

Practical case 6

Student: Yago Toro Molina (ytoro-mo).

E-Mail: ytoro-mo@student.42malaga.com

Task 1: Create a CloudFormation stack that deploys all needed resources.

Stack Template:

```
#####
# Parameters section
#####

Parameters:

EnvironmentName:
  Description: VPC Name
  Type: String
  Default: Case6VPC

VpcCIDR:
  Description: IP range (CIDR notation) for this VPC
  Type: String
  Default: 42.0.0.0/16

PublicSubnet1CIDR:
  Description: IP range (CIDR notation) for the public subnet in the first Availability Zone
  Type: String
  Default: 42.0.1.0/24

PublicSubnet2CIDR:
  Description: IP range (CIDR notation) for the public subnet in the second Availability Zone
  Type: String
  Default: 42.0.2.0/24

AmazonLinuxAMIID:
  Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
  Default: /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2

LaunchTemplateVersionNumber:
  Default: 1
  Type: String

#####
# Resources section
#####

Resources:

#####
# VPC
#####

VPC:
  Type: AWS::EC2::VPC
  Properties:
    CidrBlock: !Ref VpcCIDR
    EnableDnsSupport: true
    EnableDnsHostnames: true
    Tags:
      - Key: Name
        Value: !Ref EnvironmentName

InternetGateway:
  Type: AWS::EC2::InternetGateway
  Properties:
    Tags:
      - Key: Name
        Value: !Ref EnvironmentName

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

PublicSubnet1:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref VPC
    AvailabilityZone: !Select [ 0, !GetAZs '' ]
    CidrBlock: !Ref PublicSubnet1CIDR
    MapPublicIpOnLaunch: true
    Tags:
      - Key: Name
        Value: !Sub ${EnvironmentName} Public Subnet (AZ1)

PublicSubnet2:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref VPC
    AvailabilityZone: !Select [ 1, !GetAZs '' ]
    CidrBlock: !Ref PublicSubnet2CIDR
    MapPublicIpOnLaunch: true
    Tags:
      - Key: Name
        Value: !Sub ${EnvironmentName} Public Subnet (AZ2)

PublicRouteTable:
  Type: AWS::EC2::RouteTable
  Properties:
    VpcId: !Ref VPC
    Tags:
      - Key: Name
        Value: !Sub ${EnvironmentName} Public Routes

DefaultPublicRoute:
  Type: AWS::EC2::Route
  DependsOn: InternetGatewayAttachment
  Properties:
    RouteTableId: !Ref PublicRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId: !Ref InternetGateway

PublicSubnet1RouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    RouteTableId: !Ref PublicRouteTable
    SubnetId: !Ref PublicSubnet1

PublicSubnet2RouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    RouteTableId: !Ref PublicRouteTable
    SubnetId: !Ref PublicSubnet2
```

```
#####
# Auto Scaling
#####

LaunchTemplate:
  Type: AWS::EC2::LaunchTemplate
  Properties:
    LaunchTemplateData:
      ImageId: !Ref AmazonLinuxAMIID
      InstanceType: t3.micro
      NetworkInterfaces:
        - DeviceIndex: 0
          AssociatePublicIpAddress: true
          Groups:
            - !Ref WebServerSecurityGroup
          DeleteOnTermination: true
      Placement:
        Tenancy: default
      UserData:
        Fn::Base64:
          !Sub |
            #!/bin/bash
            sudo yum update -y
            sudo amazon-linux-extras install nginx1 -y
            sudo systemctl enable nginx
            sudo systemctl start nginx

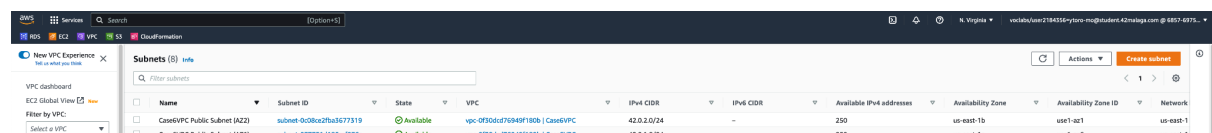
    LaunchTemplateName: !Sub '${AWS::StackName}-autoscaling-launch-template'

WebServerSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Enable HTTP ingress
    VpcId: !Ref VPC
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 80
        ToPort: 80
        CidrIp: 0.0.0.0/0
    Tags:
      - Key: Name
        Value: Web Server Security Group

AutoScalingGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    MinSize: '2'
    MaxSize: '4'
    DesiredCapacity: '2'
    LaunchTemplate:
      LaunchTemplateId: !Ref LaunchTemplate
      Version: !Ref LaunchTemplateVersionNumber
    VPCZoneIdentifier:
      - !Ref PublicSubnet1
      - !Ref PublicSubnet2
```

