with the default Lambda service key.

Key	Value
SNS_TOPIC_ARN	arn:aws:sns:ap-southeast-1:529088293565:EC2HealthCheckAlerts:165ff5c9-90d7-43e5-b399-65e700547305

## Step9 (Verify)

Folders

- Inbox 1
- Junk Email 29
- Drafts 21
- Sent Items
- Deleted Items 176
- Archive
- Notes
- Conversation Histo...
- Trash 282
- Go to Groups

Unhealthy EC2 Instances Alert

EC2HealthCheckAlerts<no-reply@sns.amazonaws.com>  
To: You

Unhealthy EC2 Instances Detected:  
Instance ID: i-0739660d67f2df2f4, State: terminated, Status: not-applicable  
Instance ID: i-067385ea7e56f11c9, State: stopping, Status: not-applicable

--

If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:  
[https://sns.ap-southeast-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:ap-southeast-1:529088293565:EC2HealthCheckAlerts:165ff5c9-90d7-43e5-b399-65e700547305&Endpoint=satthya\\_57@hotmail.com](https://sns.ap-southeast-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:ap-southeast-1:529088293565:EC2HealthCheckAlerts:165ff5c9-90d7-43e5-b399-65e700547305&Endpoint=satthya_57@hotmail.com)

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at <https://aws.amazon.com/support>

Reply Reply all Forward ...  
Tue 1/28/2025 4:00 PM