Project Documentation

Stage A : create infraStructure using CloudFormation “project9.yml”

- **Code**

```
---
AWSTemplateFormatVersion: '2010-09-09'
Description: VPC with EC2 instances, auto scaling, ELB, CloudWatch agent, SSM
agent, and NGINX installation

Parameters:
  Region:
      Type: String
      Default: "eu-west-1" #irland
      Description: "region :irland"

  VpcCIDR:
    Type: String
    Description: CIDR block for the VPC
    Default: 10.0.0.0/16
  Subnet1CIDR:
    Type: String
    Description: CIDR block for subnet 1
    Default: 10.0.0.0/24
  Subnet2CIDR:
    Type: String
    Description: CIDR block for subnet 2
    Default: 10.0.1.0/24

Resources:
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCIDR
      EnableDnsSupport: true
      EnableDnsHostnames: true

  Subnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Ref Subnet1CIDR
      MapPublicIpOnLaunch: true
      AvailabilityZone: eu-west-1a

  Subnet2:
```

```
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Ref Subnet2CIDR
      MapPublicIpOnLaunch: true
      AvailabilityZone: eu-west-1b

InternetGateway:
    Type: AWS::EC2::InternetGateway

AttachGateway:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      VpcId: !Ref VPC
      InternetGatewayId: !Ref InternetGateway

RouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC

PublicRoute:
    Type: AWS::EC2::Route
    DependsOn: AttachGateway
    Properties:
      RouteTableId: !Ref RouteTable
      DestinationCidrBlock: "0.0.0.0/0"
      GatewayId: !Ref InternetGateway

Subnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      SubnetId: !Ref Subnet1
      RouteTableId: !Ref RouteTable

Subnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      SubnetId: !Ref Subnet2
      RouteTableId: !Ref RouteTable

SecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Allow all traffic on port 80
      VpcId: !Ref VPC
      SecurityGroupIngress:
```

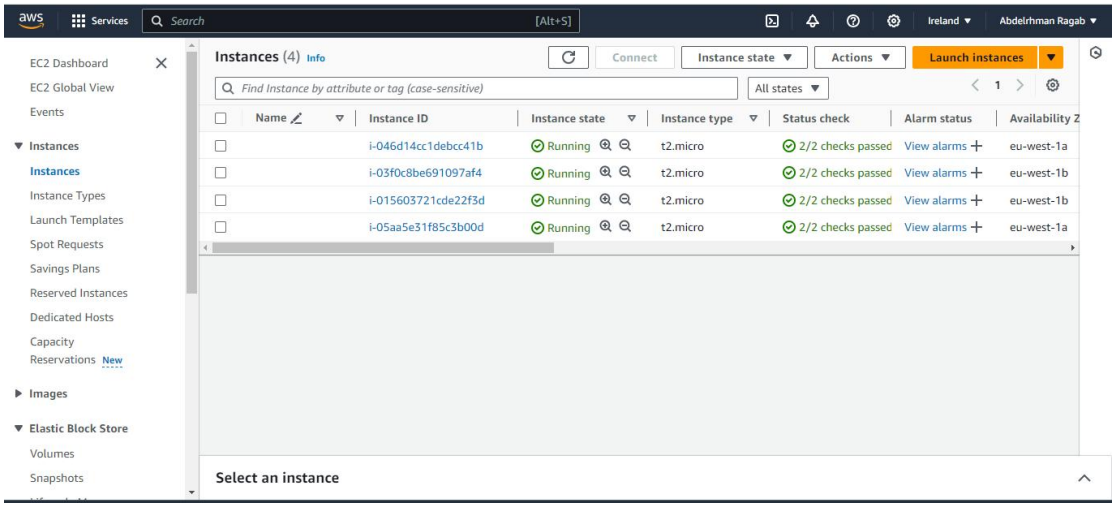
```
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0

  TargetGroup:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    Properties:
      TargetType: instance
      Protocol: HTTP
      Port: 80
      VpcId: !Ref VPC
      HealthCheckPath: /
      HealthCheckProtocol: HTTP
      HealthCheckIntervalSeconds: 30
      HealthCheckTimeoutSeconds: 10
      HealthyThresholdCount: 5
      UnhealthyThresholdCount: 2

  LoadBalancer:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Name: MyLoadBalancer
      Subnets:
        - !Ref Subnet1
        - !Ref Subnet2
      SecurityGroups:
        - !Ref SecurityGroup
      Scheme: internet-facing
      Type: application

  LoadBalancerListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    Properties:
      DefaultActions:
        - TargetGroupArn: !Ref TargetGroup
          Type: forward
      LoadBalancerArn: !Ref LoadBalancer
      Port: 80
      Protocol: HTTP

  LaunchConfiguration:
    Type: AWS::AutoScaling::LaunchConfiguration
    Properties:
      ImageId: ami-0c1c30571d2dae5c9
      InstanceType: t2.micro
      KeyName: key1
```

```
      SecurityGroups:
        - !Ref SecurityGroup
      UserData:
        Fn::Base64: |
          #!/bin/bash
          wget
https://s3.amazonaws.com/amazoncloudwatch-agent/linux/amd64/latest/Amazo
nCloudWatchAgent.zip
          sudo apt install unzip
          sudo ./install.sh
          sudo systemctl start amazon-cloudwatch-agent
          sudo systemctl enable amazon-cloudwatch-agent

          wget
https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/debian_a
md64/amazon-ssm-agent.deb
          sudo snap refresh amazon-ssm-agent
          sudo snap remove amazon-ssm-agent
          sudo dpkg -i amazon-ssm-agent.deb
          sudo systemctl start amazon-ssm-agent
          sudo systemctl enable amazon-ssm-agent

      IamInstanceProfile: !Ref InstanceProfile

  InstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Roles:
        - !Ref EC2Role

  EC2Role:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - ec2.amazonaws.com
            Action: sts:AssumeRole
      Policies:
        - PolicyName: EC2Policy
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              - Effect: Allow
                Action:
```