Deployment:

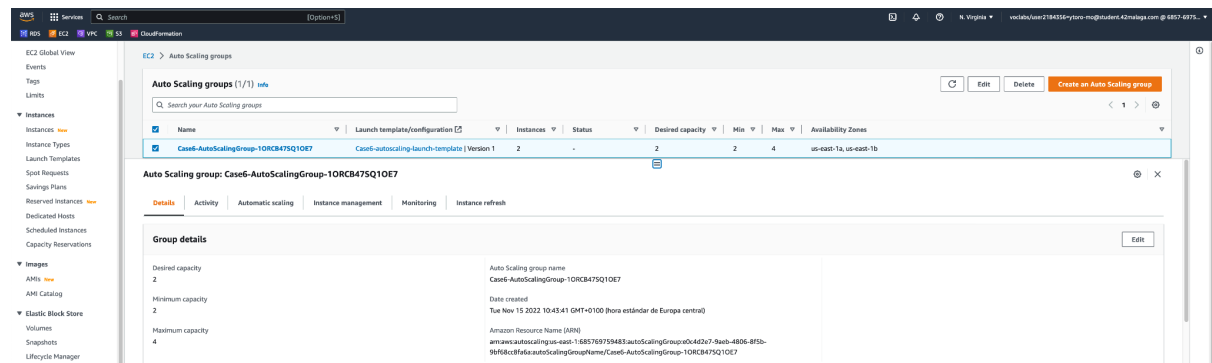
- Subnets:



The screenshot shows the AWS Management Console 'Subnets' page. A table lists two subnets: 'Case6VPC Public Subnet (AZ2)' and 'Case6VPC Public Subnet (AZ1)'. Both are in the 'Available' state, associated with VPC 'vpc-0f306d7f5949f180b', and have an IPv4 CIDR of '42.0.2.0/24'. They are located in 'us-east-1b' and 'us-east-1a' availability zones respectively.

Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR	Available IPv4 addresses	Availability Zone	Availability Zone ID	Network
Case6VPC Public Subnet (AZ2)	subnet-0c08x2fba3677319	Available	vpc-0f306d7f5949f180b Case6VPC	42.0.2.0/24	-	250	us-east-1b	us-east-1b	us-east-1
Case6VPC Public Subnet (AZ1)	subnet-077731d185cc0765a	Available	vpc-0f306d7f5949f180b Case6VPC	42.0.1.0/24	-	250	us-east-1a	us-east-1a	us-east-1

- Auto Scaling Group:



The screenshot shows the AWS Management Console 'Auto Scaling groups' page. A table lists one group: 'Case6-AutoScalingGroup-10RCB475Q10E7'. It is in the 'Active' state, using launch template 'Case6-autoscaling-launch-template' (Version 1), with 2 instances, a desired capacity of 2, and a minimum of 2 and maximum of 4 instances. It is located in 'us-east-1a, us-east-1b' availability zones.

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones
Case6-AutoScalingGroup-10RCB475Q10E7	Case6-autoscaling-launch-template Version 1	2	Active	2	2	4	us-east-1a, us-east-1b

Group details	
Desired capacity	Auto Scaling group name
2	Case6-AutoScalingGroup-10RCB475Q10E7
Minimum capacity	Date created
2	Tue Nov 15 2022 10:43:41 GMT+0100 (hora estándar de Europa central)
Maximum capacity	Amazon Resource Name (ARN)
4	arn:aws:autoscaling:us-east-1:685769759483:autoScalingGroup:auto42d27-3aeb-4806-8f5b-99f0b0c8fde:autoScalingGroup/Case6-AutoScalingGroup-10RCB475Q10E7

