Automation with AWS Lambda, Python and Boto3

AWS Lambda - the main AWS service, if we think of serverless architecture. A piece of code triggered by events written in one of six programming languages on servers whose administration we do not have to worry about.

Python - high level programming language, it is characterized by transparency and readability, as well as a large number of available libraries and documentation.

Boto3 - is the official AWS library for Python that allows you to create, monitor and manage AWS services.

AWS Lambda — HOW to create VPC, EC2 instance, start and stop the EC2 instance.

1. Select Functions > Create Function
2. Enter the name of your lambda function.
3. Select python as a runtime language.
4. Click on **Choose or create an execution role** > Select Use an existing role.
5. Click Create Function.
6. Navigate to AWS Lambda
7. Select `Functions > Create Function`
8. Click on **Choose or create an execution role** > Select `Use an existing role`.
9. Click on Create Function
10. Add the following code in created lambda function.

To create VPC

```
import boto3
ec2 = boto3.resource('ec2', region_name='us-east-1')
vpc = ec2.create_vpc(CidrBlock='100.0.0.0/16')
# Assign a name to the VPC
vpc.create_tags(Tags=[{"Key": "Name", "Value": "vpc200"}])
vpc.wait_until_available()
print(vpc.id)
# Create and Attach the Internet Gateway
ig = ec2.create_internet_gateway()
vpc.attach_internet_gateway(InternetGatewayId=ig.id)
print(ig.id)
# Create a route table and a public route to Internet Gateway
route_table = vpc.create_route_table()
route = route_table.create_route(
   DestinationCidrBlock='0.0.0.0/0',
   GatewayId=ig.id
)
print(route_table.id)

# Create a Subnet
subnet = ec2.create_subnet(CidrBlock='100.0.1.0/24', VpcId=vpc.id)
print(subnet.id)

# associate the route table with the subnet
route_table.associate_with_subnet(SubnetId=subnet.id)

def lambda_handler(event, context):
   init_script = """#!/bin/bash
            yum update -y"""
```

To create EC2 instance

```
import os
import boto3
AMI = 'ami-00874d747dde814fa'
INSTANCE_TYPE = 't2.micro'
KEY_NAME = 'sonicmaster'
SUBNET_ID = 'subnet-05226b93c5fa50219'
REGION = 'us-east-1'
ec2 = boto3.client('ec2', region_name=REGION)
def lambda_handler(event, context):
   init_script = """#!/bin/bash
            yum update -y
            yum install -y httpd24
            service httpd start
            chkconfig httpd on
            echo > /var/www/html/index.html
            shutdown -h +5"""

    instance = ec2.run_instances(
      ImageId=AMI,
      InstanceType=INSTANCE_TYPE,
      KeyName=KEY_NAME,
      SubnetId=SUBNET_ID,
      MaxCount=1,
      MinCount=1,
      InstanceInitiatedShutdownBehavior='terminate',
      UserData=init_script
  )  instance_id = instance['Instances'][0]['InstanceId']
     print(instance_id)
     return instance_id
```

To stop the ec2 instance

```
#Stop the instances:-
import boto3
region = 'us-east-1'
instances = ['i-07c0d040cca617d9c']
ec2 = boto3.client('ec2', region_name=region)

def lambda_handler(event, context):
    ec2.stop_instances(InstanceIds=instances)
    print('stoped your instances: ' + str(instances))
```

To start the ec2 instance

```
#Start the instances:-
import boto3
region = 'us-east-1'
instances = ['i-07c0d040cca617d9c']
ec2 = boto3.client('ec2', region_name=region)

def lambda_handler(event, context):
    ec2.start_instances(InstanceIds=instances)
    print('started your instances: ' + str(instances))
```