

CLOUD FORMATION PROJECT

PROJECT RECOMMENDATION

MINI-PROJECT

Each student will describe a section of the network template (will be provided)

Step 1

- Deploy a sample network configuration stack with the network template provided.
- For stackname use: SampleNetworkCrossStack

Step 2

Deploy a new stack that will setup Apache on alinux EC2 instance and reference the sample network stack previously created.

- Deploy template in us-west-2 region. Allocate an Elastic IP address using resources.
- Reference all available variables from SampleNetworkCrossStack
- Create a Mapping function to dynamically choosing regions.
- Create parameters for requesting user inputs as much as possible.
- Output the values of the private IPv4 address and DNS name of the EC2 instance.
- Log in into the ec2 server afterward and provide screenshots Apache message. Create a Git repo apache-web-cfn and upload templates.

SAMPLE TEMPLATE WITH EXPORT VALUES:

```
Resources:
  VPC:
    Type: 'AWS::EC2::VPC'
    Properties:
      EnableDnsSupport: 'true'
      EnableDnsHostnames: 'true'
      CidrBlock: 10.0.0.0/16
  PublicSubnet:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref VPC
      CidrBlock: 10.0.0.0/24
  InternetGateway:
    Type: 'AWS::EC2::InternetGateway'
  VPCGatewayAttachment:
    Type: 'AWS::EC2::VPCGatewayAttachment'
    Properties:
      VpcId: !Ref VPC
      InternetGatewayId: !Ref InternetGateway
  PublicRouteTable:
    Type: 'AWS::EC2::RouteTable'
    Properties:
      VpcId: !Ref VPC
  PublicRoute:
    Type: 'AWS::EC2::Route'
    DependsOn: VPCGatewayAttachment # forcing depencing
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnetRouteTableAssociation:
    Type: 'AWS::EC2::SubnetRouteTableAssociation'
    Properties:
      SubnetId: !Ref PublicSubnet
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetNetworkAclAssociation:
    Type: 'AWS::EC2::SubnetNetworkAclAssociation'
    Properties:
      SubnetId: !Ref PublicSubnet
      NetworkAclId: !GetAtt
        - VPC
        - DefaultNetworkAcl
```

```

SecurityGroupIngress:
  - IpProtocol: tcp
    FromPort: '80'
    ToPort: '80'
    CidrIp: 0.0.0.0/0

Outputs:
  VPCId:
    Description: VPC ID
    Value: !Ref VPC
    Export:
      Name: !Sub '${AWS::StackName}-VPCID'
  PublicSubnet:
    Description: The subnet ID to use for public web servers
    Value: !Ref PublicSubnet
    Export:
      Name: !Sub '${AWS::StackName}-SubnetID' #replacing a string
  WebServerSecurityGroup:
    Description: The security group ID to use for public web servers
65   WebServerSecurityGroup:
66     Description: The security group ID to use for public web servers
67     Value: !GetAtt
68       - WebServerSecurityGroup
69       - GroupId
70     Export:
71       Name: !Sub '${AWS::StackName}-SecurityGroupID'

```

My template with import values from the sample template

```

1  ---
2  Parameters:
3    InstanceTypeParameter:
4      Type: String
5      Description: EC2 instance type for web server
6      Default: t2.micro
7
8  Mappings:
9    EC2InstanceMappings:
10     us-west-2:
11       Key: ami-0ab193018f3e9351b
12
13  Resources:
14    EC2Instance:
15      Type: 'AWS::EC2::Instance'
16      Properties:
17        InstanceType: !Ref InstanceTypeParameter
18        ImageId: !FindInMap [EC2InstanceMappings, !Ref 'AWS::Region', Key]
19        SecurityGroupIds:
20          - "sg-023fff6b4b14e8f82"
21        SubnetId: "subnet-0d76e346bf6279ec7"
22
23  UserData:
24    Fn::Base64: !Sub |
25      #!/bin/bash
26      yum update -y
27      yum install httpd -y
28      echo '<html><body><h1 style="color:people;">welcome to CloudFormation Class as Insf
29      systemctl start httpd
30      systemctl enable httpd
31
32  WebServerEIP:
33    Type: AWS::EC2::EIP
34    Properties:
35      InstanceId: !Ref EC2Instance
36
37  Outputs:
38    PrivateIPv4Address:

```

```
Outputs:
PrivateIPv4Address:
  Description: The private IPv4 address of the EC2 instance
  Value: !GetAtt EC2Instance.PrivateIp
InstanceDNSName:
  Description: The DNS name of the EC2 instance
  Value: !GetAtt EC2Instance.PublicDnsName
```

My stack creation

CloudFormation > Stacks > EmireineWebserverStack

Stacks (2)

Filter by stack name

Active View nested

Stacks

- EmireineWebserverStack
2023-05-27 23:47:26 UTC-0400
CREATE_IN_PROGRESS
- SampleNetworkCrossStack
2023-05-27 23:07:07 UTC-0400
CREATE_COMPLETE

EmireineWebserverStack

Delete Update Stack actions Create stack

Stack info Events Resources Outputs Parameters Template Ch

Events (8)

Search events

Timestamp	Logical ID	Status	Status reason
2023-05-27 23:48:22 UTC-0400	EmireineWebserverStack	CREATE_COMPLETE	-
2023-05-27 23:48:21 UTC-0400	WebServerEIP	CREATE_COMPLETE	-
2023-05-27 23:48:04 UTC-0400	WebServerEIP	CREATE_IN_PROGRESS	Resource creation initiated
2023-05-27 23:48:03 UTC-0400	WebServerEIP	CREATE_IN_PROGRESS	-
2023-05-27 23:48:02 UTC-0400	EC2Instance	CREATE_COMPLETE	-

EmireineWebserverStack

[Delete](#)[Update](#)[Stack actions](#)[Create stack](#)[Stack info](#)[Events](#)[Resources](#)[Outputs](#)[Parameters](#)[Template](#)[Ch](#)

Resources (2)



1



Logical ID	Physical ID	Type	Status
EC2Instance	i-0c7f7db76b3f3bb30	AWS::EC2::Instance	CREATE_COMPLETE
WebServerEIP	44.228.194.239	AWS::EC2::EIP	CREATE_COMPLETE

Apache server successfully launched