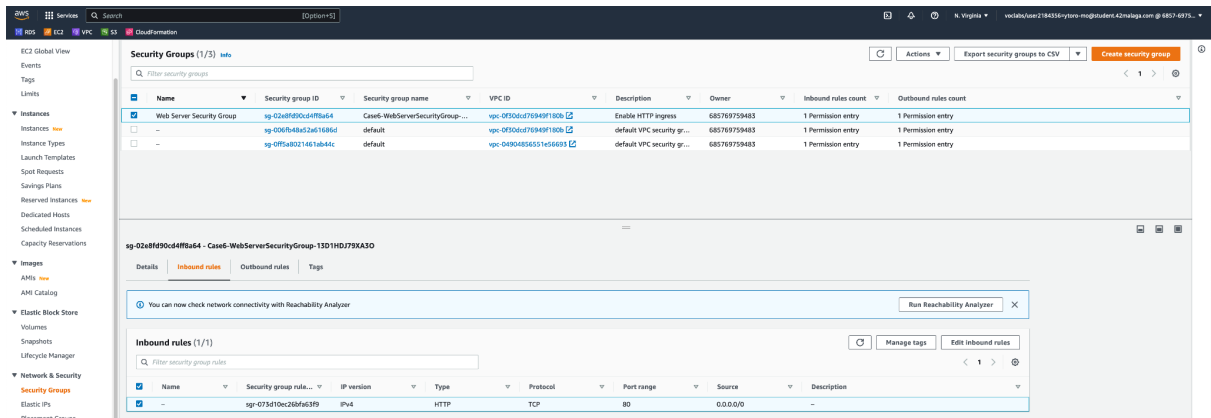
- EC2 instances deployed:



The screenshot shows the AWS Management Console 'Instances' page. Two EC2 instances are listed, both in a 'Running' state. The first instance is 'i-0891186c290569b7' and the second is 'i-098c28c209922084'. Both are t3.micro instances with a public IP address and an Elastic IP.

Name	Instance ID	Instance state	Instance type	Status check	Ala...	Availab...	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs	Monitoring	Security group name	Key name
-	i-0891186c290569b7	Running	t3.micro	Initializing	No	us-east-1a	ec2-52-90-40-45.comp...	52.90.40.45	-	-	disabled	Case6-WebServerSecurityGroup-13D1HDJ79XA3O	-
-	i-098c28c209922084	Running	t3.micro	Initializing	No	us-east-1b	ec2-44-202-77-17.com...	44.202.77.17	-	-	disabled	Case6-WebServerSecurityGroup-13D1HDJ79XA3O	-

- Security Group:



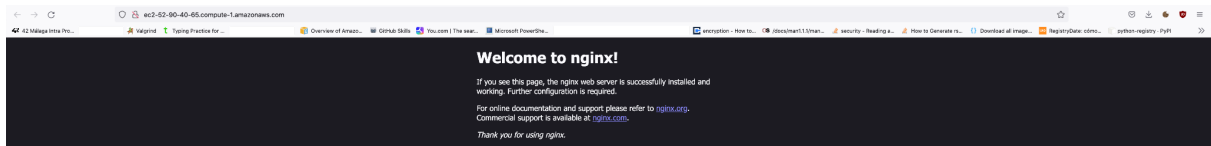
The screenshot shows the AWS Management Console 'Security Groups' page. Three security groups are listed. The first is 'sg-02a8f690c4ff8a64' (Case6-WebServerSecurityGroup-13D1HDJ79XA3O) with one inbound rule. The second is 'sg-006fa4ba33af1686d' (default VPC security gr...) with one inbound rule. The third is 'sg-0f5a8021467db4fc' (default VPC security gr...) with one inbound rule.

Name	Security group ID	Security group name	VPC ID	Description	Owner	Inbound rules count	Outbound rules count
Web Server Security Group	sg-02a8f690c4ff8a64	Case6-WebServerSecurityGroup-13D1HDJ79XA3O	vpc-0f50a07f949f180a	Enable HTTP ingress	680769793483	1 Permission entry	1 Permission entry
-	sg-006fa4ba33af1686d	default VPC security gr...	vpc-0f50a07f949f180a	default VPC security gr...	680769793483	1 Permission entry	1 Permission entry
-	sg-0f5a8021467db4fc	default VPC security gr...	vpc-0f50a07f949f180a	default VPC security gr...	680769793483	1 Permission entry	1 Permission entry

The 'Inbound rules' section for the 'Web Server Security Group' shows one rule:

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
-	sg-073d10ec26f653f9	IPv4	HTTP	TCP	80	0.0.0.0/0	-

- HTTP connection to nginx:



Task 2: Create two CloudFormation stack, one to deploy all network resources and another to deploy all computing resources.