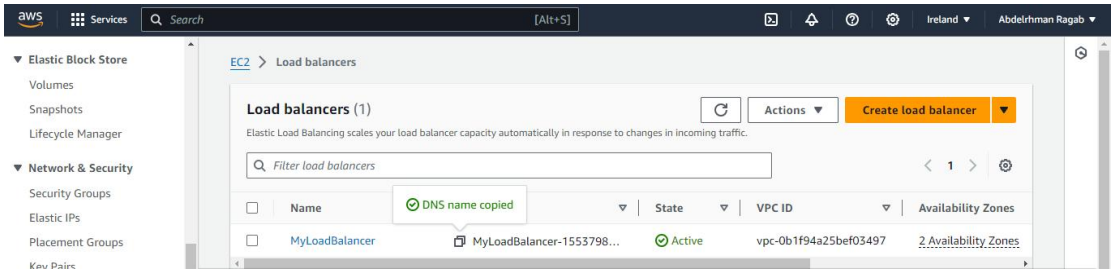
```
                - "ssm:DescribeAssociation"
                - "ssm:GetDeployablePatchSnapshotForInstance"
                - "ssm:GetDocument"
                - "ssm:DescribeDocument"
                - "ssm:GetManifest"
                - "ssm:GetParameter"
                - "ssm:GetParameters"
                - "ssm:ListAssociations"
                - "ssm:ListInstanceAssociations"
                - "ssm:PutInventory"
                - "ssm:PutComplianceItems"
                - "ssm:PutConfigurePackageResult"
                - "ssm:UpdateAssociationStatus"
                - "ssm:UpdateInstanceAssociationStatus"
                - "ssm:UpdateInstanceInformation"
                - "ssmmessages:CreateControlChannel"
                - "ssmmessages:CreateDataChannel"
                - "ssmmessages:OpenControlChannel"
                - "ssmmessages:OpenDataChannel"
              Resource: '*'
            - Effect: Allow
              Action:
                - "cloudwatch:PutMetricData"
                - "cloudwatch:PutMetricAlarm"
                - "cloudwatch:DescribeAlarms"
                - "cloudwatch:GetMetricStatistics"
                - "logs:CreateLogGroup"
                - "logs:CreateLogStream"
                - "logs:PutLogEvents"
                - "logs:DescribeLogStreams"
              Resource: "*"

AutoScalingGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    DesiredCapacity: '4'
    LaunchConfigurationName: !Ref LaunchConfiguration
    MaxSize: '7'
    MinSize: '4'
    VPCZoneIdentifier:
    - !Ref Subnet1    # First subnet
    - !Ref Subnet2    # Second subnet
```

