AWS Project

Automated EC2 Lifecycle Management with CloudFormation and Lambda

By: Subhradeep Manna | July 15, 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)**

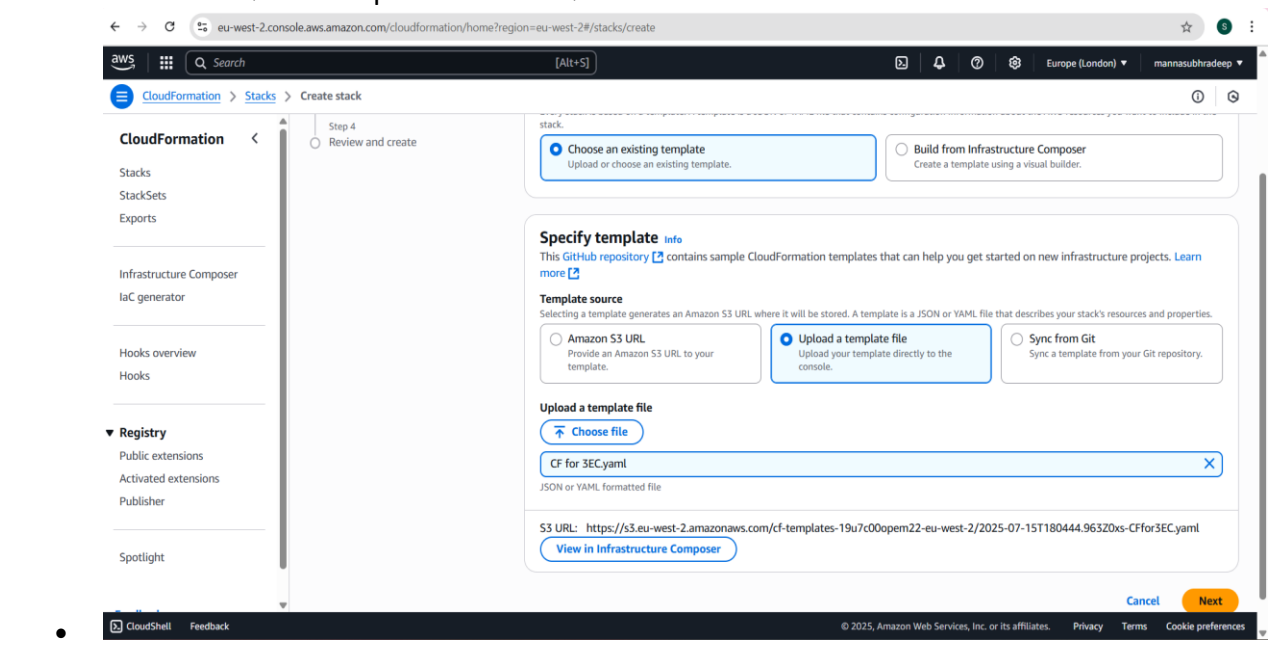
2. Upload or paste this template:

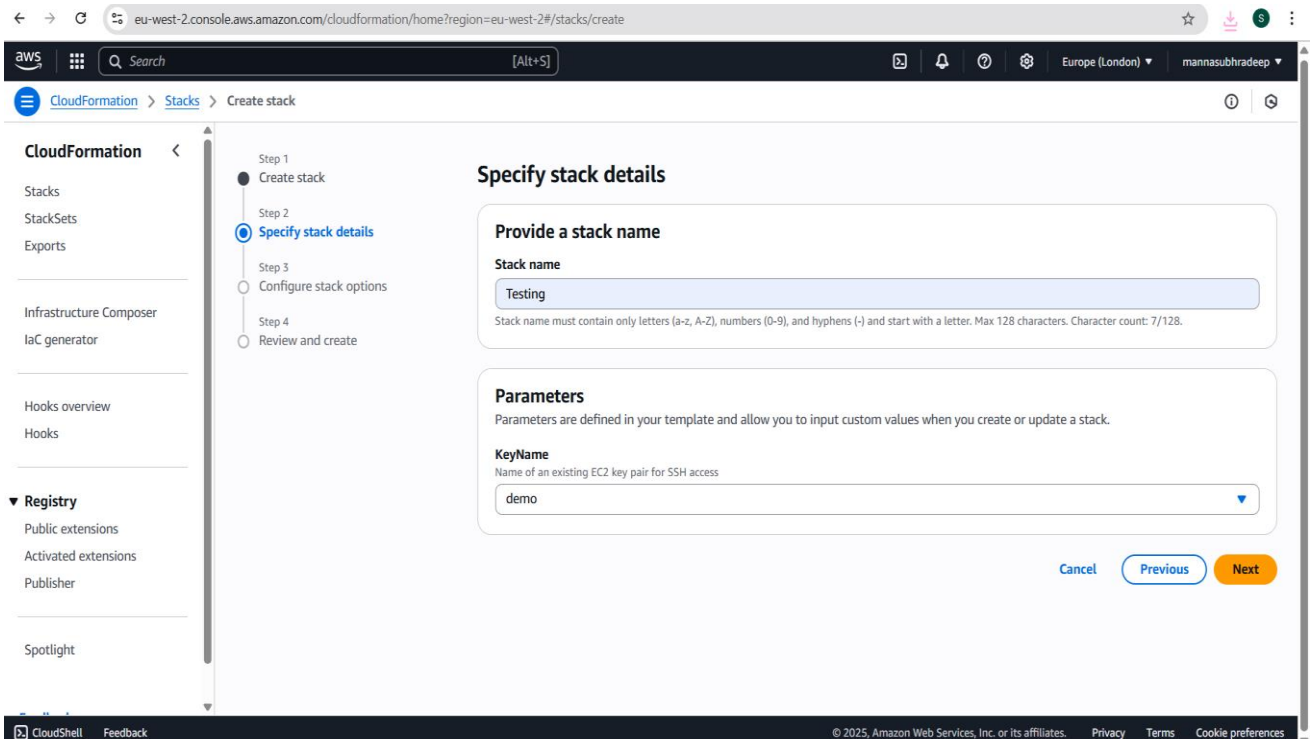
```
AWSTemplateFormatVersion: "2010-09-09"
Description: Launch 3 EC2 instances in an existing VPC and subnet
Parameters:
  KeyName:
    Type: AWS::EC2::KeyPair::KeyName
    Description: Name of an existing EC2 key pair for SSH access
Resources:
   Security group that allows inbound SSH from anywhere
  InstanceSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Enable SSH access
      VpcId: vpc-0c32d77a24c43392b
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: 0.0.0.0/0

   Make sure the subnet is explicitly associated with the given route table
  SubnetRouteAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      SubnetId: subnet-0cc35aea439788393
      RouteTableId: rtb-05e1708fdcbcc5a88
--- EC2 instances
  EC2Instance1:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: ami-0841b1152f02fa85e
      InstanceType: t3.micro
      KeyName: !Ref KeyName
      SubnetId: subnet-0cc35aea439788393
      SecurityGroupIds:
        - !Ref InstanceSecurityGroup
      Tags:
        - Key: Name
          Value: EC2-Instance-1
EC2Instance2:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: ami-0841b1152f02fa85e
      InstanceType: t3.micro
      KeyName: !Ref KeyName
      SubnetId: subnet-0cc35aea439788393
      SecurityGroupIds:
        - !Ref InstanceSecurityGroup
      Tags:
        - Key: Name
          Value: EC2-Instance-2
EC2Instance3:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: ami-0841b1152f02fa85e
      InstanceType: t3.micro
      KeyName: !Ref KeyName
      SubnetId: subnet-0cc35aea439788393
      SecurityGroupIds:
        - !Ref InstanceSecurityGroup
      Tags:
        - Key: Name
          Value: EC2-Instance-3
Outputs:
  InstanceIDs:
    Description: IDs of all EC2 instances
    Value: !Join [ ",", [ !Ref EC2Instance1, !Ref EC2Instance2, !Ref EC2Instance3 ] ]
```

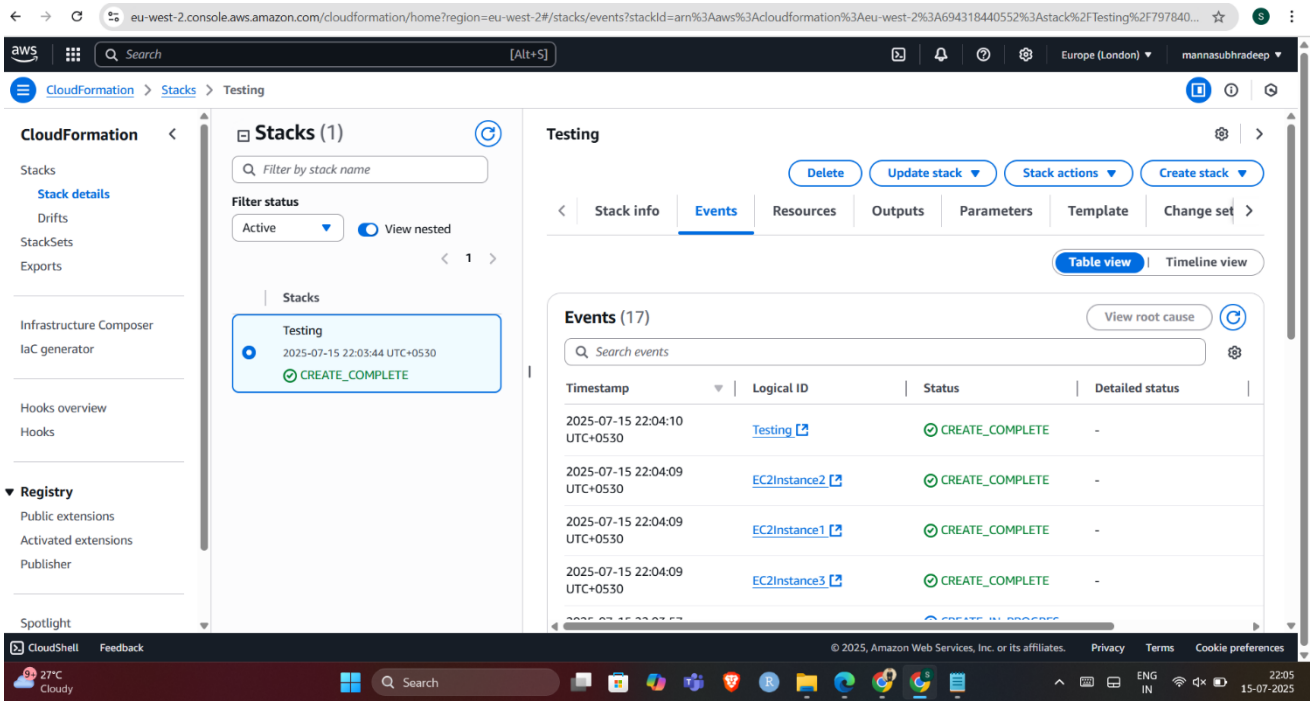
> Replace subnet ID, VPC ID, Gateway ID with a valid ones in your VPC.

3. Create the stack

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



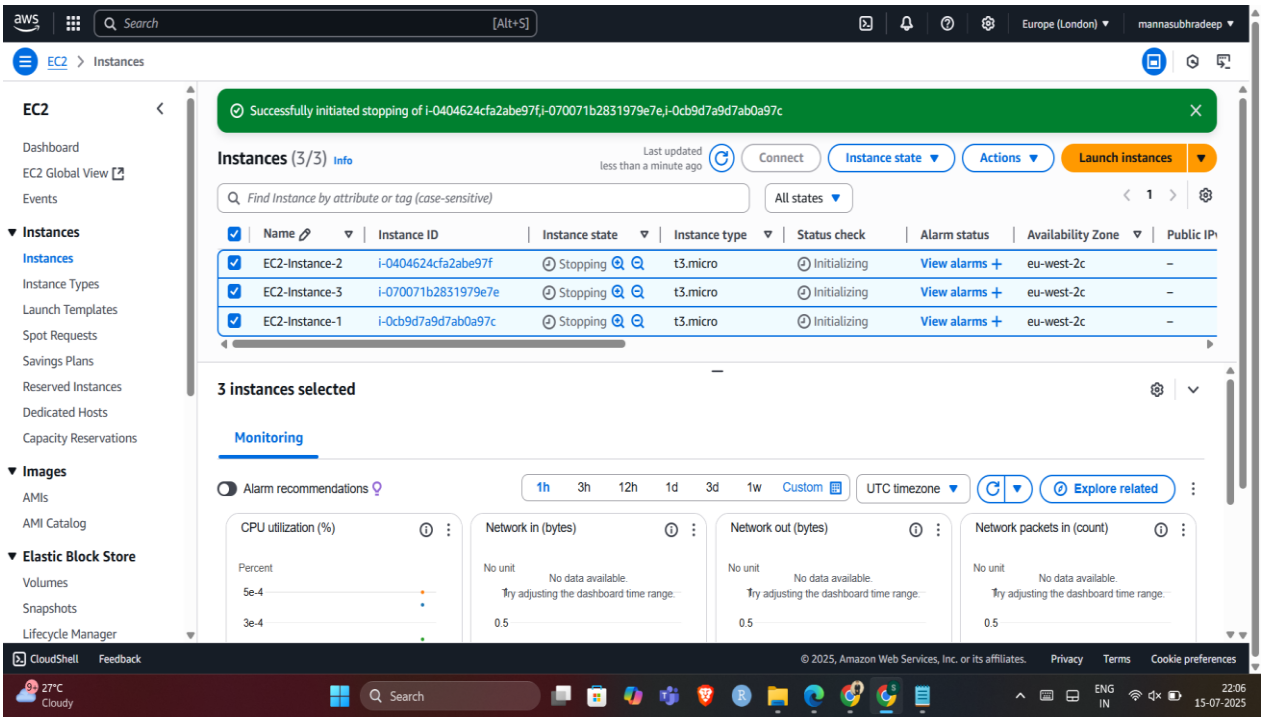




Step 2: Stop the EC2 Instances

After the stack is created:

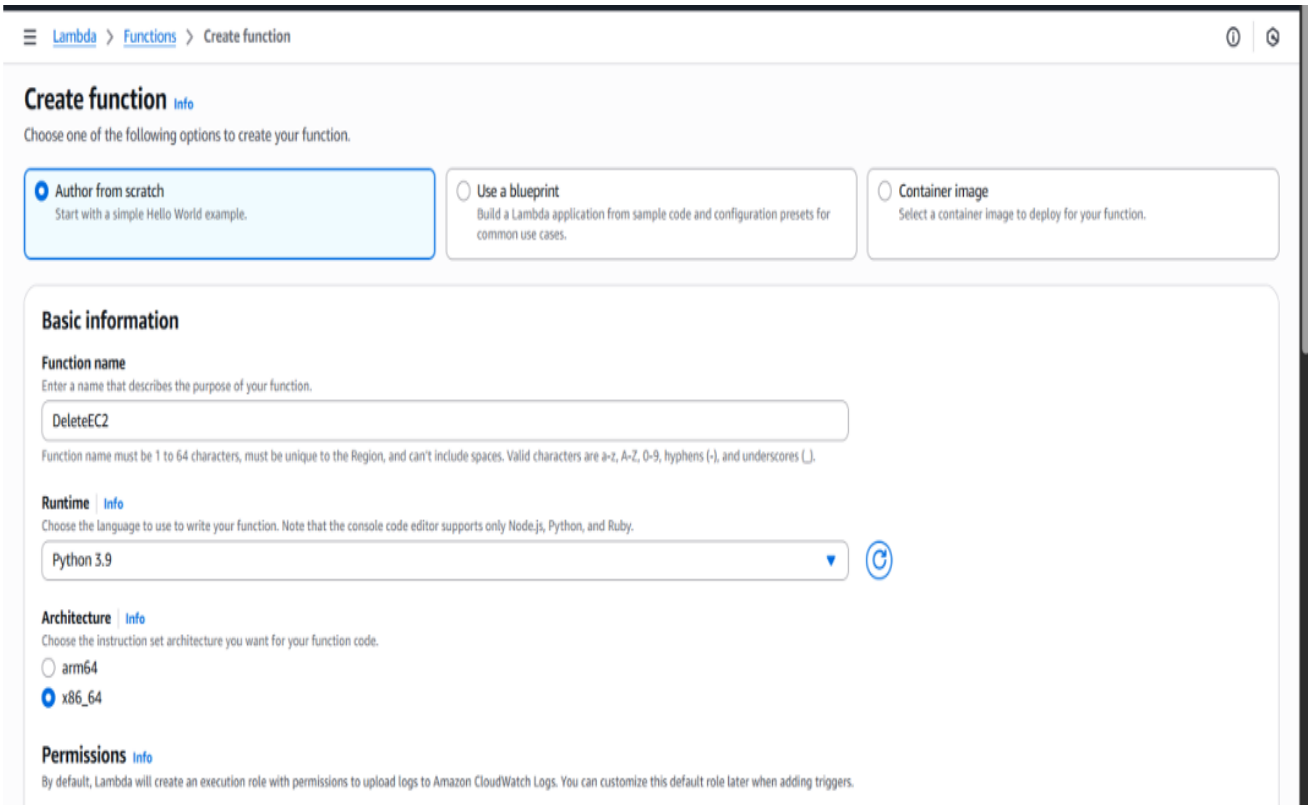
1. Go to **EC2 Console**.
2. Select all 3 instances → Click **"Instance State"** → **Stop instance**.
3. Wait for them to fully enter the **"stopped"** state.



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 permission**



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

import boto3

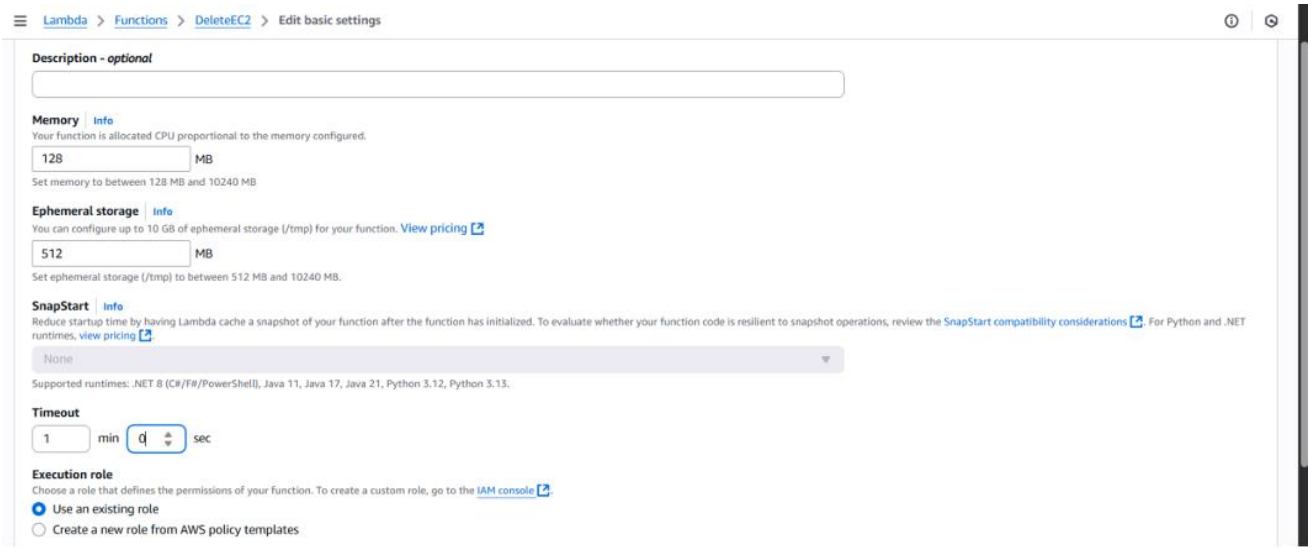
File Edit View

```
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

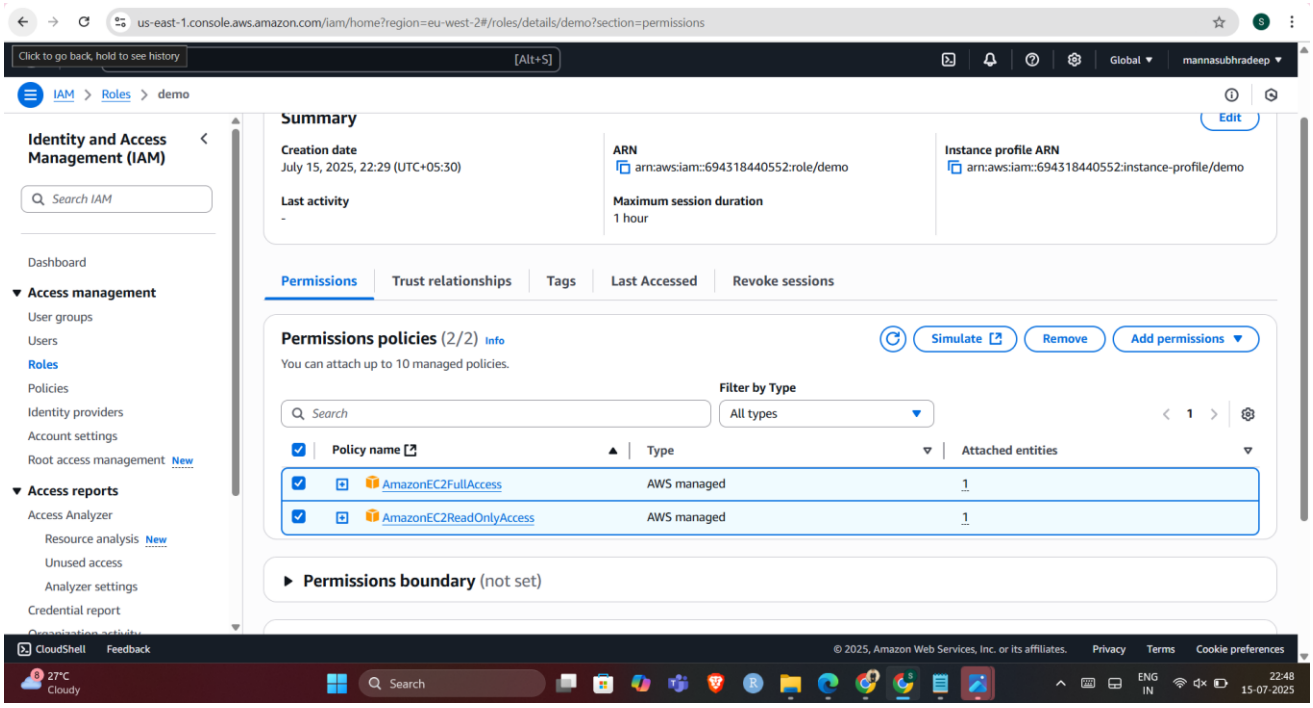
Step 4: Increase Timeout

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)***



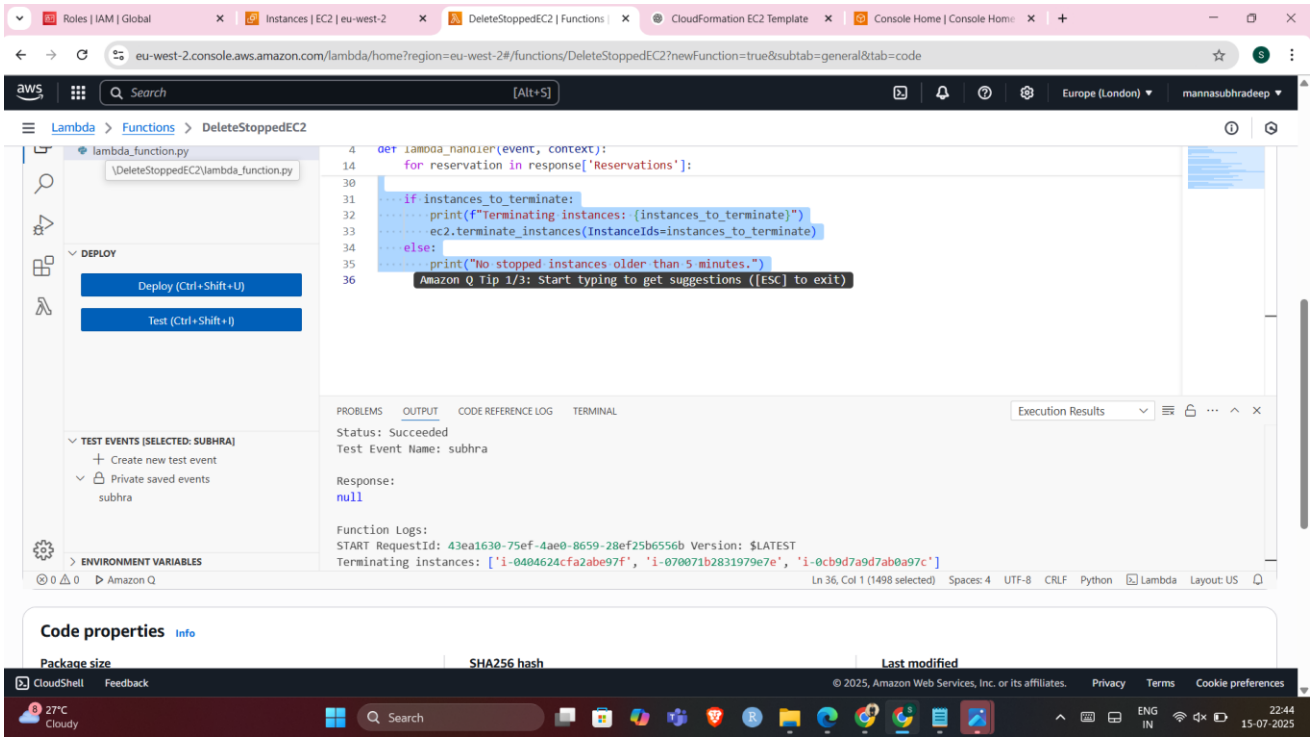
Step 6: Manually Test the Lambda Function

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.”**



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!**