Stack Template:

- Network Resources Stack:

```
#####
# Parameters section
#####

Parameters:

  EnvironmentName:
    Description: VPC Name
    Type: String
    Default: caseVPC

  VpcCIDR:
    Description: IP range (CIDR notation) for this VPC
    Type: String
    Default: 42.0.0.0/24

  PublicSubnet1CIDR:
    Description: IP range (CIDR notation) for the public subnet in the first Availability Zone
    Type: String
    Default: 42.0.1.0/24

  PublicSubnet2CIDR:
    Description: IP range (CIDR notation) for the public subnet in the second Availability Zone
    Type: String
    Default: 42.0.2.0/24

#####
# Resources section
#####

Resources:

  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCIDR
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Ref EnvironmentName

  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Ref EnvironmentName

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

  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 0, !GetAZs '' ]
      CidrBlock: !Ref PublicSubnet1CIDR
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Subnet (AZ1)

  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 1, !GetAZs '' ]
      CidrBlock: !Ref PublicSubnet2CIDR
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Subnet (AZ2)

  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Routes

  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway

  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1

  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2

Outputs:

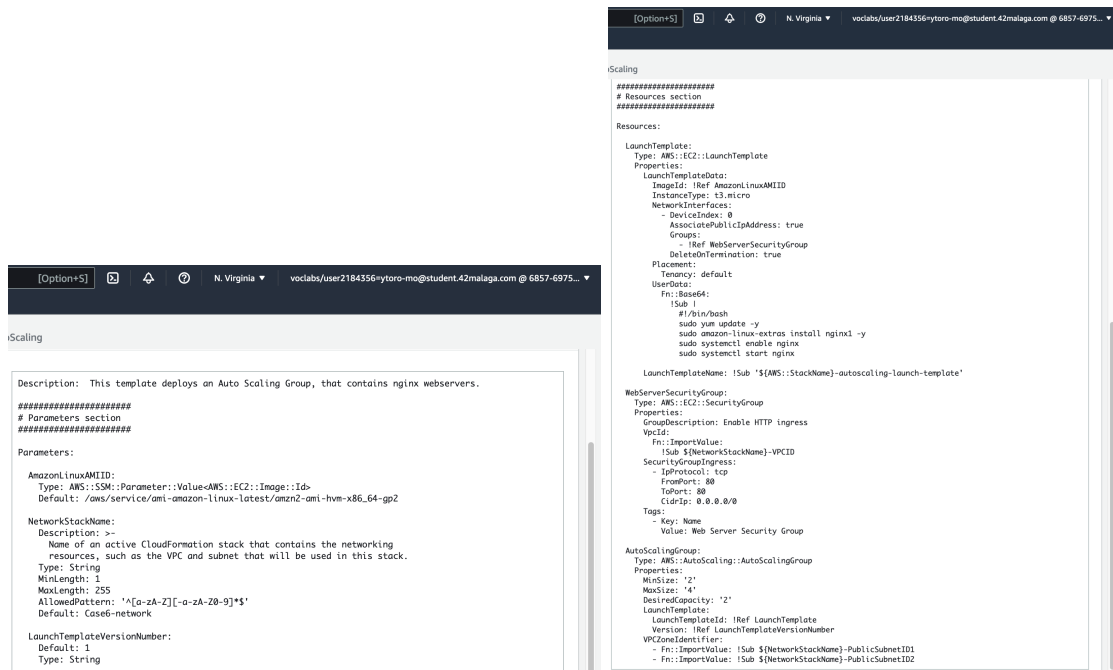
  VPC:
    Description: A reference to the created VPC
    Value: !Ref VPC
    Export:
      Name: !Sub '${AWS::StackName}-VPCID'

  PublicSubnets:
    Description: A list of the public subnets
    Value: !Join [ ",", [ !Ref PublicSubnet1, !Ref PublicSubnet2 ] ]

  PublicSubnet1:
    Description: A reference to the public subnet in the 1st Availability Zone
    Value: !Ref PublicSubnet1
    Export:
      Name: !Sub '${AWS::StackName}-PublicSubnetID1'

  PublicSubnet2:
    Description: A reference to the public subnet in the 2nd Availability Zone
    Value: !Ref PublicSubnet2
    Export:
      Name: !Sub '${AWS::StackName}-PublicSubnetID2'
```