- **Output**

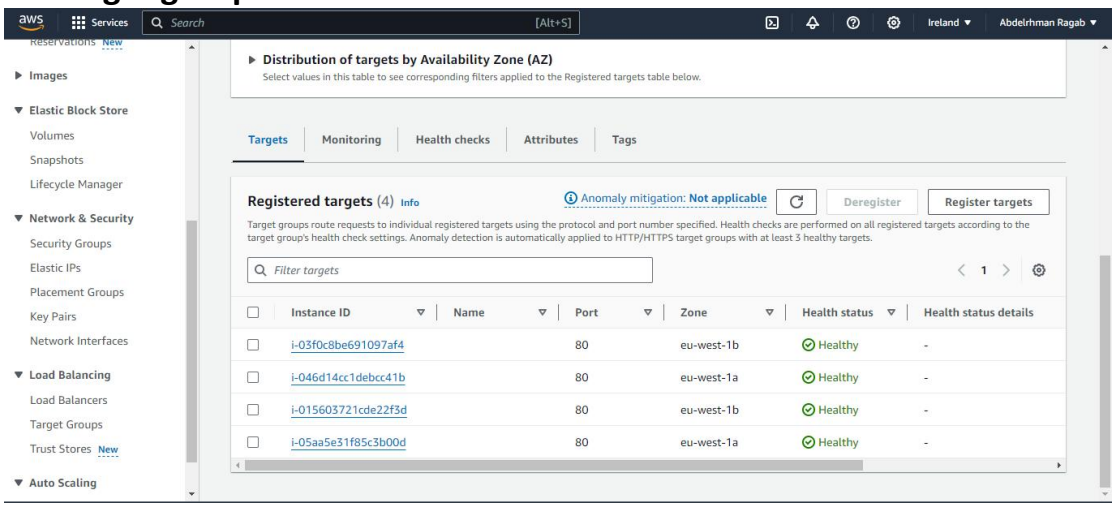
1- 2 Ec2 in each subnet



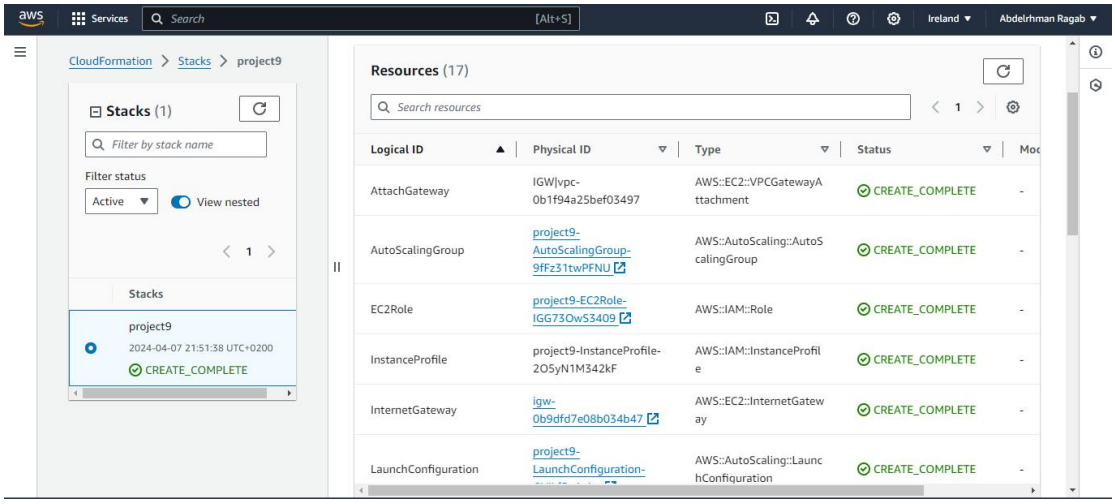
2- Load blancer



3- Target group

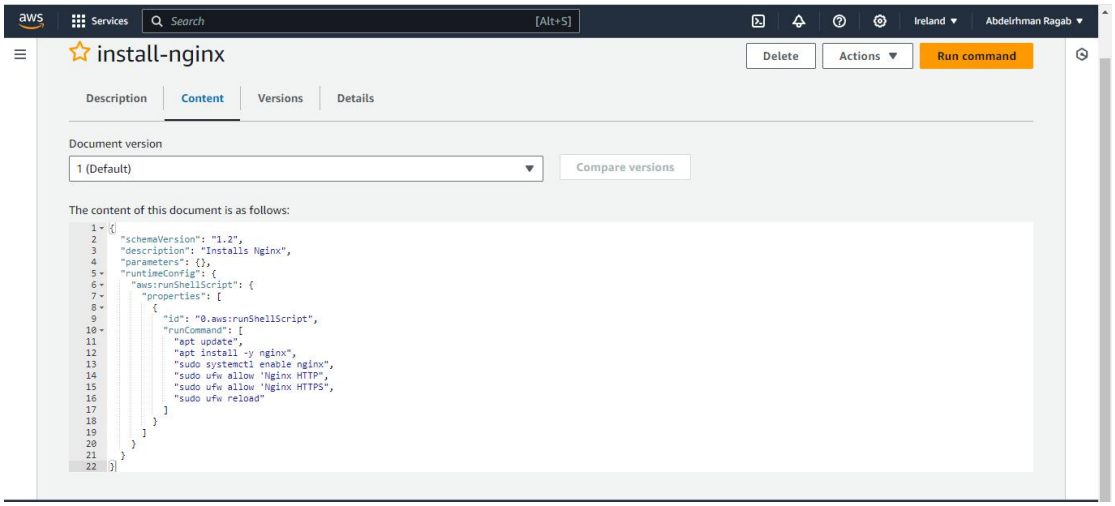


