

# Project-1



## Automated EC2 Lifecycle Management with CloudFormation and Lambda

By: Shivansh Mishra | July 11, 2025

---



### Project Description

This project demonstrates how to automate the lifecycle of AWS EC2 instances using a combination of AWS CloudFormation and AWS Lambda. The goal is to provision infrastructure as code and manage it efficiently by automatically cleaning up unused resources.



### Project Overview

This project showcases how to automate the lifecycle of EC2 instances using AWS services:

1. **Create** resources (EC2 + Security Group) using CloudFormation.
  2. **Manually stop** the EC2 instances to simulate idle use.
  3. **Use a Lambda function** (Python-based) to check which EC2 instances have been stopped for over **5 minutes**, and **automatically delete** them.
  4. **Run Lambda manually** (no CloudWatch trigger).
- 



### Step-by-Step Process

---

#### Step 1: Create EC2 Instances Using CloudFormation

1. Open AWS Console → Go to CloudFormation
  - Click "Create Stack" → With new resources (standard)
2. Upload or paste this template:

```

AWSTemplateFormatVersion: "2010-09-09"
Description: Launch an EC2 instance using CloudFormation

Parameters:
  KeyName:
    Description: Name of an existing EC2 KeyPair to enable SSH access
    Type: AWS::EC2::KeyPair::KeyName

Resources:
  MyEC2SecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Enable SSH access
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: 0.0.0.0/0

  MyEC2Instance1:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: t2.micro
      KeyName: !Ref KeyName
      ImageId: ami-0ff8a91507f77f867
      SubnetId: subnet-082759e4f2285b7b2
      SecurityGroupIds:
        - !GetAtt MyEC2SecurityGroup.GroupId
    Tags:
      - Key: Name
        Value: MyEC2Instance1

  MyEC2Instance2:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: t2.micro
      KeyName: !Ref KeyName
      ImageId: ami-0ff8a91507f77f867
      SubnetId: subnet-082759e4f2285b7b2
      SecurityGroupIds:
        - !GetAtt MyEC2SecurityGroup.GroupId
    Tags:
      - Key: Name
        Value: MyEC2Instance2

  MyEC2Instance3:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: t2.micro
      KeyName: !Ref KeyName
      ImageId: ami-0ff8a91507f77f867
      SubnetId: subnet-082759e4f2285b7b2
      SecurityGroupIds:
        - !GetAtt MyEC2SecurityGroup.GroupId
    Tags:
      - Key: Name
        Value: MyEC2Instance3

Outputs:
  Instance1Id:
    Description: ID of EC2 instance 1
    Value: !Ref MyEC2Instance1

  Instance2Id:
    Description: ID of EC2 instance 2
    Value: !Ref MyEC2Instance2

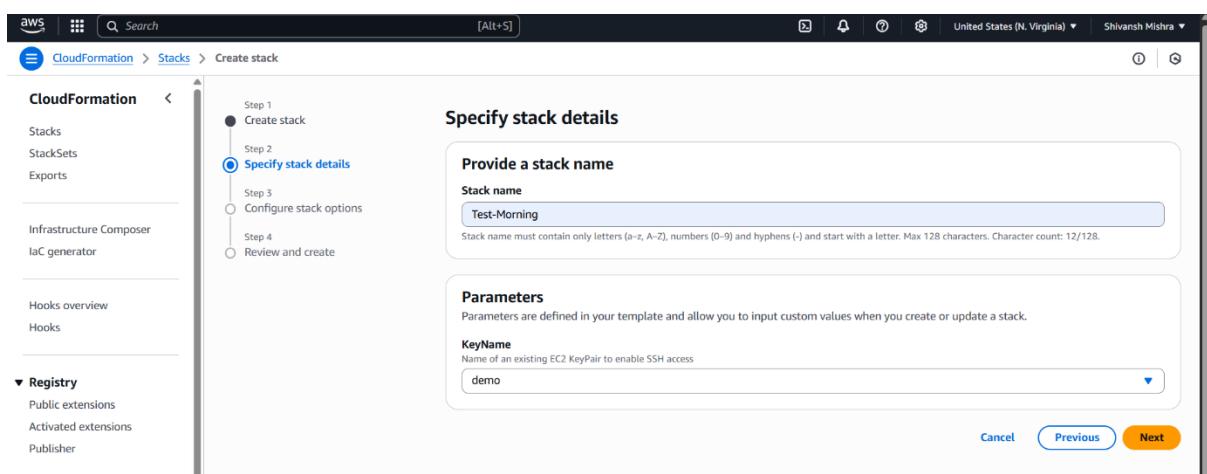
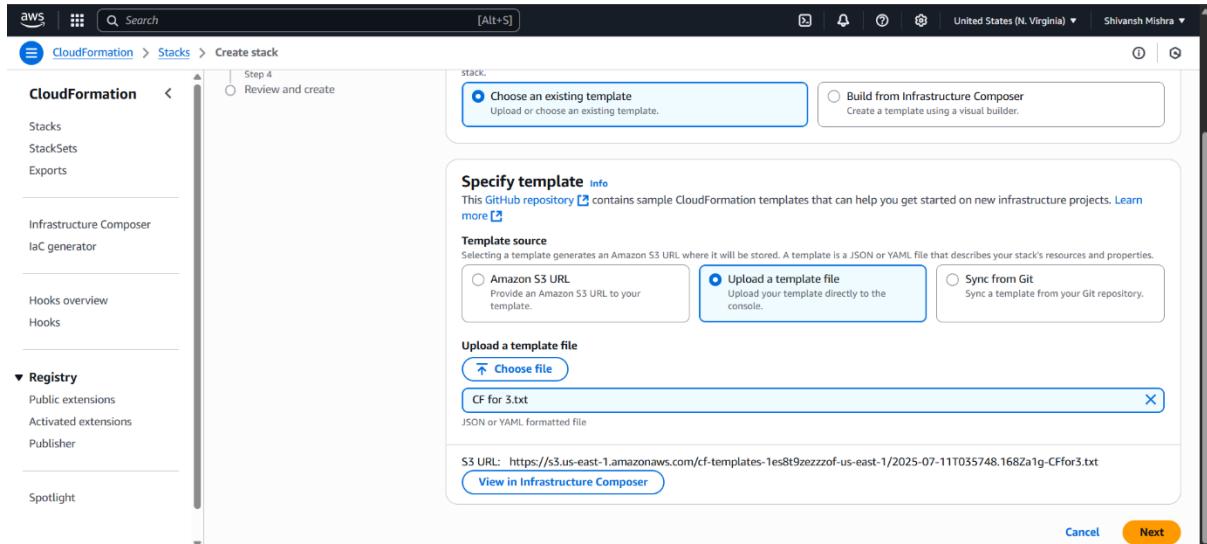
  Instance3Id:
    Description: ID of EC2 instance 3
    Value: !Ref MyEC2Instance3

