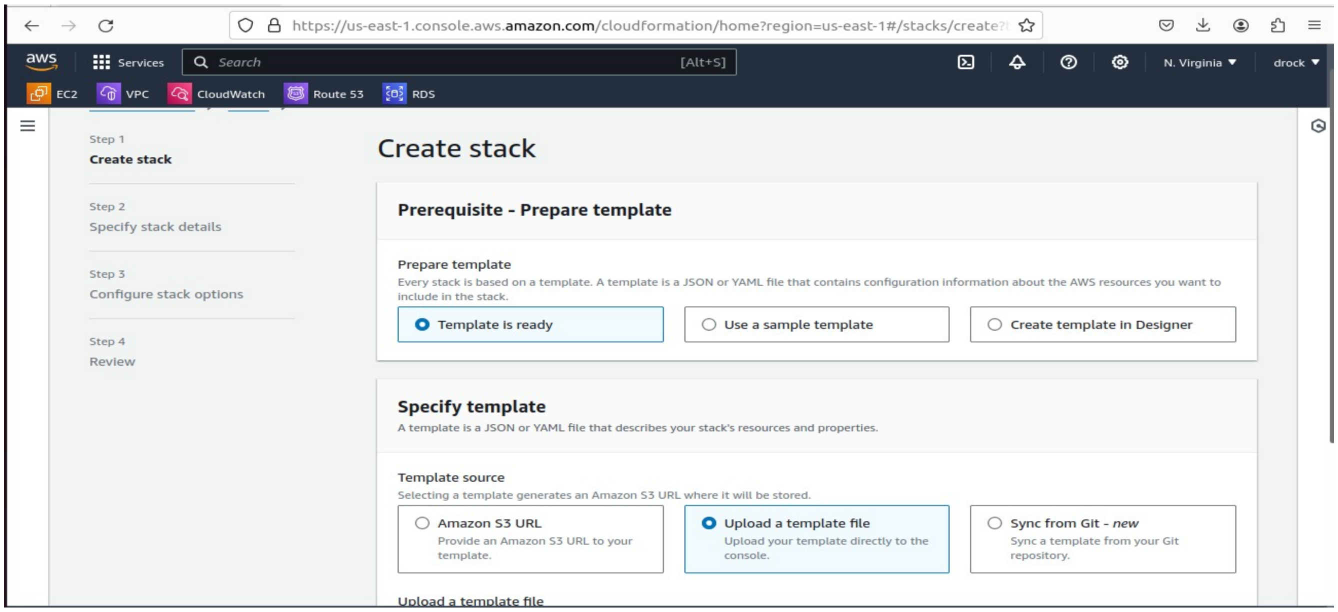
1.

I don't have actual values for the DNS parameters (HostedZoneId and Name) in the "MyDNSRecordSet" resource, it is causing an error in CloudFormation.

So, I removed "MyDNSRecordSet" resource from the template

STACK CREATION



https://us-east-1.console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/create?

Step 1
Create stack

Step 2
Specify stack details

Step 3
Configure stack options

Step 4
Review

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

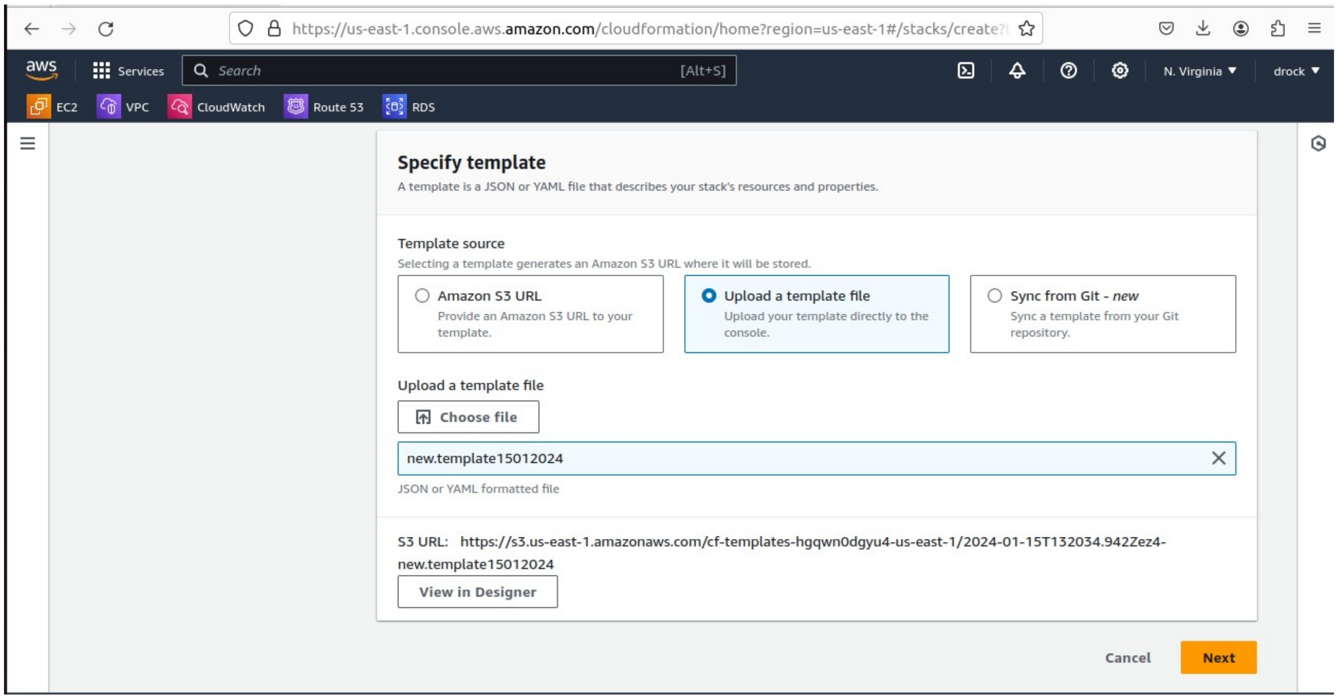
Specify template
A template is a JSON or YAML file that describes your stack's resources and properties.