4- Stack

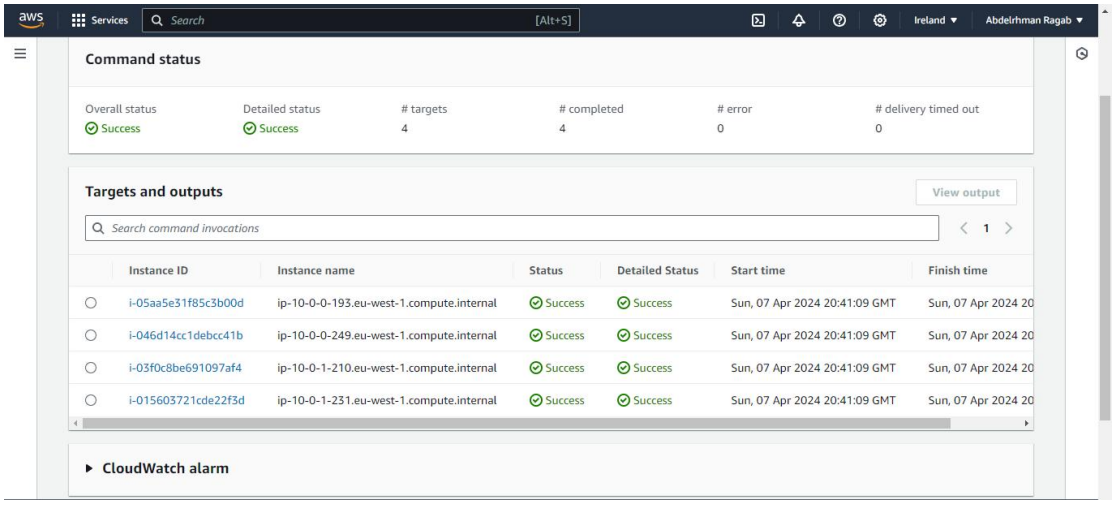


Stage B : install nginx using ssm

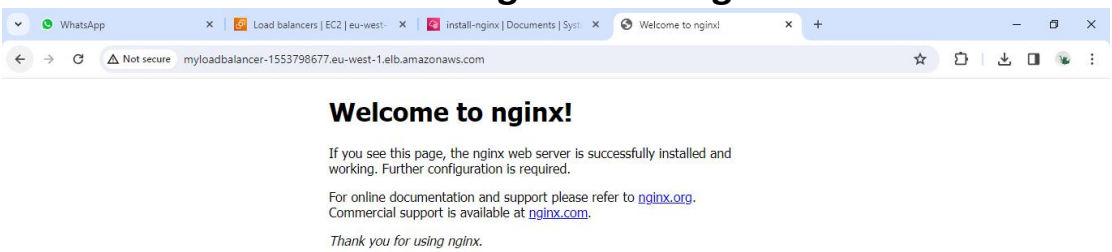
1- Installation code



2- Fleet Manager

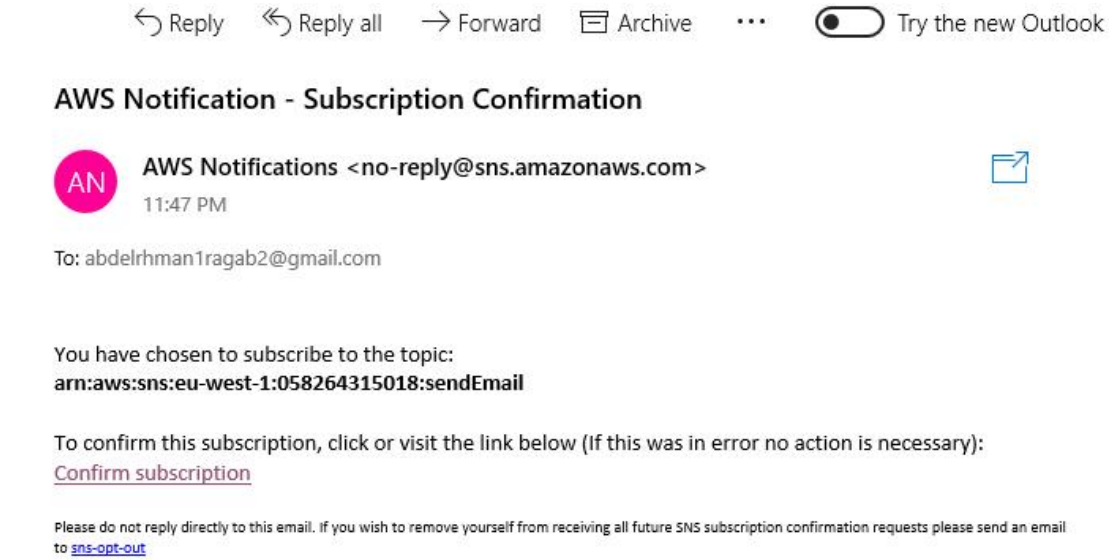


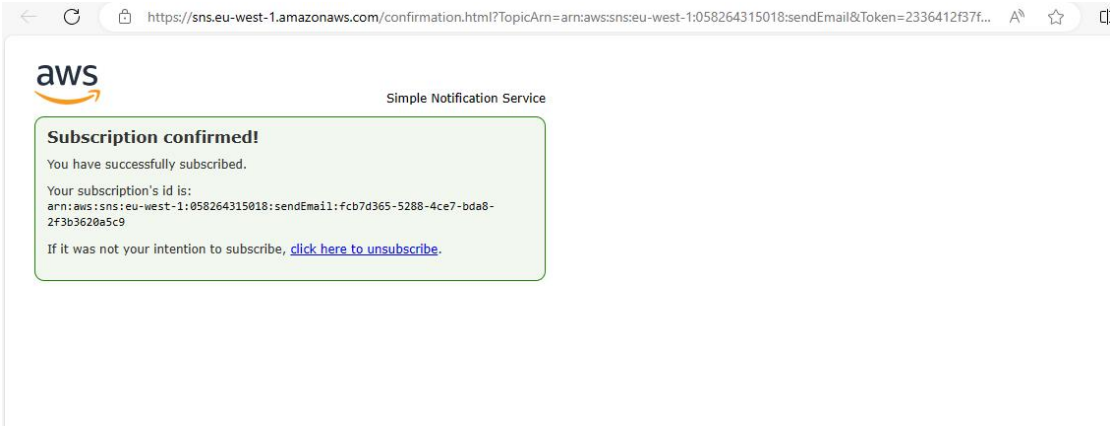
3- Access loadbalancer using DNS and nginx installed



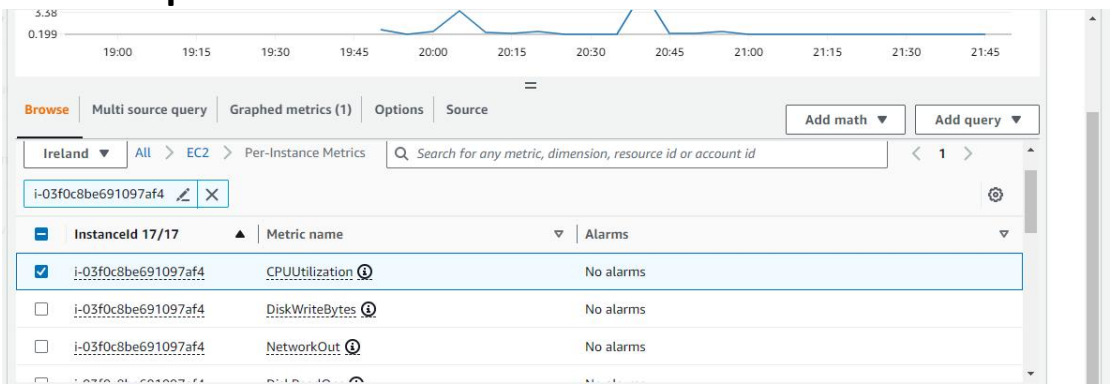
Stage C : cloudwatch send mail

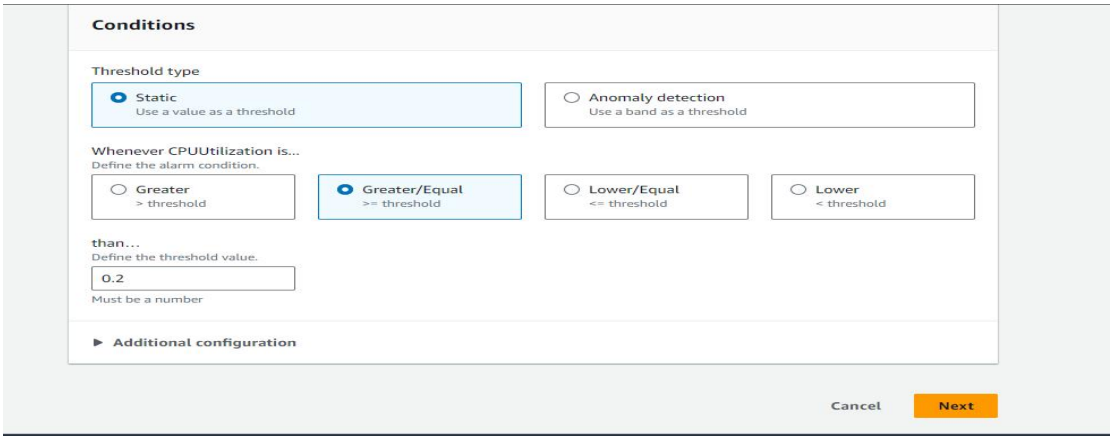
1- Create sns and activiate it





2- Create Alarm and select metrics and thrshould
For example CPU-Utilization for EC2





3- Output Email