

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 > babanot > exercise > cloudformationassignment > ! Parameters.yml
1 Parameters:
2   InstanceType:
3     Description: WebServer EC2 instance type (has default, AllowedValues)
4     Type: String
5     Default: t2.micro
6     AllowedValues:
7       - t1.micro
8       - t2.nano
9       - t2.micro
10      - t2.small
11     ConstraintDescription: must be a valid EC2 instance type.
12   KeyName:
13     Description: Name of an existing EC2 KeyPair to enable SSH access to the instances. Linked to AWS Parameter
14     Type: AWS::EC2::KeyPair::KeyName
15     ConstraintDescription: must be the name of an existing EC2 KeyPair.
16   MyVPC:
17     Description: VPC to operate in
18     Type: AWS::EC2::VPC::Id
19   MySubnetIDs:
20     Description: Subnet IDs that is a List of Subnet Id
21     Type: "List<AWS::EC2::Subnet::Id>"
22   SubnetIpBlocks:
23     Description: "Comma-delimited list of three CIDR blocks"
24     Type: CommaDelimitedList
25     Default: "10.0.48.0/24, 10.0.112.0/24,"
26
27 Resources:
28   MyEC2Instance:
29     Type: "AWS::EC2::Instance"
30     Properties:
31       #we reference the InstanceType parameter
32       InstanceType: !Ref InstanceType
```

Define the resources

```
25     Default: "10.0.48.0/24, 10.0.112.0/24,"
26
27 Resources:
28   MyEC2Instance:
29     Type: "AWS::EC2::Instance"
30     Properties:
31       #we reference the InstanceType parameter
32       InstanceType: !Ref InstanceType
33       KeyName: !Ref KeyName
34       ImageId: "ami-a4c7edb2"
35       # here we reference an internal CloudFormation resource
36       SubnetId: !Ref Subnet1
37   Subnet1:
38     Type: AWS::EC2::Subnet
39     Properties:
40       VpcId: !Ref MyVPC
41       # the select function allows us to select across a list
42       CidrBlock: !Select [0, !Ref SubnetIpBlocks]
43   Subnet2:
44     Type: AWS::EC2::Subnet
45     Properties:
46       VpcId: !Ref MyVPC
47       # the select function allows us to select across a list
48       CidrBlock: !Select [1, !Ref SubnetIpBlocks]
49
```

Go to cloudformation and create a stack and upload your yml file.

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 want to include in the stack.

☒ Template is ready ☐ Use a sample template ☐ Create template in Designer

Specify template

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
 No file chosen

Now Enter all the values

InstanceType
WebServer EC2 instance type (has default, AllowedValues)
t2.micro

KeyName
Name of an existing EC2 KeyPair to enable SSH access to the instances. Linked to AWS Parameter

MySubnetIDs
Subnet IDs that is a List of Subnet Id

MyVPC
VPC to operate in

SubnetipBlocks
Comma-delimited list of three CIDR blocks
10.0.48.0/24, 10.0.112.0/24,

Choose the correct configuration.

InstanceType
WebServer EC2 instance type (has default, AllowedValues)
t2.micro

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

MySubnetIDs
Subnet IDs that is a List of Subnet Id
subnet-0de86a035a9b6102b (10.0.1.0/24) (subnet-2) X

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

SubnetipBlocks
Comma-delimited list of three CIDR blocks
10.0.48.0/24, 10.0.112.0/24,

© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

Now stack is up.

<div>parameters</div> <div>2020-04-23 13:40:24 UTC+0530</div> <div>CREATE_COMPLETE</div>		Events (11)	
		<input type="text" value="Search events"/>	
Timestamp	Logical ID	Status	Status reason
2020-04-23 13:41:23 UTC+0530	parameters	CREATE_COMPLETE	-
2020-04-23 13:41:21 UTC+0530	MyEC2Instance	CREATE_COMPLETE	-

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

i) First by Userdata

Create yml file

