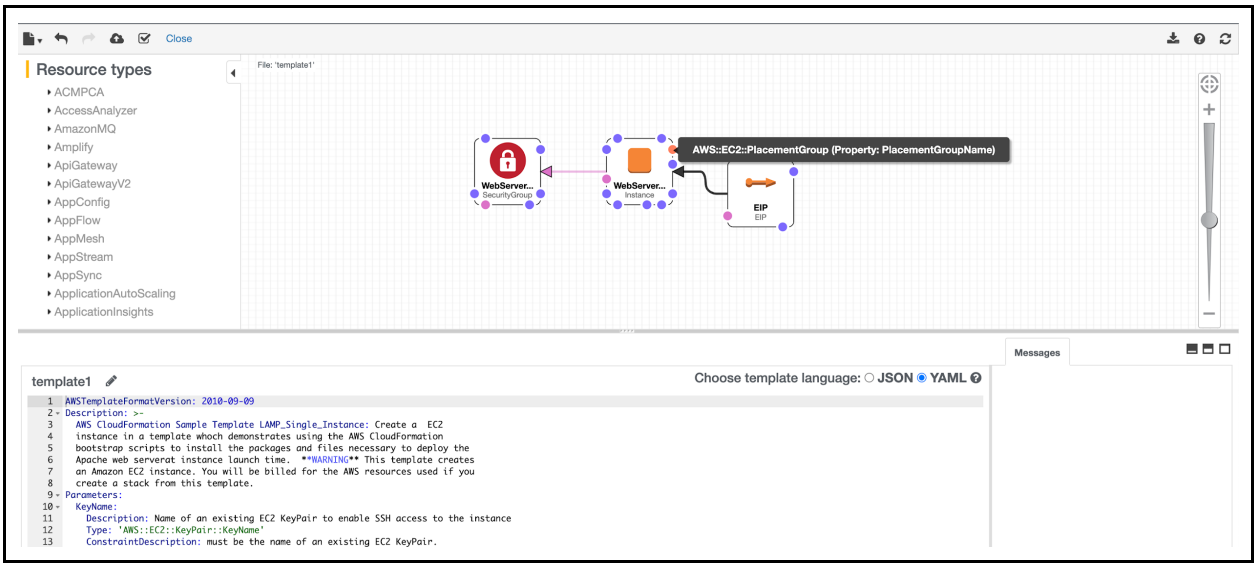
Rahul Ghosh
Homework 8: CloudFormation
(Total Points: 100) Due: Sunday November 1 11:59PM.

Build an Apache httpd stack (Points: 100) We've built many EC2 instances with an httpd installation during the semester, this time let's do it through a CloudFormation Template.

Create a template which contains an EC2 instance, and a security group.The following conditions must be met: -

1. The template should allow for the EC2 instance to be one of two instance types: either a **t2.micro, or a t2.small**. No other options should be available.
2. A **description** should be provided explaining exactly what is in this stack.
3. Stack creation **should prompt for a key pair to be used for SSH access to your EC2 instance**.
4. The **default IP address for SSH access into the EC2 instance** is your laptop/desktop IP address. This is referring to the default IP address that will be displayed during the first step of stack creation.
5. A **simple web page should be deployed automatically with a welcome message and your name displayed.** You can be as creative or as uncreative as you like.
6. The output from your stack creation should display your website URL, and the description for your output should be "'a simple website stack" - **An Elastic IP address should be used with your EC2 instance.**

CloudFormation diagram of your configuration: Your full template in either JSON or YAML format (no need to submit both). We should be able to import, validate and run your template successfully:



YAML CODE FOR STACK CREATION: -

```
AWSTemplateFormatVersion: 2010-09-09
Description: >-
  AWS CloudFormation Sample Template LAMP_Single_Instance: Create a  EC2
  instance in a template whoch demonstrates using the AWS CloudFormation
  bootstrap scripts to install the packages and files necessary to deploy the
  Apache web serverat instance launch time.  **WARNING** This template creates
  an Amazon EC2 instance. You will be billed for the AWS resources used if you
  create a stack from this template.
Parameters:
  KeyName:
    Description: Name of an existing EC2 KeyPair to enable SSH access to the instance
    Type: 'AWS::EC2::KeyPair::KeyName'
    ConstraintDescription: must be the name of an existing EC2 KeyPair.
  DBName:
    Default: MyDatabase
    Description: MySQL database name
    Type: String
    MinLength: '1'
    MaxLength: '64'
    AllowedPattern: '[a-zA-Z][a-zA-Z0-9]*'
    ConstraintDescription: must begin with a letter and contain only alphanumeric characters.
  DBUser:
    NoEcho: 'true'
    Description: Username for MySQL database access
    Type: String
    MinLength: '1'
    MaxLength: '16'
    AllowedPattern: '[a-zA-Z][a-zA-Z0-9]*'
    ConstraintDescription: must begin with a letter and contain only alphanumeric characters.
  DBPassword:
    NoEcho: 'true'
    Description: Password for MySQL database access
    Type: String
    MinLength: '1'
    MaxLength: '41'
    AllowedPattern: '[a-zA-Z0-9]*'
    ConstraintDescription: must contain only alphanumeric characters.
  DBRootPassword:
    NoEcho: 'true'
    Description: Root password for MySQL
    Type: String
    MinLength: '1'
    MaxLength: '41'
    AllowedPattern: '[a-zA-Z0-9]*'
    ConstraintDescription: must contain only alphanumeric characters.
  InstanceType:
    Description: WebServer EC2 instance type
    Type: String
    Default: t2.micro
    AllowedValues:
```

```
      - t2.micro
      - t2.small
    ConstraintDescription: must be a valid EC2 instance type.
  SSHLocation:
    Description: ' The IP address range that can be used to SSH to the EC2 instances'
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 75.169.147.148/32
    AllowedPattern: '(\d{1,3})\.(\d{1,3})\.(\d{1,3})\.(\d{1,3})/(\d{1,2})'
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
Mappings:
  AWSInstanceType2Arch:
    t1.micro:
      Arch: HVM64
    t2.nano:
      Arch: HVM64
    t2.micro:
      Arch: HVM64
    t2.small:
      Arch: HVM64
    t2.medium:
      Arch: HVM64
    t2.large:
      Arch: HVM64
    m1.small:
      Arch: HVM64
    m1.medium:
      Arch: HVM64
    m1.large:
      Arch: HVM64
    m1.xlarge:
      Arch: HVM64
    m2.xlarge:
      Arch: HVM64
    m2.2xlarge:
      Arch: HVM64
    m2.4xlarge:
      Arch: HVM64
    m3.medium:
      Arch: HVM64
    m3.large:
      Arch: HVM64
    m3.xlarge:
      Arch: HVM64
    m3.2xlarge:
      Arch: HVM64
    m4.large:
      Arch: HVM64
    m4.xlarge:
      Arch: HVM64
    m4.2xlarge:
```

```
  Arch: HVM64
m4.4xlarge:
  Arch: HVM64
m4.10xlarge:
  Arch: HVM64
c1.medium:
  Arch: HVM64
c1.xlarge:
  Arch: HVM64
c3.large:
  Arch: HVM64
c3.xlarge:
  Arch: HVM64
c3.2xlarge:
  Arch: HVM64
c3.4xlarge:
  Arch: HVM64
c3.8xlarge:
  Arch: HVM64
c4.large:
  Arch: HVM64
c4.xlarge:
  Arch: HVM64
c4.2xlarge:
  Arch: HVM64
c4.4xlarge:
  Arch: HVM64
c4.8xlarge:
  Arch: HVM64
g2.2xlarge:
  Arch: HVMG2
g2.8xlarge:
  Arch: HVMG2
r3.large:
  Arch: HVM64
r3.xlarge:
  Arch: HVM64
r3.2xlarge:
  Arch: HVM64
r3.4xlarge:
  Arch: HVM64
r3.8xlarge:
  Arch: HVM64
i2.xlarge:
  Arch: HVM64
i2.2xlarge:
  Arch: HVM64
i2.4xlarge:
  Arch: HVM64
i2.8xlarge:
  Arch: HVM64
d2.xlarge:
```

```
    Arch: HVM64
  d2.2xlarge:
    Arch: HVM64
  d2.4xlarge:
    Arch: HVM64
  d2.8xlarge:
    Arch: HVM64
  hi1.4xlarge:
    Arch: HVM64
  hs1.8xlarge:
    Arch: HVM64
  cr1.8xlarge:
    Arch: HVM64
  cc2.8xlarge:
    Arch: HVM64
AWSInstanceType2NATArch:
  t1.micro:
    Arch: NATHVM64
  t2.nano:
    Arch: NATHVM64
  t2.micro:
    Arch: NATHVM64
  t2.small:
    Arch: NATHVM64
  t2.medium:
    Arch: NATHVM64
  t2.large:
    Arch: NATHVM64
  m1.small:
    Arch: NATHVM64
  m1.medium:
    Arch: NATHVM64
  m1.large:
    Arch: NATHVM64
  m1.xlarge:
    Arch: NATHVM64
  m2.xlarge:
    Arch: NATHVM64
  m2.2xlarge:
    Arch: NATHVM64
  m2.4xlarge:
    Arch: NATHVM64
  m3.medium:
    Arch: NATHVM64
  m3.large:
    Arch: NATHVM64
  m3.xlarge:
    Arch: NATHVM64
  m3.2xlarge:
    Arch: NATHVM64
  m4.large:
    Arch: NATHVM64
```

```
m4.xlarge:
  Arch: NATHVM64
m4.2xlarge:
  Arch: NATHVM64
m4.4xlarge:
  Arch: NATHVM64
m4.10xlarge:
  Arch: NATHVM64
c1.medium:
  Arch: NATHVM64
c1.xlarge:
  Arch: NATHVM64
c3.large:
  Arch: NATHVM64
c3.xlarge:
  Arch: NATHVM64
c3.2xlarge:
  Arch: NATHVM64
c3.4xlarge:
  Arch: NATHVM64
c3.8xlarge:
  Arch: NATHVM64
c4.large:
  Arch: NATHVM64
c4.xlarge:
  Arch: NATHVM64
c4.2xlarge:
  Arch: NATHVM64
c4.4xlarge:
  Arch: NATHVM64
c4.8xlarge:
  Arch: NATHVM64
g2.2xlarge:
  Arch: NATHVMG2
g2.8xlarge:
  Arch: NATHVMG2
r3.large:
  Arch: NATHVM64
r3.xlarge:
  Arch: NATHVM64
r3.2xlarge:
  Arch: NATHVM64
r3.4xlarge:
  Arch: NATHVM64
r3.8xlarge:
  Arch: NATHVM64
i2.xlarge:
  Arch: NATHVM64
i2.2xlarge:
  Arch: NATHVM64
i2.4xlarge:
  Arch: NATHVM64
```

```
  i2.8xlarge:
    Arch: NATHVM64
  d2.xlarge:
    Arch: NATHVM64
  d2.2xlarge:
    Arch: NATHVM64
  d2.4xlarge:
    Arch: NATHVM64
  d2.8xlarge:
    Arch: NATHVM64
  hi1.4xlarge:
    Arch: NATHVM64
  hs1.8xlarge:
    Arch: NATHVM64
  cr1.8xlarge:
    Arch: NATHVM64
  cc2.8xlarge:
    Arch: NATHVM64
AWSRegionArch2AMI:
  us-east-1:
    HVM64: ami-0080e4c5bc078760e
    HVMG2: ami-0aeb704d503081ea6
  us-west-2:
    HVM64: ami-01e24be29428c15b2
    HVMG2: ami-0fe84a5b4563d8f27
  us-west-1:
    HVM64: ami-0ec6517f6edbf8044
    HVMG2: ami-0a7fc72dc0e51aa77
  eu-west-1:
    HVM64: ami-08935252a36e25f85
    HVMG2: ami-0d5299b1c6112c3c7
  eu-west-2:
    HVM64: ami-01419b804382064e4
    HVMG2: NOT_SUPPORTED
  eu-west-3:
    HVM64: ami-0dd7e7ed60da8fb83
    HVMG2: NOT_SUPPORTED
  eu-central-1:
    HVM64: ami-0cfbf4f6db41068ac
    HVMG2: ami-0aa1822e3eb913a11
  eu-north-1:
    HVM64: ami-86fe70f8
    HVMG2: ami-32d55b4c
  ap-northeast-1:
    HVM64: ami-00a5245b4816c38e6
    HVMG2: ami-09d0e0e099ecabba2
  ap-northeast-2:
    HVM64: ami-00dc207f8ba6dc919
    HVMG2: NOT_SUPPORTED
  ap-northeast-3:
    HVM64: ami-0b65f69a5c11f3522
    HVMG2: NOT_SUPPORTED
```

```
    ap-southeast-1:
      HVM64: ami-05b3bcf7f311194b3
      HVMG2: ami-0e46ce0d6a87dc979
    ap-southeast-2:
      HVM64: ami-02fd0b06f06d93dfc
      HVMG2: ami-0c0ab057a101d8ff2
    ap-south-1:
      HVM64: ami-0ad42f4f66f6c1cc9
      HVMG2: ami-0244c1d42815af84a
    us-east-2:
      HVM64: ami-0cd3dfa4e37921605
      HVMG2: NOT_SUPPORTED
    ca-central-1:
      HVM64: ami-07423fb63ea0a0930
      HVMG2: NOT_SUPPORTED
    sa-east-1:
      HVM64: ami-05145e0b28ad8e0b2
      HVMG2: NOT_SUPPORTED
    cn-north-1:
      HVM64: ami-053617c9d818c1189
      HVMG2: NOT_SUPPORTED
    cn-northwest-1:
      HVM64: ami-0f7937761741dc640
      HVMG2: NOT_SUPPORTED
Resources:
  WebServerInstance:
    Type: 'AWS::EC2::Instance'
    Metadata:
      'AWS::CloudFormation::Init':
        configSets:
          InstallAndRun:
            - Install
            - Configure
        Install:
          packages:
            yum:
              mysql: []
              mysql-server: []
              mysql-libs: []
              httpd: []
              php: []
              php-mysql: []
          files:
            /var/www/html/index.php:
              content: !Join
                - ''
                - - |
                    <html>
                  - |2
                      <head>
                  - |2
                        <title>Not that nice of a webpage - Rahul Ghosh :) </title>
```

```
            - |2
                <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
            - |2
               </head>
            - |2
                <body>
            - |2
                <h1>Welcome to the AWS CloudFormation Rahuls Site</h1>
            - |2
                <p/>
            - |2
                <?php
            - |2
                 // Print out the current data and time
            - |2
                 print "The Current Date and Time is: <br/>";
            - |2
                 print date("g:i A l, F j Y.");
            - |2
                ?>
            - |2
                <p/>
            - |2
                <?php
            - |2
                 // Setup a handle for CURL
            - |2
                 $curl_handle=curl_init();
            - |2
                 curl_setopt($curl_handle,CURLOPT_CONNECTTIMEOUT,2);
            - |2
                 curl_setopt($curl_handle,CURLOPT_RETURNTRANSFER,1);
            - |2
                 // Get the hostname of the intance from the instance metadata
            - |2
                 curl_setopt($curl_handle,CURLOPT_URL,'http://169.254.169.254/latest/meta-
data/public-hostname');
            - |2
                 $hostname = curl_exec($curl_handle);
            - |2
                 if (empty($hostname))
            - |2
                 {
            - |2
                   print "Sorry, for some reason, we got no hostname back <br />";
            - |2
                 }
            - |2
                 else
            - |2
                 {
            - |2
```

```
                  print "Server = " . $hostname . "<br />";
            - |2
                }
            - |2
                // Get the instance-id of the intance from the instance metadata
            - |2
                curl_setopt($curl_handle,CURLOPT_URL,'http://169.254.169.254/latest/meta-
data/instance-id');
            - |2
                $instanceid = curl_exec($curl_handle);
            - |2
                if (empty($instanceid))
            - |2
                {
            - |2
                  print "Sorry, for some reason, we got no instance id back <br />";
            - |2
                }
            - |2
                else
            - |2
                {
            - |2
                  print "EC2 instance-id = " . $instanceid . "<br />";
            - |2
                }
            - |2
                $Database   = "localhost";
            - '     $DBUser     = "'
            - !Ref DBUser
            - |
              ";
            - '     $DBPassword = "'
            - !Ref DBPassword
            - |
              ";
            - |2
                print "Database = " . $Database . "<br />";
            - |2
                $dbconnection = mysql_connect($Database, $DBUser, $DBPassword)
            - |2
                        or die("Could not connect: " . mysql_error());
            - |2
                print ("Connected to $Database successfully");
            - |2
                mysql_close($dbconnection);
            - |2
               ?>
            - |2
               <h2>PHP Information</h2>
            - |2
               <p/>
```

```
        - |2
            <?php
        - |2
             phpinfo();
        - |2
            ?>
        - |2
           </body>
        - |
          </html>
    mode: '000600'
    owner: apache
    group: apache
  /tmp/setup.mysql:
    content: !Join
      - ''
      - - 'CREATE DATABASE '
        - !Ref DBName
        - |
          ;
        - 'GRANT ALL ON '
        - !Ref DBName
        - .* TO '
        - !Ref DBUser
        - '''@localhost IDENTIFIED BY '''
        - !Ref DBPassword
        - |
          ';
    mode: '000400'
    owner: root
    group: root
  /etc/cfn/cfn-hup.conf:
    content: !Join
      - ''
      - - |
          [main]
        - stack=
        - !Ref 'AWS::StackId'
        - |+

        - region=
        - !Ref 'AWS::Region'
        - |+

    mode: '000400'
    owner: root
    group: root
  /etc/cfn/hooks.d/cfn-auto-reloader.conf:
    content: !Join
      - ''
      - - |
          [cfn-auto-reloader-hook]
```

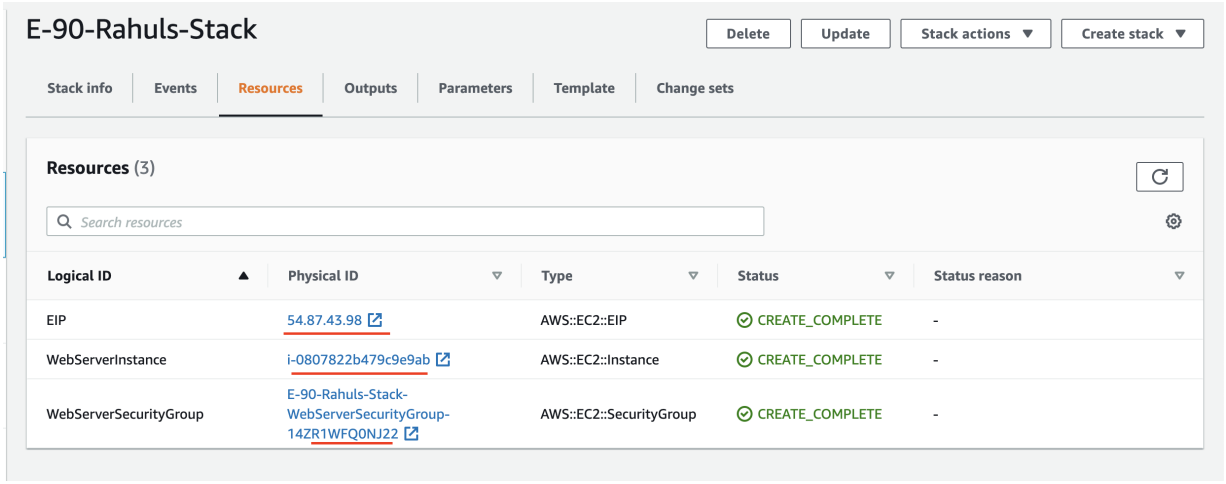
```
          - |
            triggers=post.update
          - >
            path=Resources.WebServerInstance.Metadata.AWS::CloudFormation::Init
          - 'action=/opt/aws/bin/cfn-init -v '
          - '         --stack '
          - !Ref 'AWS::StackName'
          - '         --resource WebServerInstance '
          - '         --configsets InstallAndRun '
          - '         --region '
          - !Ref 'AWS::Region'
          - |+

          - |
            runas=root
        mode: '000400'
        owner: root
        group: root
    services:
      sysvinit:
        mysqld:
          enabled: 'true'
          ensureRunning: 'true'
        httpd:
          enabled: 'true'
          ensureRunning: 'true'
        cfn-hup:
          enabled: 'true'
          ensureRunning: 'true'
          files:
            - /etc/cfn/cfn-hup.conf
            - /etc/cfn/hooks.d/cfn-auto-reloader.conf
  Configure:
    commands:
      01_set_mysql_root_password:
        command: !Join
          - ''
          - - mysqladmin -u root password '
            - !Ref DBRootPassword
            - ''''
        test: !Join
          - ''
          - - '$(mysql '
            - !Ref DBName
            - ' -u root --password='''
            - !Ref DBRootPassword
            - ''' >/dev/null 2>&1 </dev/null); (( $? != 0 ))'
      02_create_database:
        command: !Join
          - ''
          - - mysql -u root --password='
            - !Ref DBRootPassword
```

```
            - "' < /tmp/setup.mysql'
          test: !Join
            - ''
            - - '$(mysql '
              - !Ref DBName
              - ' -u root --password="'
              - !Ref DBRootPassword
              - '" >/dev/null 2>&1 </dev/null); (( $? != 0 ))'
    'AWS::CloudFormation::Designer':
      id: bc9952b7-0ed1-4da4-a1b7-9583a0e68d9b
  Properties:
    ImageId: !FindInMap
      - AWSRegionArch2AMI
      - !Ref 'AWS::Region'
      - !FindInMap
        - AWSInstanceType2Arch
        - !Ref InstanceType
        - Arch
    InstanceType: !Ref InstanceType
    KeyName: !Ref KeyName
    UserData: !Base64
      'Fn::Join':
        - ''
        - - |
            #!/bin/bash -xe
          - |
            yum update -y aws-cfn-bootstrap
          - |
            # Install the files and packages from the metadata
          - '/opt/aws/bin/cfn-init -v '
          - '         --stack '
          - !Ref 'AWS::StackName'
          - '         --resource WebServerInstance '
          - '         --configsets InstallAndRun '
          - '         --region '
          - !Ref 'AWS::Region'
          - |+

          - |
            # Signal the status from cfn-init
          - '/opt/aws/bin/cfn-signal -e $? '
          - '         --stack '
          - !Ref 'AWS::StackName'
          - '         --resource WebServerInstance '
          - '         --region '
          - !Ref 'AWS::Region'
          - |+

    CreationPolicy:
      ResourceSignal:
        Timeout: PT5M
  WebServerSecurityGroup:
```

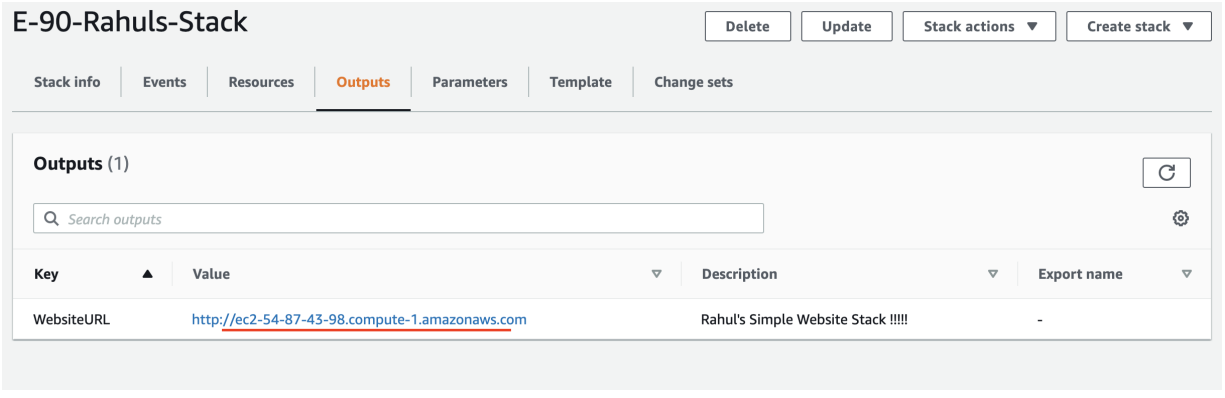
```
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Enable HTTP access via port 80
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: '80'
          ToPort: '80'
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: '22'
          ToPort: '22'
          CidrIp: !Ref SSHLocation
    Metadata:
      'AWS::CloudFormation::Designer':
        id: 48cb5cdf-6968-4aa5-a663-5e596b10fa3c
  EIP:
    Type: 'AWS::EC2::EIP'
    Properties:
      InstanceId: !Ref WebServerInstance
    Metadata:
      'AWS::CloudFormation::Designer':
        id: c159695c-4056-45e9-ae8e-6d1a736303eb
Outputs:
  WebsiteURL:
    Description: Rahul's Simple Website Stack !!!!!
    Value: !Join
      - ''
      - - 'http://'
        - !GetAtt
          - WebServerInstance
          - PublicDnsName
Metadata:
  'AWS::CloudFormation::Designer':
    48cb5cdf-6968-4aa5-a663-5e596b10fa3c:
      size:
        width: 60
        height: 60
      position:
        x: 60
        'y': 90
      z: 1
      embeds: []
    bc9952b7-0ed1-4da4-a1b7-9583a0e68d9b:
      size:
        width: 60
        height: 60
      position:
        x: 180
        'y': 90
      z: 1
      embeds: []
    c159695c-4056-45e9-ae8e-6d1a736303eb:
```

```
size:
  width: 60
  height: 60
position:
  x: 290
  'y': 110
z: 0
embeds: []
isassociatedwith:
  - bc9952b7-0ed1-4da4-a1b7-9583a0e68d9b
```

EIP, EC2 AND SECURITY GROUP CREATION



OUTPUT: SIMPLE WEBPAGE



Bonus: (10 Points) Create a template that will generate a stack with the ALB and ASG exactly as we constructed them in Homework 7. CloudFormation diagram of your configuration: Your full template in either JSON or YAML format (no need to submit both). We should be able to import, validate and run your template successfully:

ASG CREATION WITH LOAD BALANCERS

```
AWSTemplateFormatVersion: 2010-09-09
Description: >-
  AWS CloudFormation Sample Template AutoScalingMultiAZWithNotifications: Create
  a multi-az, load balanced and auto-scaled sample web site running on an Apache
  Web Server. The application is configured to span all Availability Zones in
  the region and is auto-scaled based on the CPU utilization of the web servers.
  Notifications will be sent to the operator email address on scaling events.
  The instances are load balanced with a simple health check against the default
  web page. **WARNING** This template creates one or more Amazon EC2 instances
  and an Elastic Load Balancer.
Parameters:
  InstanceType:
    Description: WebServer EC2 instance type
    Type: String
    Default: t2.small
    AllowedValues:
      - t2.micro
      - t2.small
    ConstraintDescription: must be a valid EC2 instance type.
  OperatorEMail:
    Description: EMail address to notify if there are any scaling operations
    Type: String
    AllowedPattern: >-
      ([a-zA-Z0-9_\-\.]+)@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.)|(([a-zA-Z0-9\-]+\.)+))([a-zA-Z]{2,4}|[0-
9]{1,3})(\]?)
    ConstraintDescription: must be a valid email address.
  KeyName:
    Description: The EC2 Key Pair to allow SSH access to the instances
    Type: 'AWS::EC2::KeyPair::KeyName'
    ConstraintDescription: must be the name of an existing EC2 KeyPair.
  SSHLocation:
    Description: The IP address range that can be used to SSH to the EC2 instances
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 75.169.147.148/32
    AllowedPattern: '(\d{1,3})\.(\d{1,3})\.(\d{1,3})\.(\d{1,3})/(\d{1,2})'
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
Mappings:
```

```
AWSInstanceType2Arch:
  t1.micro:
    Arch: HVM64
  t2.nano:
    Arch: HVM64
  t2.micro:
    Arch: HVM64
  t2.small:
    Arch: HVM64
  t2.medium:
    Arch: HVM64
  t2.large:
    Arch: HVM64
  m1.small:
    Arch: HVM64
  m1.medium:
    Arch: HVM64
  m1.large:
    Arch: HVM64
  m1.xlarge:
    Arch: HVM64
  m2.xlarge:
    Arch: HVM64
  m2.2xlarge:
    Arch: HVM64
  m2.4xlarge:
    Arch: HVM64
  m3.medium:
    Arch: HVM64
  m3.large:
    Arch: HVM64
  m3.xlarge:
    Arch: HVM64
  m3.2xlarge:
    Arch: HVM64
  m4.large:
    Arch: HVM64
  m4.xlarge:
    Arch: HVM64
  m4.2xlarge:
    Arch: HVM64
  m4.4xlarge:
    Arch: HVM64
  m4.10xlarge:
    Arch: HVM64
  c1.medium:
```

```
    Arch: HVM64
  c1.xlarge:
    Arch: HVM64
  c3.large:
    Arch: HVM64
  c3.xlarge:
    Arch: HVM64
  c3.2xlarge:
    Arch: HVM64
  c3.4xlarge:
    Arch: HVM64
  c3.8xlarge:
    Arch: HVM64
  c4.large:
    Arch: HVM64
  c4.xlarge:
    Arch: HVM64
  c4.2xlarge:
    Arch: HVM64
  c4.4xlarge:
    Arch: HVM64
  c4.8xlarge:
    Arch: HVM64
  g2.2xlarge:
    Arch: HVMG2
  g2.8xlarge:
    Arch: HVMG2
  r3.large:
    Arch: HVM64
  r3.xlarge:
    Arch: HVM64
  r3.2xlarge:
    Arch: HVM64
  r3.4xlarge:
    Arch: HVM64
  r3.8xlarge:
    Arch: HVM64
  i2.xlarge:
    Arch: HVM64
  i2.2xlarge:
    Arch: HVM64
  i2.4xlarge:
    Arch: HVM64
  i2.8xlarge:
    Arch: HVM64
  d2.xlarge:
```

```
    Arch: HVM64
  d2.2xlarge:
    Arch: HVM64
  d2.4xlarge:
    Arch: HVM64
  d2.8xlarge:
    Arch: HVM64
  hi1.4xlarge:
    Arch: HVM64
  hs1.8xlarge:
    Arch: HVM64
  cr1.8xlarge:
    Arch: HVM64
  cc2.8xlarge:
    Arch: HVM64
AWSRegionArch2AMI:
  us-east-1:
    HVM64: ami-0ff8a91507f77f867
    HVMG2: ami-0a584ac55a7631c0c
  us-west-2:
    HVM64: ami-a0cfeed8
    HVMG2: ami-0e09505bc235aa82d
  us-west-1:
    HVM64: ami-0bdb828fd58c52235
    HVMG2: ami-066ee5fd4a9ef77f1
  eu-west-1:
    HVM64: ami-047bb4163c506cd98
    HVMG2: ami-0a7c483d527806435
  eu-west-2:
    HVM64: ami-f976839e
    HVMG2: NOT_SUPPORTED
  eu-west-3:
    HVM64: ami-0ebc281c20e89ba4b
    HVMG2: NOT_SUPPORTED
  eu-central-1:
    HVM64: ami-0233214e13e500f77
    HVMG2: ami-06223d46a6d0661c7
  ap-northeast-1:
    HVM64: ami-06cd52961ce9f0d85
    HVMG2: ami-053cdd503598e4a9d
  ap-northeast-2:
    HVM64: ami-0a10b2721688ce9d2
    HVMG2: NOT_SUPPORTED
  ap-northeast-3:
    HVM64: ami-0d98120a9fb693f07
    HVMG2: NOT_SUPPORTED
```

```
    ap-southeast-1:
      HVM64: ami-08569b978cc4dfa10
      HVMG2: ami-0be9df32ae9f92309
    ap-southeast-2:
      HVM64: ami-09b42976632b27e9b
      HVMG2: ami-0a9ce9fecc3d1daf8
    ap-south-1:
      HVM64: ami-0912f71e06545ad88
      HVMG2: ami-097b15e89dbdcfcf4
    us-east-2:
      HVM64: ami-0b59bfac6be064b78
      HVMG2: NOT_SUPPORTED
    ca-central-1:
      HVM64: ami-0b18956f
      HVMG2: NOT_SUPPORTED
    sa-east-1:
      HVM64: ami-07b14488da8ea02a0
      HVMG2: NOT_SUPPORTED
    cn-north-1:
      HVM64: ami-0a4eaf6c4454eda75
      HVMG2: NOT_SUPPORTED
    cn-northwest-1:
      HVM64: ami-6b6a7d09
      HVMG2: NOT_SUPPORTED
Resources:
  NotificationTopic:
    Type: 'AWS::SNS::Topic'
    Properties:
      Subscription:
        - Endpoint: !Ref OperatorEMail
          Protocol: email
  WebServerInstance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ImageId: !FindInMap
        - AWSRegionArch2AMI
        - !Ref 'AWS::Region'
        - !FindInMap
          - AWSInstanceType2Arch
          - !Ref InstanceType
          - Arch
      InstanceType: !Ref InstanceType
      SecurityGroups:
        - !Ref WebServerSecurityGroup
      KeyName: !Ref KeyName
  WebServerGroup:
```

```
  Type: 'AWS::AutoScaling::AutoScalingGroup'
  Properties:
    AvailabilityZones: !GetAZs ''
    LaunchConfigurationName: !Ref LaunchConfig
    MinSize: '1'
    MaxSize: '3'
    LoadBalancerNames:
      - !Ref ElasticLoadBalancer
    NotificationConfiguration:
      TopicARN: !Ref NotificationTopic
      NotificationTypes:
        - 'autoscaling:EC2_INSTANCE_LAUNCH'
        - 'autoscaling:EC2_INSTANCE_LAUNCH_ERROR'
        - 'autoscaling:EC2_INSTANCE_TERMINATE'
        - 'autoscaling:EC2_INSTANCE_TERMINATE_ERROR'
  CreationPolicy:
    ResourceSignal:
      Timeout: PT15M
      Count: '1'
  UpdatePolicy:
    AutoScalingRollingUpdate:
      MinInstancesInService: '1'
      MaxBatchSize: '1'
      PauseTime: PT15M
      WaitOnResourceSignals: 'true'
LaunchConfig:
  Type: 'AWS::AutoScaling::LaunchConfiguration'
  Metadata:
    Comment: Install a simple application
    'AWS::CloudFormation::Init':
      config:
        packages:
          yum:
            httpd: []
        files:
          /var/www/html/index.html:
            content: !Join
              - |+

              - - <img src="
                - !FindInMap
                  - Region2Examples
                  - !Ref 'AWS::Region'
                  - Examples
                - /cloudformation_graphic.png" alt="AWS CloudFormation Logo"/>
                - <h1>This is the bonus question answer</h1>
```

```
      mode: '000644'
      owner: root
      group: root
    /etc/cfn/cfn-hup.conf:
      content: !Join
        - ''
        - - |
            [main]
          - stack=
          - !Ref 'AWS::StackId'
          - |+

          - region=
          - !Ref 'AWS::Region'
          - |+

      mode: '000400'
      owner: root
      group: root
    /etc/cfn/hooks.d/cfn-auto-reloader.conf:
      content: !Join
        - ''
        - - |
            [cfn-auto-reloader-hook]
          - |
            triggers=post.update