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

Amazon S3 URL — Provide an Amazon S3 URL to your template.
[✓ Upload a template file — Upload your template directly to the console.]
Sync from Git - *new* — Sync a template from your Git repository.

Upload a template file

PICK DESIGNER TEMPLATE CREATED AND SAVED IN LOCAL FURTHER DETAILS BELOW



https://us-east-1.console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/create?

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

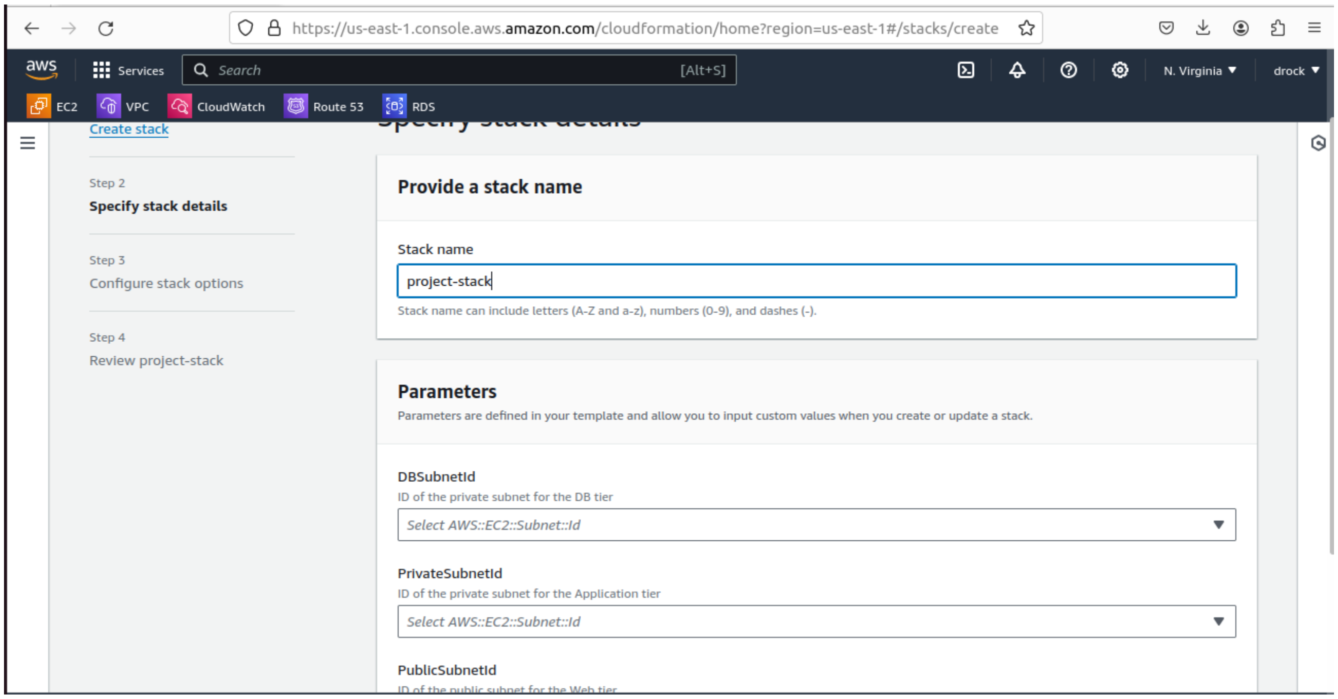
Template source
Selecting a template generates an Amazon S3 URL where it will be stored.