```
home > babanjot > exercise > cloudformationassignment > ! nginxUD.yml
34 Resources:
35   MySecurityGroup:
36     Type: "AWS::EC2::SecurityGroup"
37     Properties:
38       GroupDescription: Security Group for instance
39       SecurityGroupIngress:
40         - IpProtocol: 'tcp'
41           FromPort: '80'
42           ToPort: '80'
43           CidrIp: !Ref SecurityGroupIngressCIDR
44         - IpProtocol: 'tcp'
45           FromPort: '443'
46           ToPort: '443'
47           CidrIp: !Ref SecurityGroupIngressCIDR
48       VpcId: !Ref MyVPC
49
50
51   DeployAppLaunchConfig:
52     Type: AWS::AutoScaling::LaunchConfiguration
53     Properties:
54       ImageId: ami-07ebfd5b3428b6f4d
55       InstanceType: !Ref InstanceType
56       KeyName: !Ref KeyName
57       SecurityGroups:
58         - !Ref MySecurityGroup
59       UserData:
60         Fn::Base64:
61           Fn::Sub: |
62             #!/bin/bash
63             sudo apt-get update -y
64             sudo apt-get install nginx -y
65             sudo systemctl enable nginx
```

Write resource block for ASG and also userdata

```

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

```

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.yaml

```

67
68 DeployAppASG:
69   Type: AWS::AutoScaling::AutoScalingGroup
70   Properties:
71     VPCZoneIdentifier:
72       - !Ref MySubnetIDa
73       - !Ref MySubnetIDb
74     DesiredCapacity: 1
75     LaunchConfigurationName: !Ref DeployAppLaunchConfig
76     MaxSize: 2
77     MinSize: 1
78     Tags:
79       - Key: Name
80         Value: Babanjot-ASG
81       PropagateAtLaunch: True
82   UpdatePolicy:
83     AutoScalingReplacingUpdate:
84       WillReplace: True

```

Now upload the yml file and create stack in CCloud 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 want to create in the stack.

☒ Template is ready

☐ Use a sample template

☐ Create template in Designer

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-1445bjl7u3na9-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

Stacks (1)

Filter by stack name

Active View nested

NglnxinASG
2020-04-23 15:47:53 UTC+0530
CREATE_COMPLETE

Stack info Events Resources Outputs Parameters Template Change sets

Events (11)

Search events

Timestamp	Logical ID	Status	Status reason
2020-04-23 15:49:36 UTC+0530	NglnxinASG	CREATE_COMPLETE	-
2020-04-23 15:49:34 UTC+0530	DeployAppASG	CREATE_COMPLETE	-

Check your infrastructure.

Filter by tags and attributes or search by keyword

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
Babanjot-ASG	i-0ad21726ee876b7...	t2.micro	us-east-1e	running	2/2 checks passed

Create launch configuration Create Auto Scaling group Copy to launch template Actions

Filter: Filter launch configurations...

Name	AMI ID	Instance Type	Spot Price	Creation Time
NglnxinASG-D...	ami-07ebfd5b3...	t2.micro		April 23, 2020 at 3:48:06 PM UT...

Name	Launch Configuration	Instances	Desired	Min	Max	Availability Zones	Default Cooldown
------	----------------------	-----------	---------	-----	-----	--------------------	------------------

Auto Scaling Group: NglnxinASG-DeployAppASG-1SQKCIGR2CD1W

Details Activity History Scaling Policies Instances Monitoring Notifications Tags Scheduled Actions Lifecycle Hooks

Launch Configuration NglnxinASG-DeployAppLaunchConfig-1FV23D68S52B7

Availability Zone(s) us-east-1e

Subnet(s) subnet-0b...

ii) By Metadata

Activities Visual Studio Code Tue 3:39 PM
stackq2.yml - Visual Studio Code

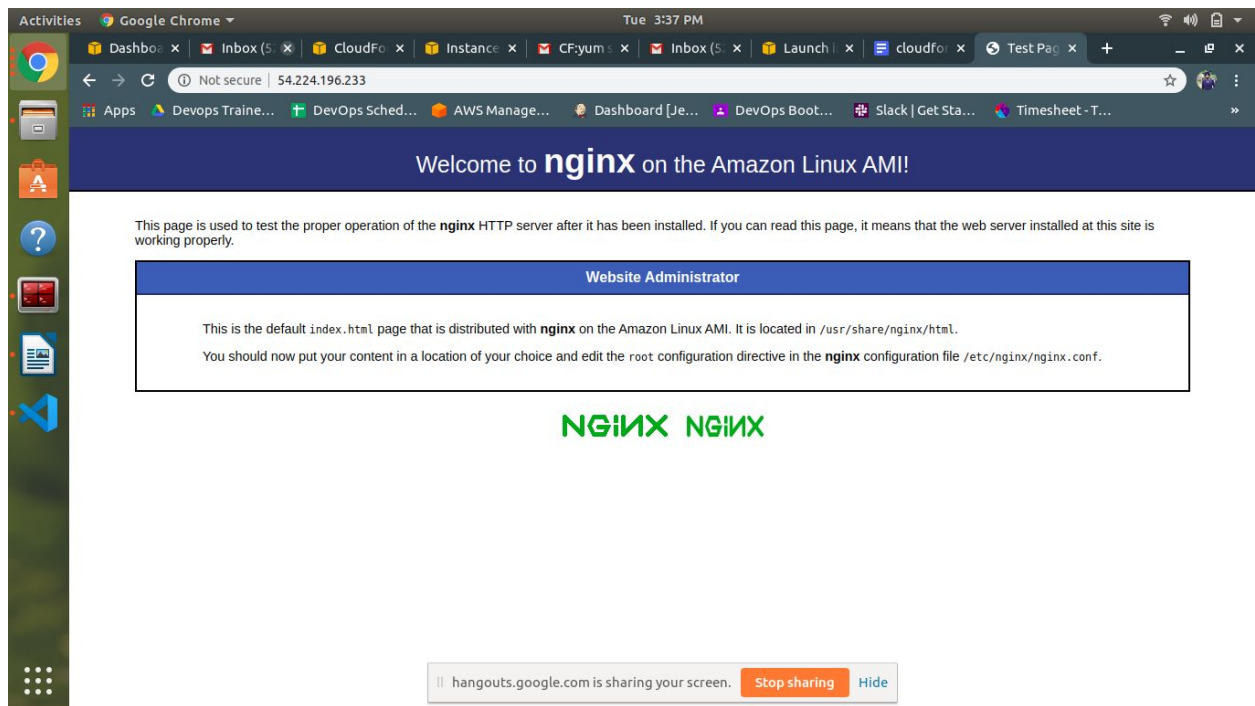
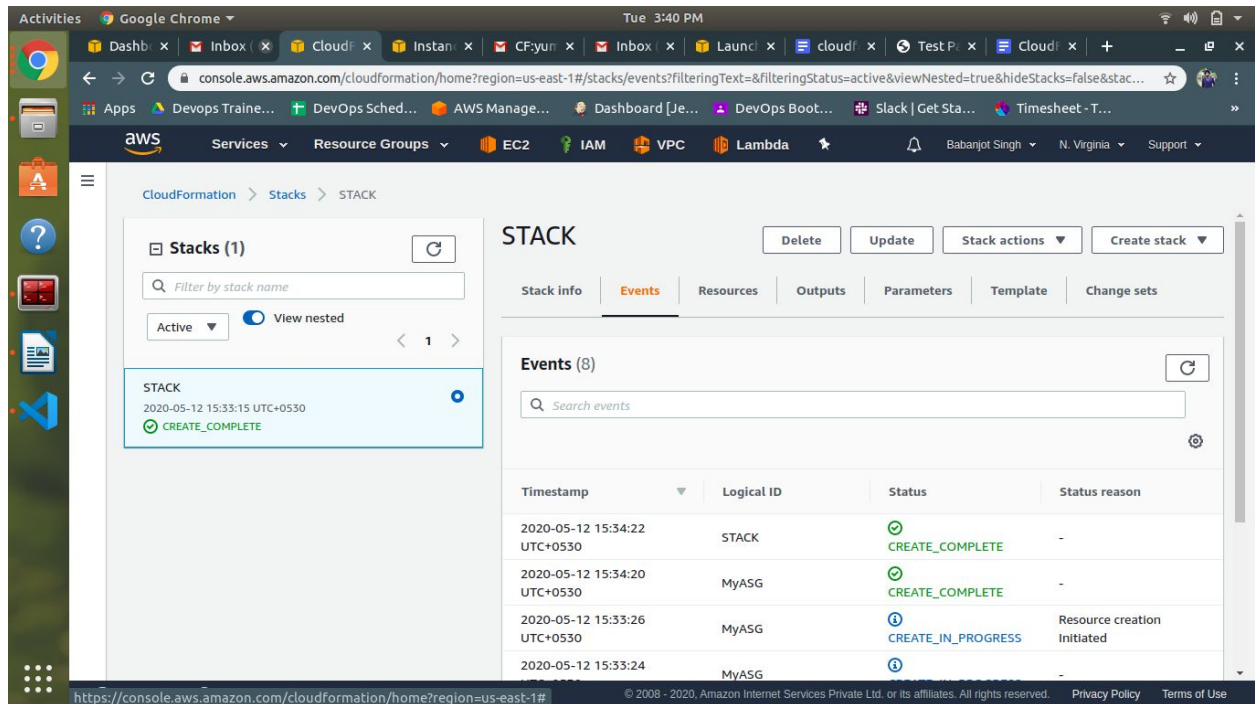
```
1 Parameters:
2   MySubnetIDa:
3     Description: Subnet IDs that is a List of Subnet Id
4     Type: AWS::EC2::Subnet::Id
5   MySubnetIDb:
6     Description: Subnet IDs that is a List of Subnet Id
7     Type: AWS::EC2::Subnet::Id
8
9
10  Resources:
11
12    NginxLaunchTemplate:
13      Type: AWS::EC2::LaunchTemplate
14      Metadata:
15        AWS::CloudFormation::Init:
16          config:
17            packages:
18              yum:
19                nginx: []
20      Properties:
21        LaunchTemplateName: "Nginx-Template-MD"
22        LaunchTemplateData:
23          InstanceType: t2.micro
24          KeyName: elk
25          ImageId: ami-0915e09cc7ceee3ab
26          SecurityGroupIds:
27            - sg-0189301dd7c4863a2
28          UserData:
29            Fn::Base64:
30              Fn::Sub: |
31                #!/bin/bash
32                yum update -y aws-cfn-bootstrap
```

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

Activities Visual Studio Code Tue 3:39 PM
stackq2.yml - Visual Studio Code

```
20 Properties:
21   LaunchTemplateName: "Nginx-Template-MD"
22   LaunchTemplateData:
23     InstanceType: t2.micro
24     KeyName: elk
25     ImageId: ami-0915e09cc7ceee3ab
26     SecurityGroupIds:
27       - sg-0189301dd7c4863a2
28     UserData:
29       Fn::Base64:
30         Fn::Sub: |
31           #!/bin/bash
32           yum update -y aws-cfn-bootstrap
33           service nginx start
34           /opt/aws/bin/cfn-init -s ${AWS::StackId} -r NginxLaunchTemplate --region ${AWS::Region} || error_exit 'Failed
35           /opt/aws/bin/cfn-hup || error_exit "Failed to start cfn-hup"
36           /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackId} --resource NginxLaunchTemplate --region ${AWS::Region}
37   MyASG:
38     Type: AWS::AutoScaling::AutoScalingGroup
39     Properties:
40       VPCZoneIdentifier:
41         - !Ref MySubnetIDa
42         - !Ref MySubnetIDb
43       LaunchTemplate:
44         LaunchTemplateId:
45           Ref: NginxLaunchTemplate
46         Version:
47           Fn::GetAtt: "NginxLaunchTemplate.LatestVersionNumber"
48       MinSize: 1
49       MaxSize: 2
```

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


```
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:
    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":
          ~~~~~
```

~/exercise/cloudformationassignment/sampleIndex.yml 79L, 2391C 1,11 Top

```
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

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

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 want to include in the stack.

☒ Template is ready ☐ Use a sample template ☐ Create template in Designer

Specify template

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

JSON or YAML formatted file

S3 URL: <https://s3-external-1.amazonaws.com/cf-templates-1445bjl7u3na9-us-east-1/2020115gsR-sampleIndex.yml>

© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

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

© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

Timeout
-

Termination protection
Disabled

► Quick-create link

Cancel Previous Create change set **Create stack**

active View nested

pleIndex
2020-04-24 15:30:05 UTC+0530
CREATE_COMPLETE

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

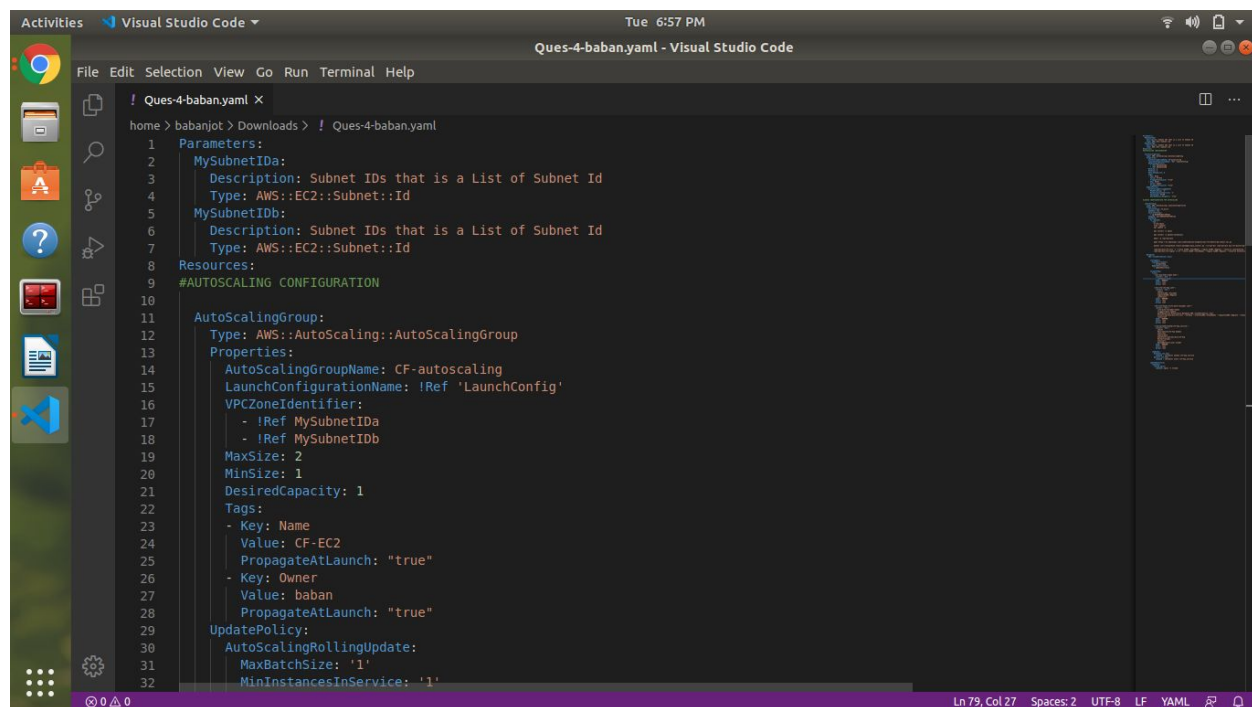
```
babanjot@Babanjot:~/Downloads$ ssh -i "elk.pem" ec2-user@100.25.13.48
The authenticity of host '100.25.13.48 (100.25.13.48)' can't be established.
ECDSA key fingerprint is SHA256:qq5CwpmzccP0GLv5uqXYZMVPQS0eMveJ8yTzUEX3pxk.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently 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-chrond.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



```
Ques-4-baban.yaml
1 Parameters:
2   MySubnetIDa:
3     Description: Subnet IDs that is a List of Subnet Id
4     Type: AWS::EC2::Subnet::Id
5   MySubnetIDb:
6     Description: Subnet IDs that is a List of Subnet Id
7     Type: AWS::EC2::Subnet::Id
8 Resources:
9   #AUTOSCALING CONFIGURATION
10
11   AutoScalingGroup:
12     Type: AWS::AutoScaling::AutoScalingGroup
13     Properties:
14       AutoScalingGroupName: CF-autoscaling
15       LaunchConfigurationName: !Ref 'LaunchConfig'
16       VPCZoneIdentifier:
17         - !Ref MySubnetIDa
18         - !Ref MySubnetIDb
19       MaxSize: 2
20       MinSize: 1
21       DesiredCapacity: 1
22       Tags:
23         - Key: Name
24           Value: CF-EC2
25         - Key: Owner
26           Value: baban
27         - Key: PropagateAtLaunch: "true"
28         - Key: PropagateAtLaunch: "true"
29       UpdatePolicy:
30         AutoScalingRollingUpdate:
31           MaxBatchSize: '1'
32           MinInstancesInService: '1'
```

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 X
home > babanjot > Downloads > ! Ques-4-baban.yaml
34     WaitOnResourceSignals: 'true'
35
36 #LAUNCH COUNFIGURATION FOR AUTOSCALING
37
38 LaunchConfig:
39   Type: AWS::AutoScaling::LaunchConfiguration
40   Properties:
41     InstanceType: t2.micro
42     KeyName: elk
43     SecurityGroups:
44       - sg-0189301dd7c4863a2
45     ImageId: ami-085925f297f89fce1
46     UserData:
47       Fn::Base64:
48         !Sub |
49           #!/bin/bash
50           echo "update"
51           apt update -y
52
53           apt install -y nginx
54
55           apt install -y python-setuptools
56
57           mkdir -p /opt/aws/bin
58
59           wget https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.tar.gz
60
61           python /usr/lib/python2.7/dist-packages/easy_install.py --script-dir /opt/aws/bin aws-cfn-bootstrap-latest.tar.gz
62
63           /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --region ${AWS::Region} --resource LaunchConfig --configsets
64           /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --region ${AWS::Region} --resource AutoScalingGroup
65
```

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 X
home > babanjot > Downloads > ! Ques-4-baban.yaml
66     Metadata:
67       AWS::CloudFormation::Init:
68
69       configSets:
70         SetupEnvironment:
71           - setupCfnHup
72         UpdateEnvironment:
73           - updateHostsFile
74
75       setupCfnHup:
76         files:
77           '/var/www/html/index.html':
78             content: !Sub |
79               <h1>Hello </h1>
80             mode: '000644'
81             owner: root
82             group: root
83
84           '/etc/cfn/cfn-hup.conf':
85             content: !Sub |
86               [main]
87               stack=${AWS::StackId}
88               region=${AWS::Region}
89               interval=1
90             mode: '000400'
91             owner: root
92             group: root
93
94           '/etc/cfn/hooks.d/cfn-auto-reloader.conf':
95             content: !Sub |
96               [cfn-auto-reloader-hook]
97               triggers=post.update
98               path=Resource::ElasticLoadBalancing::LoadBalancer::Metadata::AWS::CloudFormation::Init

```

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