- Computing Resources Stack:



```
#####
# Resources section
#####

Resources:

LaunchTemplate:
  Type: AWS::EC2::LaunchTemplate
  Properties:
    LaunchTemplateData:
      ImageId: !Ref AmazonLinuxAMIID
      InstanceType: t3.micro
      NetworkInterfaces:
        - DeviceIndex: 0
          AssociatePublicAddress: true
          Groups:
            - !Ref WebServerSecurityGroup
      DeleteOnTermination: true
    Placement:
      Tenancy: default
    UserData:
      Fn::Base64:
        !Sub |
        #!/bin/bash
        sudo yum update -y
        sudo amazon-linux-extras install nginx1 -y
        sudo systemctl enable nginx
        sudo systemctl start nginx

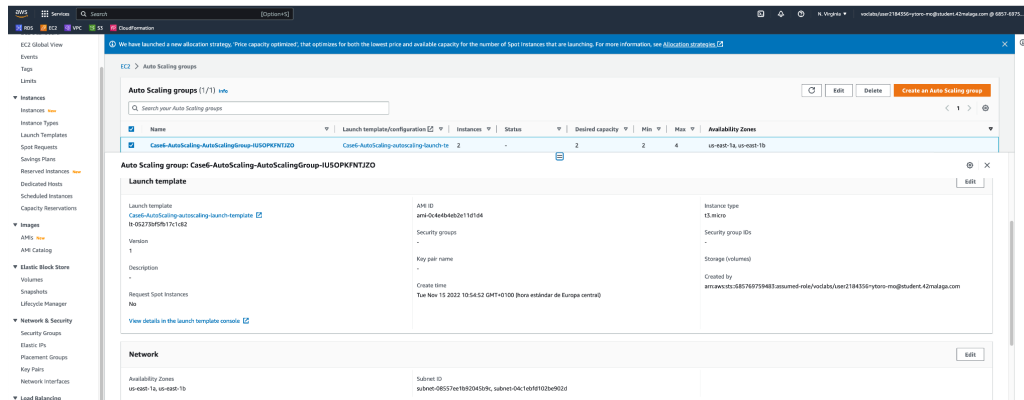
    LaunchTemplateName: !Sub '${AWS::StackName}-autoscaling-launch-template'

WebServerSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Enable HTTP ingress
    VpcId:
      Fn::ImportValue:
        !Sub ${NetworkStackName}-VPCID
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 80
        ToPort: 80
        CidrIp: 0.0.0.0/0
    Tags:
      - Key: Name
        Value: Web Server Security Group

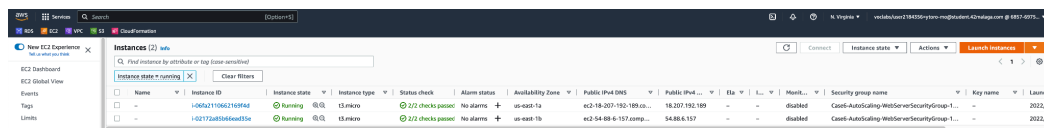
AutoScalingGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    MinSize: '2'
    MaxSize: '4'
    DesiredCapacity: '2'
    LaunchTemplate:
      LaunchTemplateId: !Ref LaunchTemplate
      Version: !Ref LaunchTemplateVersionNumber
    VPCZoneIdentifier:
      - Fn::ImportValue: !Sub ${NetworkStackName}-PublicSubnetID1
      - Fn::ImportValue: !Sub ${NetworkStackName}-PublicSubnetID2
```

Deployment:

- Auto Scaling Group:



- EC2 instances deployed:



- HTTP connection to nginx:

