1. Start a EC2 Instance with Parameter KEY NAME, VPC, Subnet using Cloudformation Template.

Create yml file and set up all the parameters you want to take as input.

home > babanjot > exercise > cloudformationassignment > ! Parameters.yml

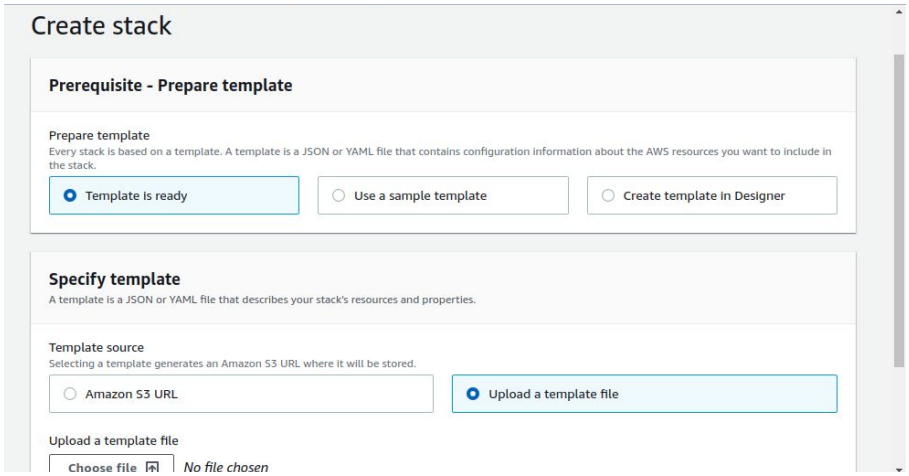
```yaml
Parameters:
  InstanceType:
    Description: WebServer EC2 instance type (has default, AllowedValues)
    Type: String
    Default: t2.micro
    AllowedValues:
      - t1.micro
      - t2.nano
      - t2.micro
      - t2.small
    ConstraintDescription: must be a valid EC2 instance type.
  KeyName:
    Description: Name of an existing EC2 KeyPair to enable SSH access to the instances. Linked to AWS Parameter
    Type: AWS::EC2::KeyPair::KeyName
    ConstraintDescription: must be the name of an existing EC2 KeyPair.
  MyVPC:
    Description: VPC to operate in
    Type: AWS::EC2::VPC::Id
  MySubnetIDs:
    Description: Subnet IDs that is a List of Subnet Id
    Type: "List<AWS::EC2::Subnet::Id>"
  SubnetIpBlocks:
    Description: "Comma-delimited list of three CIDR blocks"
    Type: CommaDelimitedList
    Default: "10.0.48.0/24, 10.0.112.0/24,"

Resources:
  MyEC2Instance:
    Type: "AWS::EC2::Instance"
    Properties:
      #we reference the InstanceType parameter
      InstanceType: !Ref InstanceType
```

Define the resources

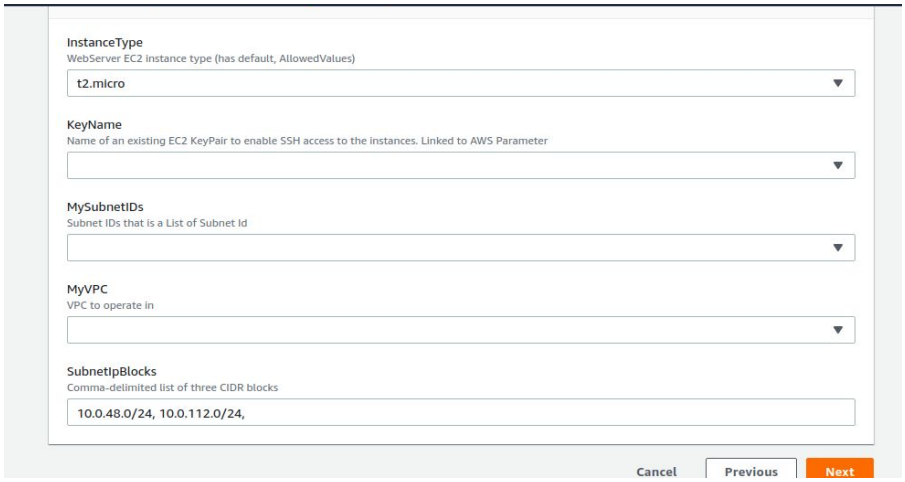
```yaml
    Default: "10.0.48.0/24, 10.0.112.0/24,"

Resources:
  MyEC2Instance:
    Type: "AWS::EC2::Instance"
    Properties:
      #we reference the InstanceType parameter
      InstanceType: !Ref InstanceType
      KeyName: !Ref KeyName
      ImageId: "ami-a4c7edb2"
      # here we reference an internal CloudFormation resource
      SubnetId: !Ref Subnet1
  Subnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref MyVPC
      # the select function allows us to select across a list
      CidrBlock: !Select [0, !Ref SubnetIpBlocks]
  Subnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref MyVPC
      # the select function allows us to select across a list
      CidrBlock: !Select [1, !Ref SubnetIpBlocks]
```

ster*+! ⊗ 0 ⚠ 0

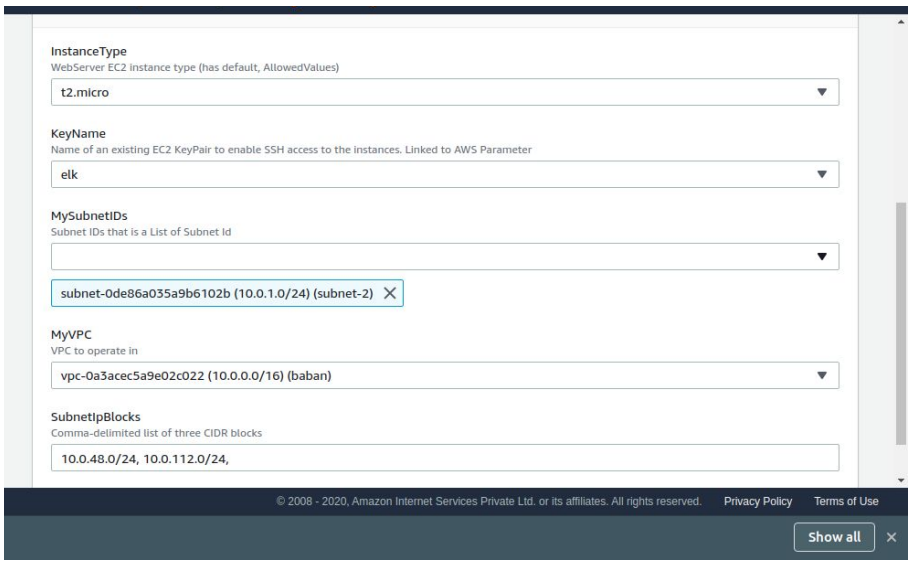
Go to cloudformation and create a stack and upload youryml file.



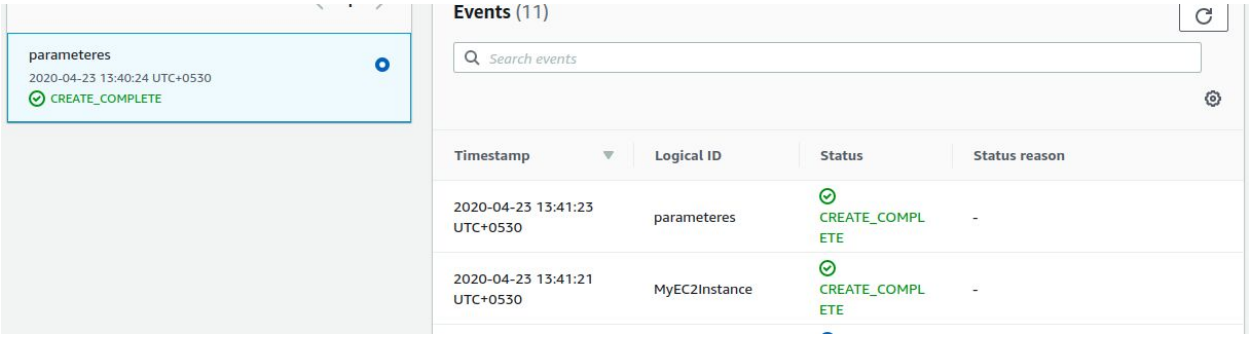
Now Enter all the values



Choose the correct configuration.



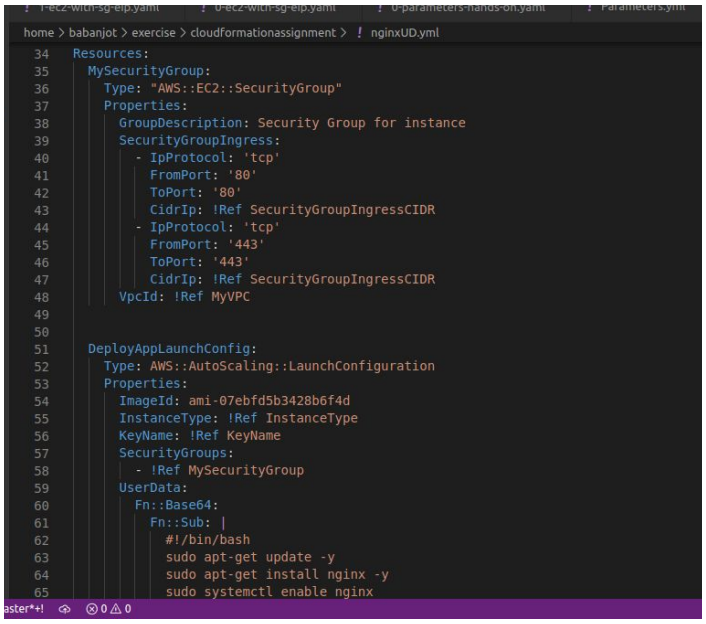
Now stack is up.



2. Instll Nginx in EC2 and put it in a ASG.

i) First by Userdata

Create yml file




```yaml
Resources:
  MySecurityGroup:
    Type: "AWS::EC2::SecurityGroup"
    Properties:
      GroupDescription: Security Group for instance
      SecurityGroupIngress:
        - IpProtocol: 'tcp'
          FromPort: '80'
          ToPort: '80'
          CidrIp: !Ref SecurityGroupIngressCIDR
        - IpProtocol: 'tcp'
          FromPort: '443'
          ToPort: '443'
          CidrIp: !Ref SecurityGroupIngressCIDR
      VpcId: !Ref MyVPC


  DeployAppLaunchConfig:
    Type: AWS::AutoScaling::LaunchConfiguration
    Properties:
      ImageId: ami-07ebfd5b3428b6f4d
      InstanceType: !Ref InstanceType
      KeyName: !Ref KeyName
      SecurityGroups:
        - !Ref MySecurityGroup
      UserData:
        Fn::Base64:
          Fn::Sub: |
            #!/bin/bash
            sudo apt-get update -y
            sudo apt-get install nginx -y
            sudo systemctl enable nginx
```

Write resource block for ASG and also userdata

```yaml
DeployAppLaunchConfig:
  Type: AWS::AutoScaling::LaunchConfiguration
  Properties:
    ImageId: ami-07ebfd5b3428b6f4d
    InstanceType: !Ref InstanceType
    KeyName: !Ref KeyName
    SecurityGroups:
      - !Ref MySecurityGroup
    UserData:
      Fn::Base64:
        Fn::Sub: |
          #!/bin/bash
          sudo apt-get update -y
          sudo apt-get install nginx -y
          sudo systemctl enable nginx
          sudo systemctl start nginx

DeployAppASG:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    VPCZoneIdentifier:
      - !Ref MySubnetIDa
      - !Ref MySubnetIDb
    DesiredCapacity: 1
```

⊗ 0 ⚠ 0 Ln 84, Col 26 Spaces: 2 UTF-8 LF YAML

File Edit Selection View Go Run Terminal Help

! 1-ec2-with-sg-eip.yaml ! 0-ec2-with-sg-eip.yaml ! 0-parameters-hands-on.yaml

home > babanjot > exercise > cloudformationassignment > ! nginxUD.yml

```yaml
DeployAppASG:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    VPCZoneIdentifier:
      - !Ref MySubnetIDa
      - !Ref MySubnetIDb
    DesiredCapacity: 1
    LaunchConfigurationName: !Ref DeployAppLaunchConfig
    MaxSize: 2
    MinSize: 1
    Tags:
      - Key: Name
        Value: Babanjot-ASG
        PropagateAtLaunch: True
  UpdatePolicy:
    AutoScalingReplacingUpdate:
      WillReplace: True
```

Now upload the yml file and create stack in CLoud Formation

Create stack

Prerequisite - Prepare template

Prepare template

Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you wa
the stack.