```
Ques-4-baban.yaml
home > babanjot > Downloads > ! Ques-4-baban.yaml
199:  - post_update:
200:    path=Resources.EC2Instance.Metadata.AWS::CloudFormation::Init
201:    action=/opt/aws/bin/cfn-init --verbose --stack=${AWS::StackName} --region=${AWS::Region} --resource=LaunchCo
202:    runas=root
203:    mode: '000400'
204:    owner: root
205:    group: root
206:
207:  '/lib/systemd/system/cfn-hup.service':
208:    content: !Sub |
209:      [Unit]
210:      Description=cfn-hup daemon
211:      [Service]
212:      Type=simple
213:      ExecStart=/opt/aws/bin/cfn-hup
214:      Restart=always
215:      [Install]
216:      WantedBy=multi-user.target
217:    mode: '000400'
218:    owner: root
219:    group: root
220:
221:  commands:
222:    01enable_cfn_hup:
223:      command : systemctl enable cfn-hup.service
224:    02start_cfn_hup:
225:      command : systemctl start cfn-hup.service
226:
227:  updateHostsFile:
228:    commands:
229:      reload_nginx:
230:        command: nginx -s reload
```

Create the stack

CloudFormation > Stacks > Stack

Stacks (1)

Filter by stack name

Active View nested

Stack

2020-05-12 18:55:41 UTC+0530

UPDATE_COMPLETE

Events (12)

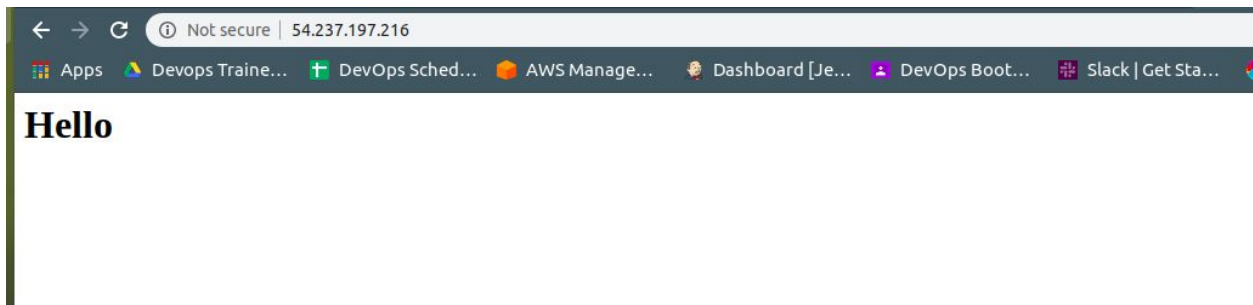
Search events

Timestamp	Logical ID	Status	Status reason
2020-05-12 18:55:49 UTC+0530	AutoScalingGroup	CREATE_IN_PROGRESS	Resource creation Initiated
2020-05-12 18:55:49 UTC+0530	AutoScalingGroup	CREATE_IN_PROGRESS	-
2020-05-12 18:55:46 UTC+0530	LaunchConfig	CREATE_COMPLETE	-
2020-05-12 18:55:46 UTC+0530	LaunchConfig	CREATE_IN_PROGRESS	Resource creation Initiated
2020-05-12 18:55:45 UTC+0530	LaunchConfig	CREATE_IN_PROGRESS	-
2020-05-12 18:55:41 UTC+0530	Stack	CREATE_IN_PROGRESS	User Initiated

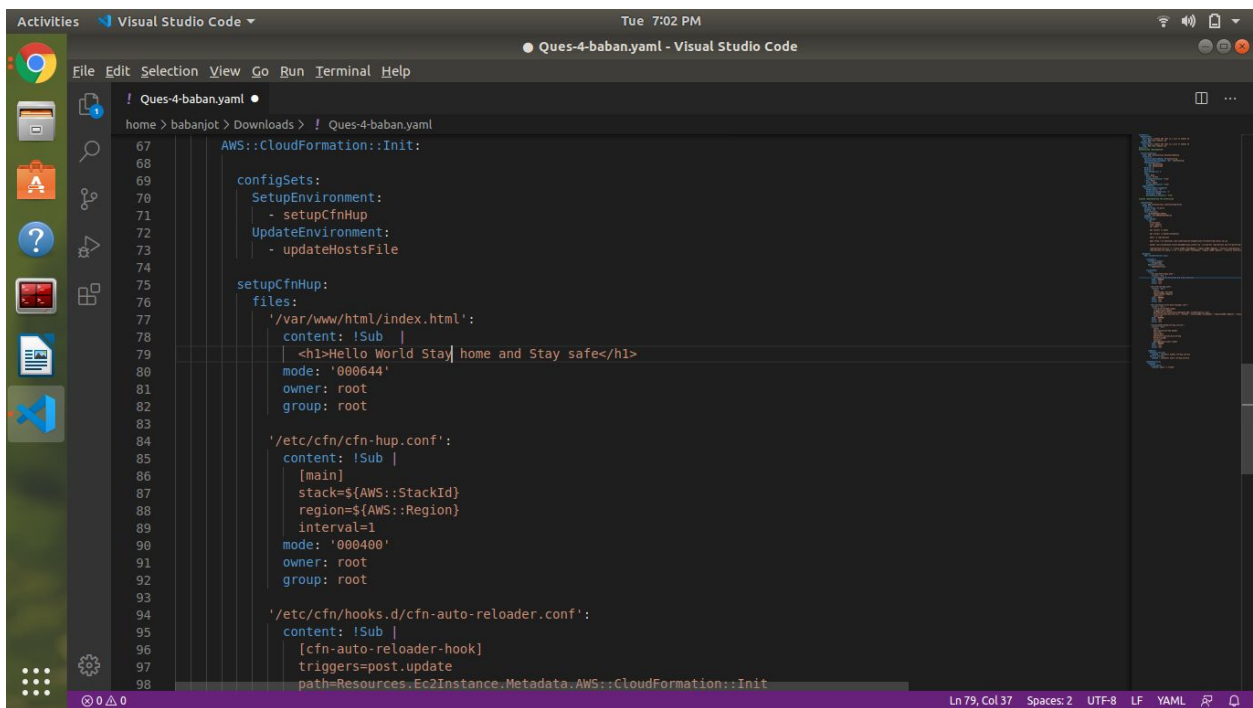
Feedback English (US)

© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use

After stack is created



Now apply changes in template



Update the stack

The screenshot shows the AWS CloudFormation console in Google Chrome. The browser tabs include 'CloudForm', 'Instances', 'Inbox (531)', 'Your Work', 'CloudForm', 'Inbox (531)', 'CloudForm', '\$4,237.197', and '+'. The address bar shows the URL: `console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/update/template?stackId=arn%3Aaws%3Acloudformation%3Aus-east-1%3A5886...`. The AWS navigation bar at the top lists services like EC2, IAM, VPC, and Lambda. The left sidebar shows the 'Specify stack details' section with steps: Step 3 (Configure stack options), Step 4 (Review), and a 'Specify template details' section.

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 want to include in the stack.

☐ Use current template ☒ Replace current template ☐ Edit template in designer

Specify template
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.

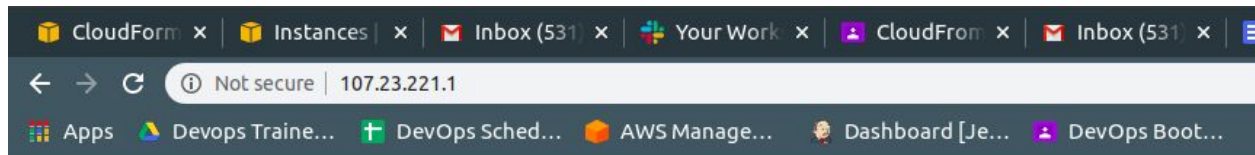
☐ Amazon S3 URL ☒ Upload a template file

Upload a template file
Choose file `Ques-4-baban.yaml`
JSON or YAML formatted file

S3 URL: `https://s3-external-1.amazonaws.com/cf-templates-1445bj7u3na9-us-east-1/2020133QJH-Ques-4-baban.yaml` [View in Designer](#)

Cancel **Next**

© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)



Hello World Stay home and Stay safe