```

💡 Replace subnet-xxxxxxxx with a valid subnet ID in your VPC.

### 3. Create the stack

- Give a name like EC2AutoDeleteStack
- Select your **Key Pair**
- Click **Next** → Leave options default → **Create Stack**



The screenshot shows the AWS CloudFormation console with a stack named 'Test-Morning' created on 2025-07-11 at 09:27:15 UTC+0530. The 'Events' tab is active, showing 14 events. All events are marked as 'CREATE\_COMPLETE'.

Timestamp	Logical ID	Status	Detailed status
2025-07-11 09:27:42 UTC+0530	Test-Morning	CREATE_COMPLETE	-
2025-07-11 09:27:41 UTC+0530	MyEC2Instance1	CREATE_COMPLETE	-
2025-07-11 09:27:41 UTC+0530	MyEC2Instance3	CREATE_COMPLETE	-
2025-07-11 09:27:40 UTC+0530	MyEC2Instance2	CREATE_COMPLETE	-

## ✓ Step 2: Stop the EC2 Instances

After the stack is created:

1. Go to **EC2 Console**.
2. Select all 3 instances → Click "Instance State" → **Stop instance**.
3. Wait for them to fully enter the "**stopped**" state.

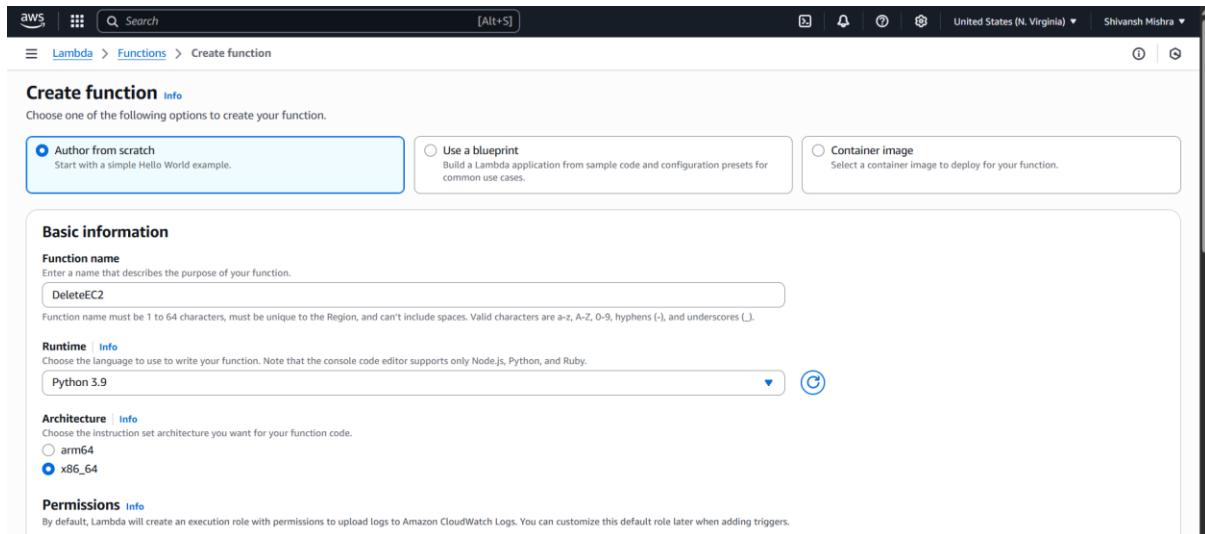
The screenshot shows the AWS EC2 Instances page with three instances listed: MyEC2Instance2, MyEC2Instance3, and MyEC2Instance1. All three instances are currently marked as 'Stopping'.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
MyEC2Instance2	i-0a4add5578af40b07	Stopping	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-
MyEC2Instance3	i-02949469b2a5d7f6	Stopping	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-
MyEC2Instance1	i-038d58911fd63d735	Stopped	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-

## ✓ Step 3: Create the Lambda Function

### 1. Go to AWS Console → Search Lambda → Click “Create Function”

- **Name:** DeleteStoppedEC2
- **Runtime:** Python 3.9
- **Permissions:** Choose **Create a new role with basic Lambda permissions**



## 2. After function is created, open the Code tab and paste this:

```

import boto3
from datetime import datetime, timezone, timedelta

def lambda_handler(event, context):
    ec2 = boto3.client('ec2')

    # Get all stopped EC2 instances
    response = ec2.describe_instances(
        Filters=[{'Name': 'instance-state-name', 'Values': ['stopped']}]
    )

    instances_to_terminate = []

    for reservation in response['Reservations']:
        for instance in reservation['Instances']:
            instance_id = instance['InstanceId']
            state_transition_reason = instance.get('StateTransitionReason', '')

            # Format: 'User initiated (2025-07-11 05:10:45 GMT)'
            if 'User initiated' in state_transition_reason:
                try:
                    timestamp_str = state_transition_reason.split('(')[-1].strip(')')
                    stopped_time = datetime.strptime(timestamp_str, '%Y-%m-%d %H:%M:%S %Z').replace(tzinfo=timezone.utc)
                    current_time = datetime.now(timezone.utc)

                    if current_time - stopped_time > timedelta(minutes=3):
                        instances_to_terminate.append(instance_id)
                except Exception as e:
                    print(f"Error parsing time for instance {instance_id}: {e}")

    if instances_to_terminate:
        print(f"Terminating instances: {instances_to_terminate}")
        ec2.terminate_instances(InstanceIds=instances_to_terminate)
    else:
        print("No stopped instances older than 5 minutes.")

```

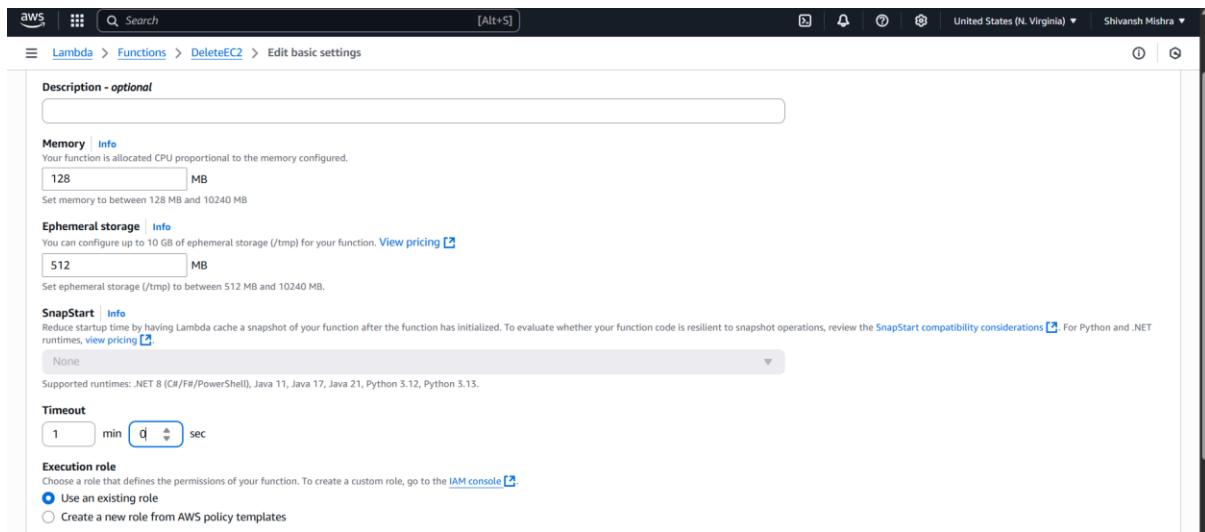
## 3. Click Deploy

### Step 4: Increase Timeout

1. Go to Configuration → General configuration

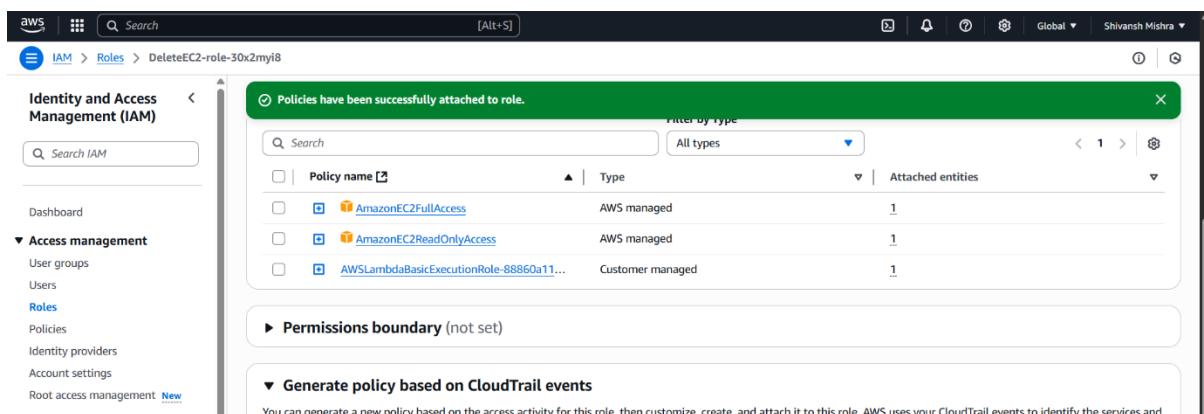
2. Click Edit → Set Timeout = 1 minute

3. Click Save



## ✓ Step 5: Add EC2 Permissions to Lambda

1. Go to Permissions tab → Click IAM Role name
2. Click Attach policies
3. Search and attach:
  - AmazonEC2ReadOnlyAccess
  - AmazonEC2FullAccess (*or create custom policy*)