          - >
            path=Resources.LaunchConfig.Metadata.AWS::CloudFormation::Init
          - 'action=/opt/aws/bin/cfn-init -v '
          - '         --stack '
          - !Ref 'AWS::StackName'
          - '         --resource LaunchConfig '
          - '         --region '
          - !Ref 'AWS::Region'
          - |+

          - |
            runas=root
  services:
    sysvinit:
      httpd:
        enabled: 'true'
        ensureRunning: 'true'
      cfn-hup:
        enabled: 'true'
        ensureRunning: 'true'
```

```
            files:
              - /etc/cfn/cfn-hup.conf
              - /etc/cfn/hooks.d/cfn-auto-reloader.conf
  Properties:
    KeyName: !Ref KeyName
    ImageId: !FindInMap
      - AWSRegionArch2AMI
      - !Ref 'AWS::Region'
      - !FindInMap
        - AWSInstanceType2Arch
        - !Ref InstanceType
        - Arch
    SecurityGroups:
      - !Ref WebServerSecurityGroup
    InstanceType: !Ref InstanceType
    UserData: !Base64
      'Fn::Join':
        - ''
        - - |
            #!/bin/bash -xe
          - |
            yum update -y aws-cfn-bootstrap
          - '/opt/aws/bin/cfn-init -v '
          - '         --stack '
          - !Ref 'AWS::StackName'
          - '         --resource LaunchConfig '
          - '         --region '
          - !Ref 'AWS::Region'
          - |+

          - '/opt/aws/bin/cfn-signal -e $? '
          - '         --stack '
          - !Ref 'AWS::StackName'
          - '         --resource WebServerGroup '
          - '         --region '
          - !Ref 'AWS::Region'
          - |+

  WebServerScaleUpPolicy:
    Type: 'AWS::AutoScaling::ScalingPolicy'
    Properties:
      AdjustmentType: ChangeInCapacity
      AutoScalingGroupName: !Ref WebServerGroup
      Cooldown: '60'
      ScalingAdjustment: '1'
  WebServerScaleDownPolicy:
```

```
    Type: 'AWS::AutoScaling::ScalingPolicy'
    Properties:
      AdjustmentType: ChangeInCapacity
      AutoScalingGroupName: !Ref WebServerGroup
      Cooldown: '60'
      ScalingAdjustment: '-1'
  CPUAlarmHigh:
    Type: 'AWS::CloudWatch::Alarm'
    Properties:
      AlarmDescription: Scale-up if CPU > 90% for 10 minutes
      MetricName: CPUUtilization
      Namespace: AWS/EC2
      Statistic: Average
      Period: '300'
      EvaluationPeriods: '2'
      Threshold: '90'
      AlarmActions:
        - !Ref WebServerScaleUpPolicy
      Dimensions:
        - Name: AutoScalingGroupName
          Value: !Ref WebServerGroup
      ComparisonOperator: GreaterThanThreshold
  CPUAlarmLow:
    Type: 'AWS::CloudWatch::Alarm'
    Properties:
      AlarmDescription: Scale-down if CPU < 70% for 10 minutes
      MetricName: CPUUtilization
      Namespace: AWS/EC2
      Statistic: Average
      Period: '300'
      EvaluationPeriods: '2'
      Threshold: '70'
      AlarmActions:
        - !Ref WebServerScaleDownPolicy
      Dimensions:
        - Name: AutoScalingGroupName
          Value: !Ref WebServerGroup
      ComparisonOperator: LessThanThreshold
  ElasticLoadBalancer:
    Type: 'AWS::ElasticLoadBalancing::LoadBalancer'
    Properties:
      AvailabilityZones: !GetAZs ''
      CrossZone: 'true'
      Listeners:
        - LoadBalancerPort: '80'
          InstancePort: '80'
```

```
        Protocol: HTTP
      HealthCheck:
        Target: 'HTTP:80/'
        HealthyThreshold: '3'
        UnhealthyThreshold: '5'
        Interval: '30'
        Timeout: '5'
  WebServerSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Enable SSH access and HTTP from the load balancer only
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: '22'
          ToPort: '22'
          CidrIp: !Ref SSHLocation
        - IpProtocol: tcp
          FromPort: '80'
          ToPort: '80'
          SourceSecurityGroupOwnerId: !GetAtt
            - ElasticLoadBalancer
            - SourceSecurityGroup.OwnerAlias
          SourceSecurityGroupName: !GetAtt
            - ElasticLoadBalancer
            - SourceSecurityGroup.GroupName
Outputs:
  URL:
    Description: Rahuls Simple Website Stack !!!
    Value: !Join
      - ''
      - - 'http://'
        - !GetAtt
          - ElasticLoadBalancer
          - DNSName
```