- (●) Template is ready
- ( ) Use a sample template
- ( ) Create template in Desi

A template is a JSON or YAML file that describes your stack's resources and properties.

Template source

Selecting a template generates an Amazon S3 URL where it will be stored.

- ( ) Amazon S3 URL
- (●) Upload a template file

Upload a template file

Choose file *nginxUD.yml*

JSON or YAML formatted file

S3 URL: https://s3-external-1.amazonaws.com/cf-templates-1445bji7u3na9-us-east-1/2020114zUM-nginxUD.yml

View in Designer

Choose the correct configuration and then create the stack

source Groups | EC2 | IAM | VPC | Lambda | Babanjot Singh | N. Virginia | Support

t2.micro

KeyName

Name of an existing EC2 KeyPair to enable SSH access to the instances.

elk

MySubnetIDa

Subnet IDs that is a List of Subnet Id

subnet-0b405d0b76ce7f054 (10.0.0.0/24) (subnet-1)

MySubnetIDb

Subnet IDs that is a List of Subnet Id

subnet-0de86a035a9b6102b (10.0.1.0/24) (subnet-2)

MyVPC

VPC to operate in

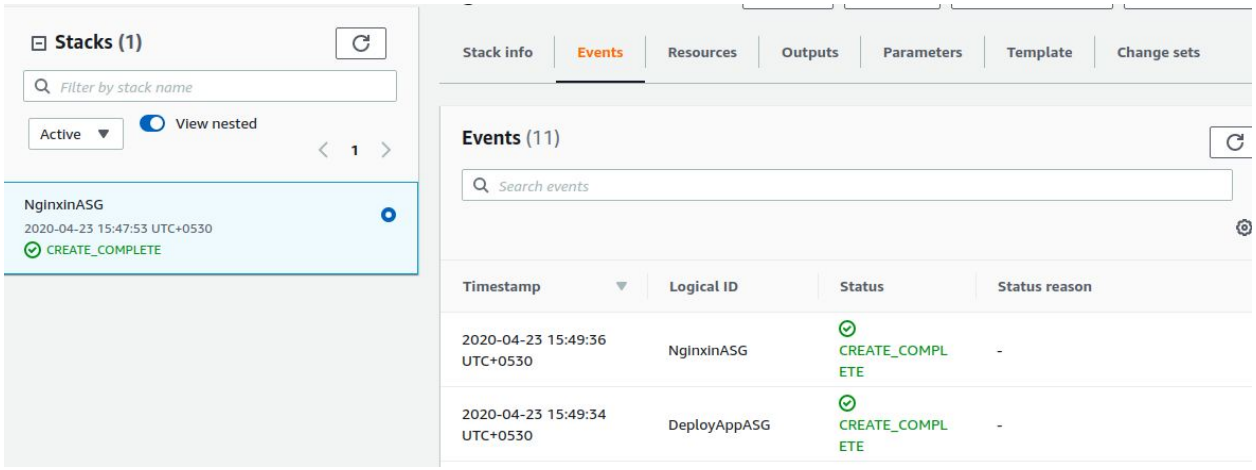
vpc-0a3acec5a9e02c022 (10.0.0.0/16) (baban)

SecurityGroupIngressCIDR

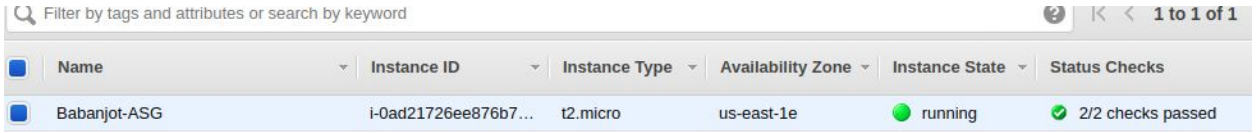
The IP address range that can be used to communicate to the EC2 instances

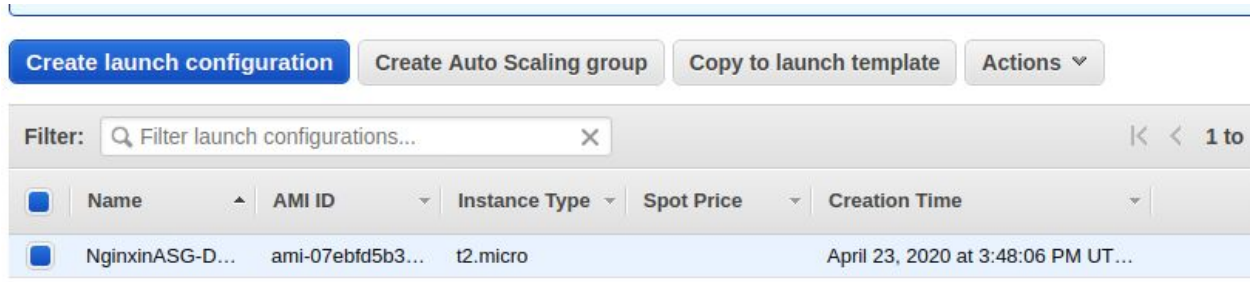
0.0.0.0/0

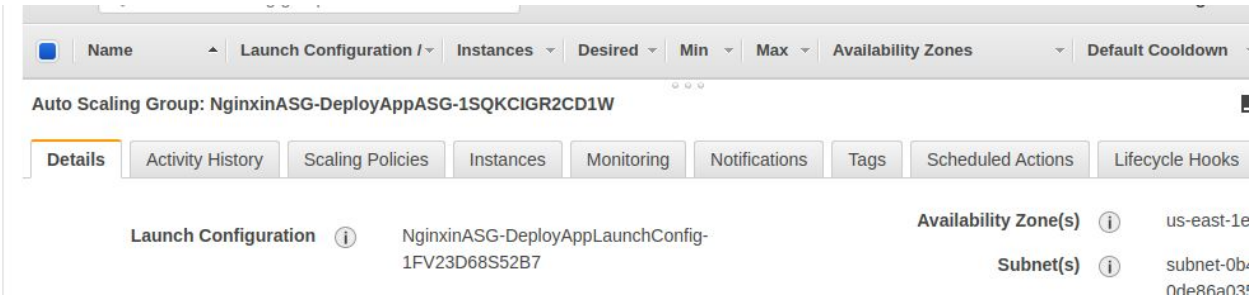
Stack is created



Check your infrastructure.







ii) By Metadata

Activities Visual Studio Code Tue 3:39 PM

stackq2.yml - Visual Studio Code

File Edit Selection View Go Run Terminal Help

stackq2.yml

home > babanjot > Downloads > stackq2.yml

```yaml
Parameters:
  MySubnetIDa:
    Description: Subnet IDs that is a List of Subnet Id
    Type: AWS::EC2::Subnet::Id
  MySubnetIDb:
    Description: Subnet IDs that is a List of Subnet Id
    Type: AWS::EC2::Subnet::Id


Resources:

  NginxLaunchTemplate:
    Type: AWS::EC2::LaunchTemplate
    Metadata:
      AWS::CloudFormation::Init:
        config:
          packages:
            yum:
              nginx: []
    Properties:
      LaunchTemplateName: "Nginx-Template-MD"
      LaunchTemplateData:
        InstanceType: t2.micro
        KeyName: elk
        ImageId: ami-0915e09cc7ceee3ab
        SecurityGroupIds:
          - sg-0189301dd7c4863a2
        UserData:
          Fn::Base64:
            Fn::Sub: |
              #!/bin/bash
              yum update -y aws-cfn-bootstrap
```

0 0 Ln 17, Col 20 Spaces: 2 UTF-8 LF YAML

Activities Visual Studio Code Tue 3:39 PM

stackq2.yml - Visual Studio Code

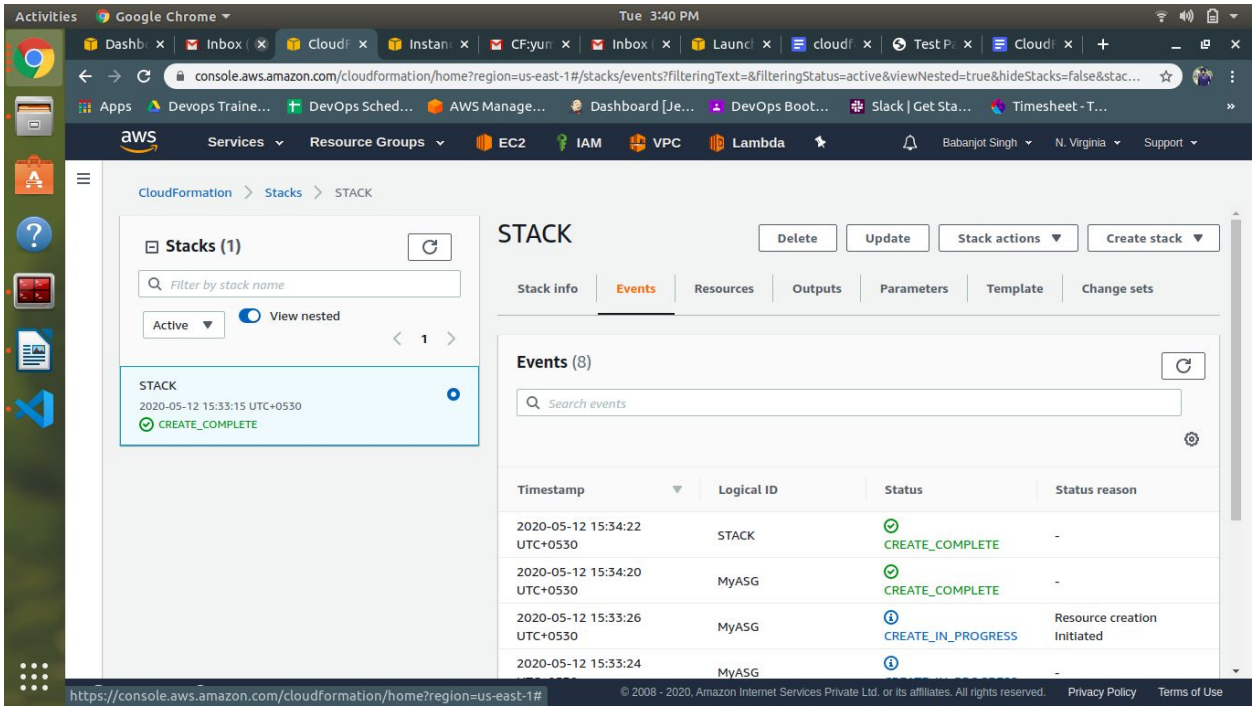
File Edit Selection View Go Run Terminal Help

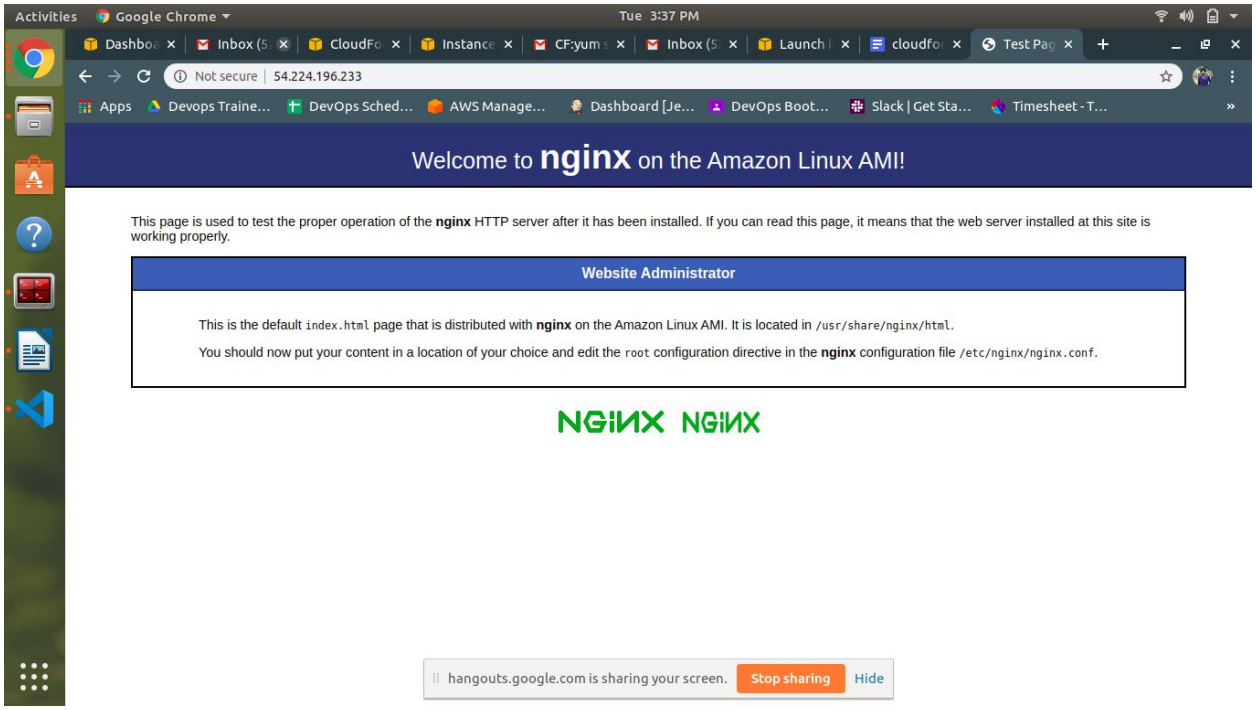
stackq2.yml

home > babanjot > Downloads > stackq2.yml

```yaml
    Properties:
      LaunchTemplateName: "Nginx-Template-MD"
      LaunchTemplateData:
        InstanceType: t2.micro
        KeyName: elk
        ImageId: ami-0915e09cc7ceee3ab
        SecurityGroupIds:
          - sg-0189301dd7c4863a2
        UserData:
          Fn::Base64:
            Fn::Sub: |
              #!/bin/bash
              yum update -y aws-cfn-bootstrap
              service nginx start
              /opt/aws/bin/cfn-init -s ${AWS::StackId} -r NginxLaunchTemplate --region ${AWS::Region} || error_exit 'Failed
              /opt/aws/bin/cfn-hup || error_exit "Failed to start cfn-hup"
              /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackId} --resource NginxLaunchTemplate --region ${AWS::Region}
  MyASG:
    Type: AWS::AutoScaling::AutoScalingGroup
    Properties:
      VPCZoneIdentifier:
        - !Ref MySubnetIDa
        - !Ref MySubnetIDb

      LaunchTemplate:
        LaunchTemplateId:
          Ref: NginxLaunchTemplate
        Version:
          Fn::GetAtt: "NginxLaunchTemplate.LatestVersionNumber"
      MinSize: 1
      MaxSize: 2
```

0 0 Ln 17, Col 20 Spaces: 2 UTF-8 LF YAML





3. Create a Sample Index file and copy this file using MetaData into EC2 Instance

Ans.

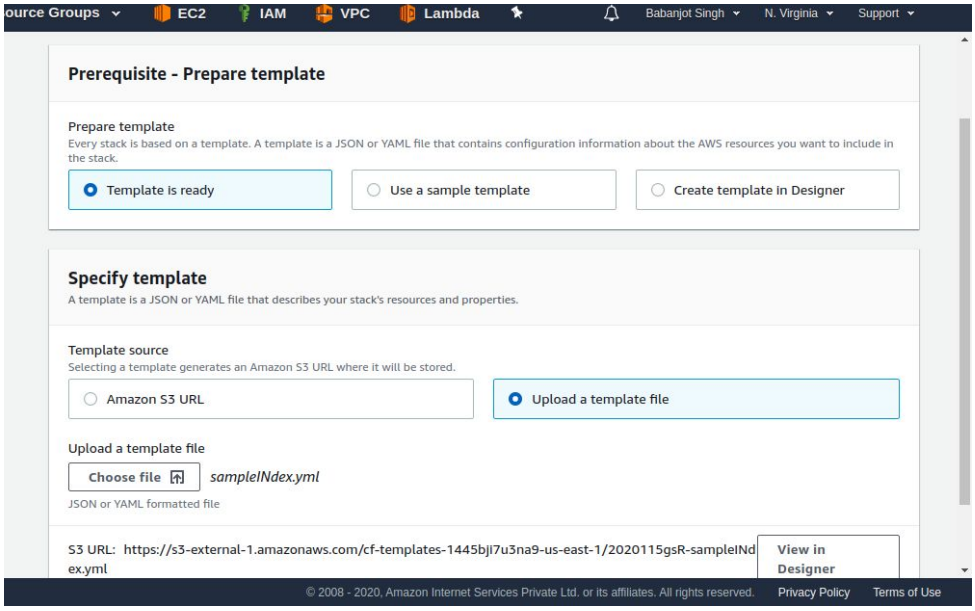
Step 1: Create a cloudformation template

```yaml
Parameters:
  SecurityGroupDescription:
    Description: Security Group for instance
    Type: String
  SecurityGroupPort:
    Description: Mention port
    Type: Number
    MinValue: 0
    MaxValue: 65535
  InstanceType:
    Description: EC2 instance type
    Type: String
    Default: t2.micro
    AllowedValues:
      - t1.micro
      - t2.nano
      - t2.micro
      - t2.small
    ConstraintDescription: Must be a valid EC2 instance type.
  KeyName:
    Description: Name of an existing EC2 KeyPair to enable SSH access to the instances.
    Type: AWS::EC2::KeyPair::KeyName
    ConstraintDescription: must be an existing EC2 KeyPair.
  SecurityGroupIngressCIDR:
    Description: The IP address range that can be used to communicate to the EC2 instances
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 0.0.0.0/0
    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.
  MyVPC:
"~/exercise/cloudformationassignment/sampleINdex.yml" 79L, 2391C                    1,11          Top
```

```yaml
  MyVPC:
    Description: VPC to operate in
    Type: AWS::EC2::VPC::Id
  MySubnetID:
    Description: Subnet IDs that is a List of Subnet Id
    Type: AWS::EC2::Subnet::Id

Resources:
  MyEC2Instance:
    Type: "AWS::EC2::Instance"
    Metadata:
      Comment: Create a file index.txt
      AWS::CloudFormation::Init:
        config:
          files:
            "/tmp/SampleIndex.html":
              content: |
                Copied file from Metadata
                Hello 1.2.3.4.5.6.
              mode: '000400'
              owner: root
              group: root

    Properties:
      InstanceType: !Ref InstanceType
      KeyName: !Ref KeyName
      ImageId: "ami-0323c3dd2da7fb37d"
      SubnetId: !Ref MySubnetID
      SecurityGroupIds:
        - !Ref MySecurityGroup
      UserData:
        "Fn::Base64":
                                                                                  37,11         65%
```

```
            content: |
              Copied file from Metadata
              Hello 1.2.3.4.5.6.
            mode: '000400'
            owner: root
            group: root

    Properties:
      InstanceType: !Ref InstanceType
      KeyName: !Ref KeyName
      ImageId: "ami-0323c3dd2da7fb37d"
      SubnetId: !Ref MySubnetID
      SecurityGroupIds:
        - !Ref MySecurityGroup
      UserData:
        "Fn::Base64":
          !Sub |
            #!/bin/bash -xe
            sudo yum update -y aws-cfn-bootstrap

            /opt/aws/bin/cfn-init -s ${AWS::StackId} -r MyEC2Instance --region ${AWS::Region}

  MySecurityGroup:
    Type: "AWS::EC2::SecurityGroup"
    Properties:
      GroupDescription: !Ref SecurityGroupDescription
      SecurityGroupIngress:
        - CidrIp: !Ref SecurityGroupIngressCIDR
          FromPort: !Ref SecurityGroupPort
          ToPort: !Ref SecurityGroupPort
          IpProtocol: tcp
      VpcId: !Ref MyVPC
                                                                                    79,11         Bot
```

Step 2: Go to Cloudformation → Create Stack



Step 3: Specify Parameters

KeyName
Name of an existing EC2 KeyPair to enable SSH access to the instances.
elk

MySubnetID
Subnet IDs that is a List of Subnet Id
subnet-0de57c46b1cf53ed4 (10.0.2.0/24) (subnet3)

MyVPC
VPC to operate in
vpc-0a3acec5a9e02c022 (10.0.0.0/16) (baban)

SecurityGroupDescription
Security Group for instance
SG_11

SecurityGroupIngressCIDR
The IP address range that can be used to communicate to the EC2 instances
0.0.0.0/0

SecurityGroupPort
Mention port
22

Timeout
-

Termination protection
Disabled

Quick-create link

Cancel Previous Create change set Create stack

View nested

1

pleINdex
)-04-24 15:30:05 UTC+0530
CREATE_COMPLETE

Search events

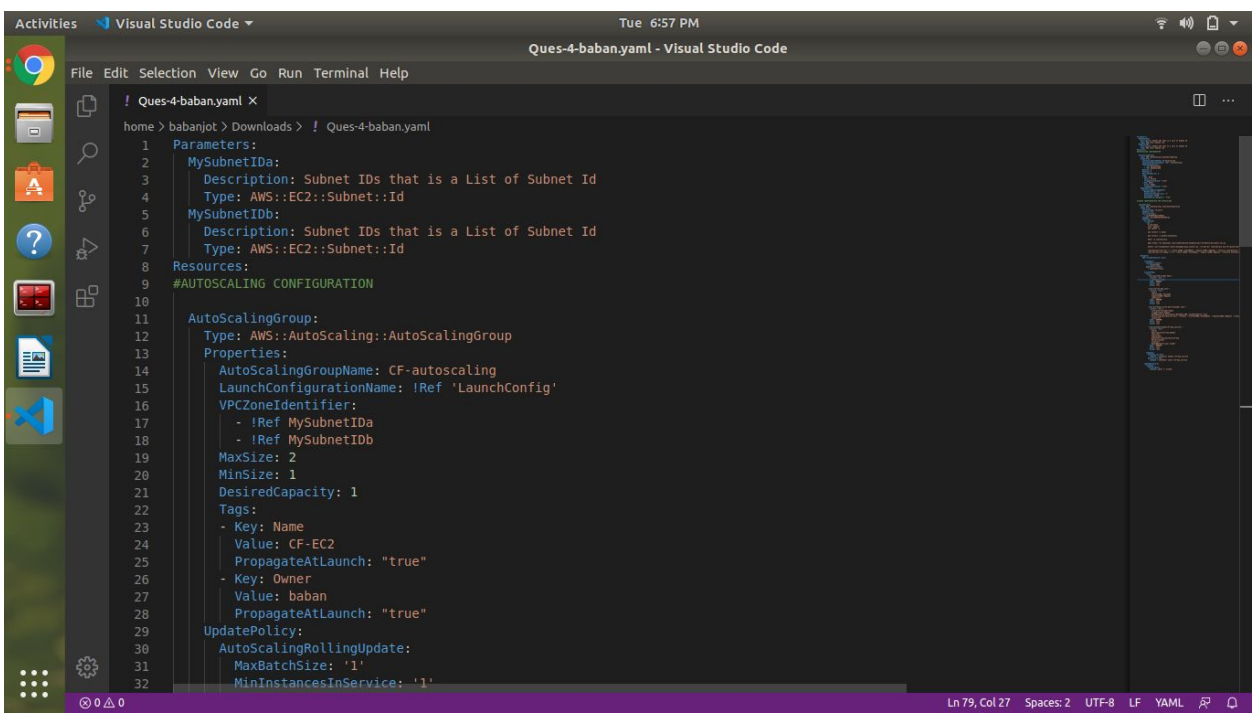
Timestamp	Logical ID	Status	Status reason
2020-04-24 15:30:53 UTC+0530	sampleINdex	CREATE_COMPLETE	-
2020-04-24 15:30:52 UTC+0530	MyEC2Instance	CREATE_COMPLETE	-
2020-04-24 15:30:20 UTC+0530	MyEC2Instance	CREATE_IN_PROGRESS	Resource creation Initiated
2020-04-24 15:30:18 UTC+0530	MyEC2Instance	CREATE_IN_PROGRESS	-
2020-04-24 15:30:16 UTC+0530	MySecurityGroup	CREATE_COMPLETE	-
2020-04-24 15:30:14 UTC+0530	MySecurityGroup	CREATE_IN_PROGRESS	Resource creation Initiated

Now ssh into the ec2 and check for the file

```
babanjot@Babanjot:~/Downloads$ ssh -i "elk.pem" ec2-user@100.25.13.48

The authenticity of host '100.25.13.48 (100.25.13.48)' can't be established.
ECDSA key fingerprint is SHA256:qq5CwpmzccP0GLv5uqXYZMVPQS0eMveJ8yTzUEX3pxk.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '100.25.13.48' (ECDSA) to the list of known hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
No packages needed for security; 4 packages available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-10-0-0-112 ~]$ cd /tmp
[ec2-user@ip-10-0-0-112 tmp]$ ls
SampleIndex.html  systemd-private-4b9c5c8e0a3b49f3a6456c17534d7e5d-chronyd.service-m9VKNY
[ec2-user@ip-10-0-0-112 tmp]$ cat SampleIndex.html
cat: SampleIndex.html: Permission denied
[ec2-user@ip-10-0-0-112 tmp]$ sudo !!
sudo cat SampleIndex.html
Copied file from Metadata
Hello 1.2.3.4.5.6.
[ec2-user@ip-10-0-0-112 tmp]$
```

4. Changing the content of Index should reload the nginx config automatically in EC2 Instance

5. Perform ASG Rolling Update with the change in UserData in above Cloudformation Template



Activities Visual Studio Code Tue 6:58 PM

Ques-4-baban.yaml - Visual Studio Code

File Edit Selection View Go Run Terminal Help

Ques-4-baban.yaml

home > babanjot > Downloads > Ques-4-baban.yaml

```yaml
          WaitOnResourceSignals: 'true'

#LAUNCH COUNFIGURATION FOR AUTOSCALING

  LaunchConfig:
    Type: AWS::AutoScaling::LaunchConfiguration
    Properties:
      InstanceType: t2.micro
      KeyName: elk
      SecurityGroups:
        - sg-0189301dd7c4863a2
      ImageId: ami-085925f297f89fce1
      UserData:
        Fn::Base64:
          !Sub |
            #!/bin/bash
            echo "update"
            apt update -y

            apt install -y nginx

            apt install -y python-setuptools

            mkdir -p /opt/aws/bin

            wget https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.tar.gz

            python /usr/lib/python2.7/dist-packages/easy_install.py --script-dir /opt/aws/bin aws-cfn-bootstrap-latest.tar.g

            /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --region ${AWS::Region} --resource LaunchConfig --configsets
            /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --region ${AWS::Region} --resource AutoScalingGroup

```

⊗ 0 ⚠ 0 Ln 79, Col 27 Spaces: 2 UTF-8 LF YAML

Activities Visual Studio Code Tue 6:58 PM

Ques-4-baban.yaml - Visual Studio Code

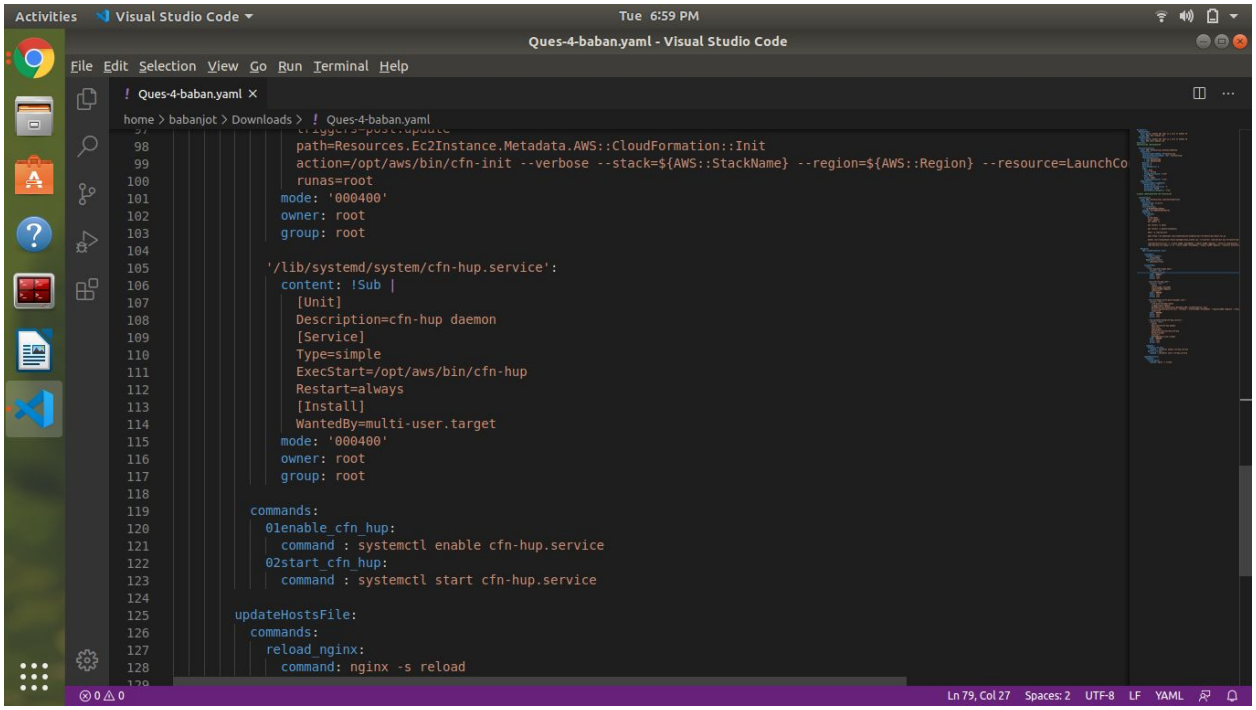
File Edit Selection View Go Run Terminal Help

Ques-4-baban.yaml

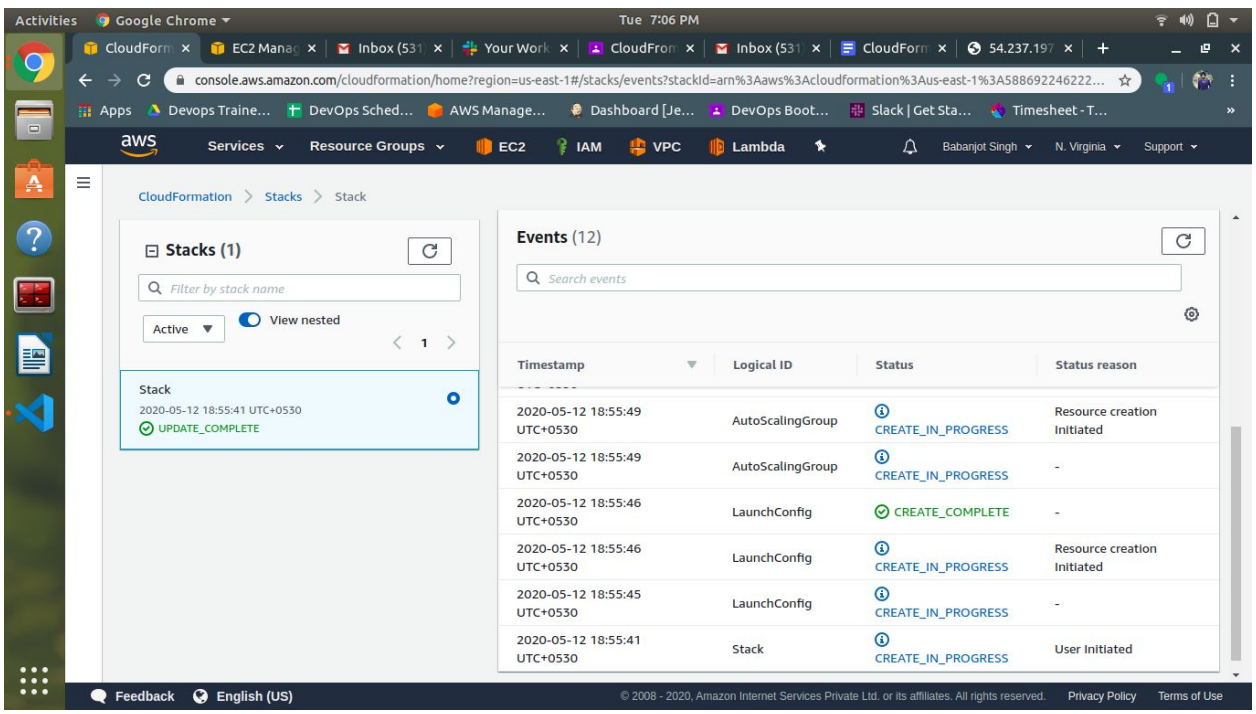
home > babanjot > Downloads > Ques-4-baban.yaml

```yaml
    Metadata:
      AWS::CloudFormation::Init:

        configSets:
          SetupEnvironment:
            - setupCfnHup
          UpdateEnvironment:
            - updateHostsFile

        setupCfnHup:
          files:
            '/var/www/html/index.html':
              content: !Sub  |
                <h1>Hello </h1>
              mode: '000644'
              owner: root
              group: root

            '/etc/cfn/cfn-hup.conf':
              content: !Sub |
                [main]
                stack=${AWS::StackId}
                region=${AWS::Region}
                interval=1
              mode: '000400'
              owner: root
              group: root

            '/etc/cfn/hooks.d/cfn-auto-reloader.conf':
              content: !Sub |
                [cfn-auto-reloader-hook]
                triggers=post.update
```

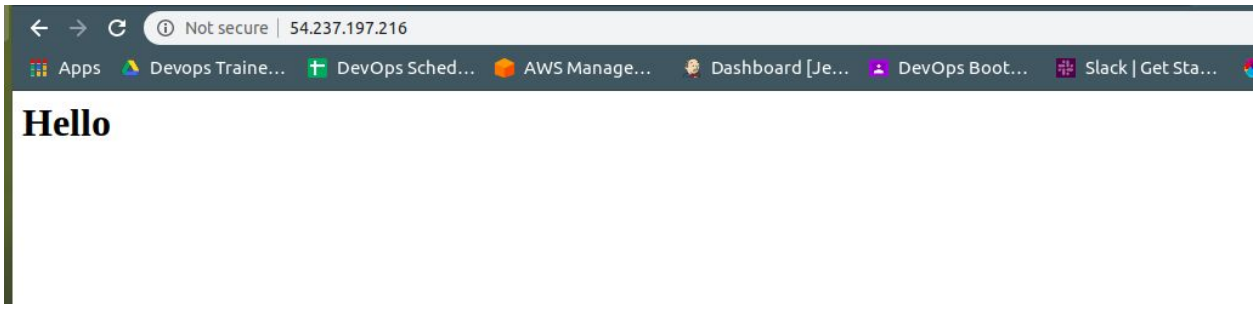
⊗ 0 ⚠ 0 Ln 79, Col 27 Spaces: 2 UTF-8 LF YAML



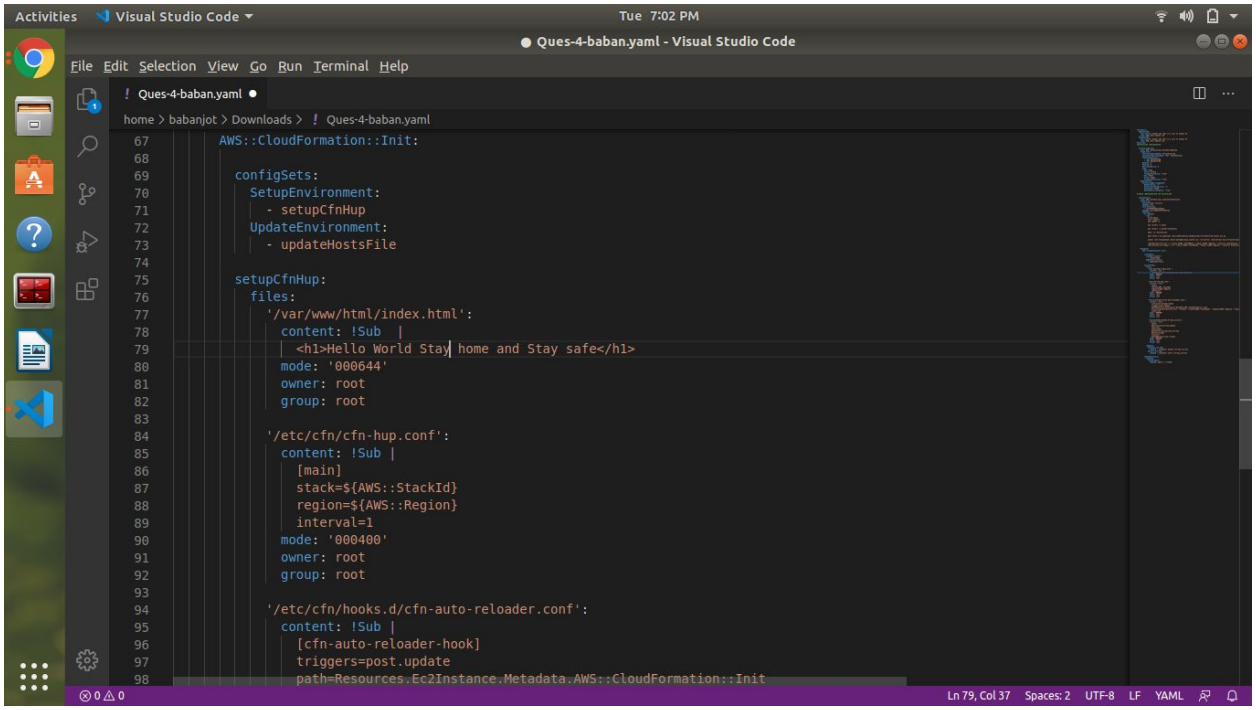
Create the stack



After stack is created



Now apply changes in template



Update the stack