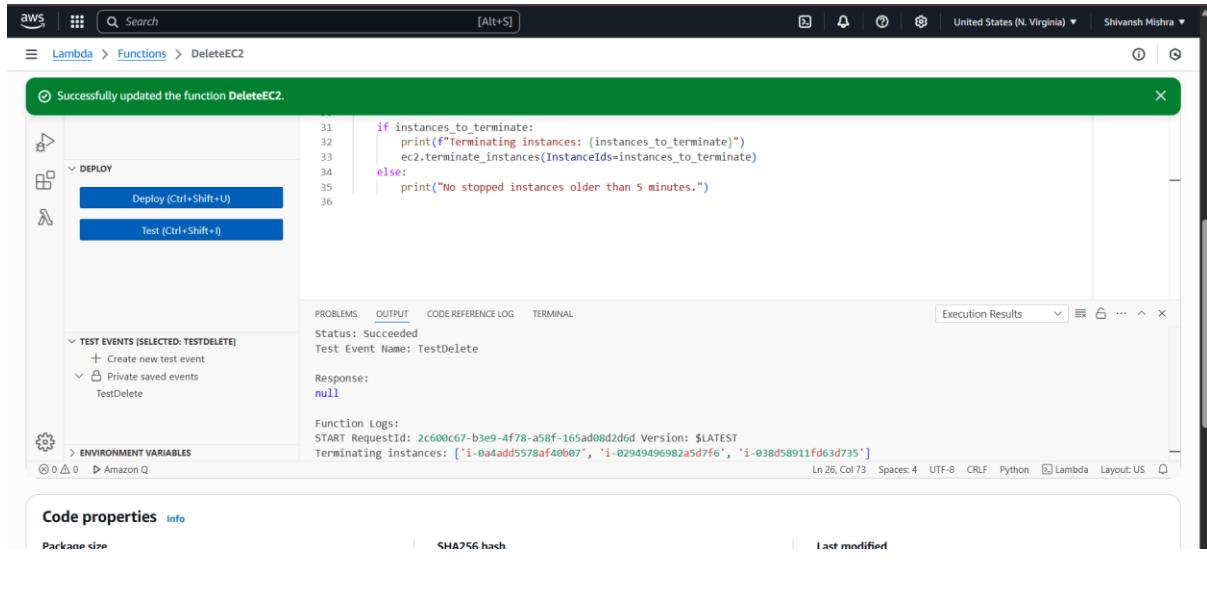


## ✓ Step 6: Manually Test the Lambda Function

1. Go to Code tab → Click Test
2. Configure a new test event (leave JSON empty {})
3. Click Test

You will see:

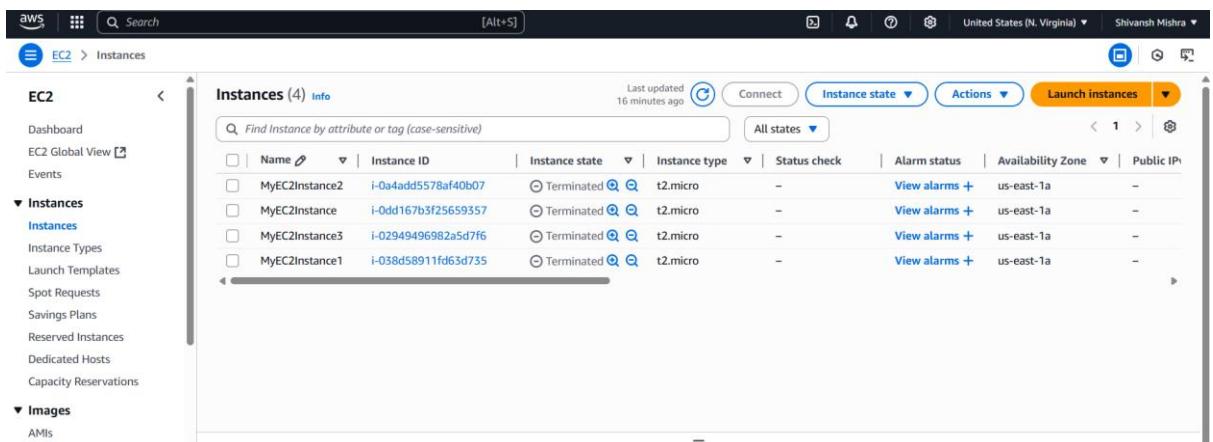
- **Terminated: ['i-xxxx'] if any instance is older than 5 mins**
- **Or “No instances found for termination.”**



The screenshot shows the AWS Lambda console. In the left sidebar, under 'Lambda > Functions > DeleteEC2', there's a success message: 'Successfully updated the function DeleteEC2.' Below it, a 'DEPLOY' section has 'Deploy (Ctrl+Shift+U)' highlighted. Under 'TEST EVENTS [SELECTED: TESTDELETE]', it shows a 'Test Event Name: TestDelete' and a 'Response: null'. The 'FUNCTION LOGS' section displays the Lambda function code and its execution results, which include the termination of three EC2 instances: 'i-0a4add5578af40b07', 'i-02949496982a5d7f6', and 'i-038d58911fd63d735'.

## Output

- **All EC2s that were stopped for more than 5 minutes will be automatically deleted.**
- **Logs show which were terminated.**
- **You save cost and keep your environment clean!**



The screenshot shows the AWS EC2 Instances page. The left sidebar lists 'Instances' and 'Images'. The main area shows a table titled 'Instances (4) Info' with columns: Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IP. All four instances listed ('MyEC2Instance1' through 'MyEC2Instance4') are in the 'Terminated' state. The table includes filters for 'Name' and 'Instance ID', and buttons for 'Connect', 'Actions', and 'Launch instances'.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
MyEC2Instance2	i-0a4add5578af40b07	Terminated	t2.micro	-	<a href="#">View alarms +</a>	us-east-1a	-
MyEC2Instance	i-0dd167b3f25659357	Terminated	t2.micro	-	<a href="#">View alarms +</a>	us-east-1a	-
MyEC2Instance3	i-02949496982a5d7f6	Terminated	t2.micro	-	<a href="#">View alarms +</a>	us-east-1a	-
MyEC2Instance1	i-038d58911fd63d735	Terminated	t2.micro	-	<a href="#">View alarms +</a>	us-east-1a	-