Amazon S3 URL — Provide an Amazon S3 URL to your template.
[✓ Upload a template file — Upload your template directly to the console.]
Sync from Git - *new* — Sync a template from your Git repository.

Upload a template file
Choose file
new.template15012024
JSON or YAML formatted file

S3 URL: https://s3.us-east-1.amazonaws.com/cf-templates-hgqwn0dgyu4-us-east-1/2024-01-15T132034.942Zez4-new.template15012024

CLICK NEXT AND NAME THE STACK


https://us-east-1.console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/create

Create stack

Step 2
Specify stack details

Step 3
Configure stack options

Step 4
Review project-stack

Provide a stack name

Stack name

project-stack

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

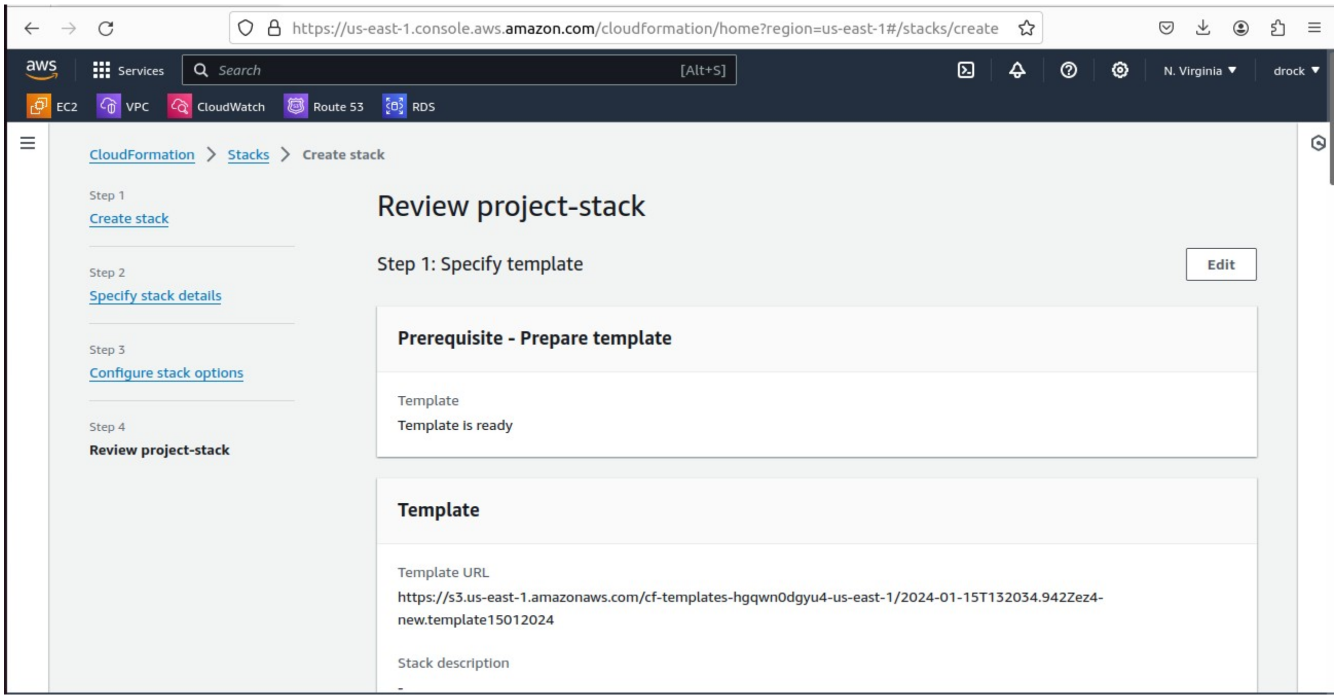
Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

DBSubnetId
ID of the private subnet for the DB tier
Select AWS::EC2::Subnet::Id

PrivateSubnetId
ID of the private subnet for the Application tier
Select AWS::EC2::Subnet::Id

PublicSubnetId


https://us-east-1.console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/create

CloudFormation > Stacks > Create stack

Step 1
Create stack

Step 2
Specify stack details

Step 3
Configure stack options

Step 4
Review project-stack

Review project-stack

Step 1: Specify template

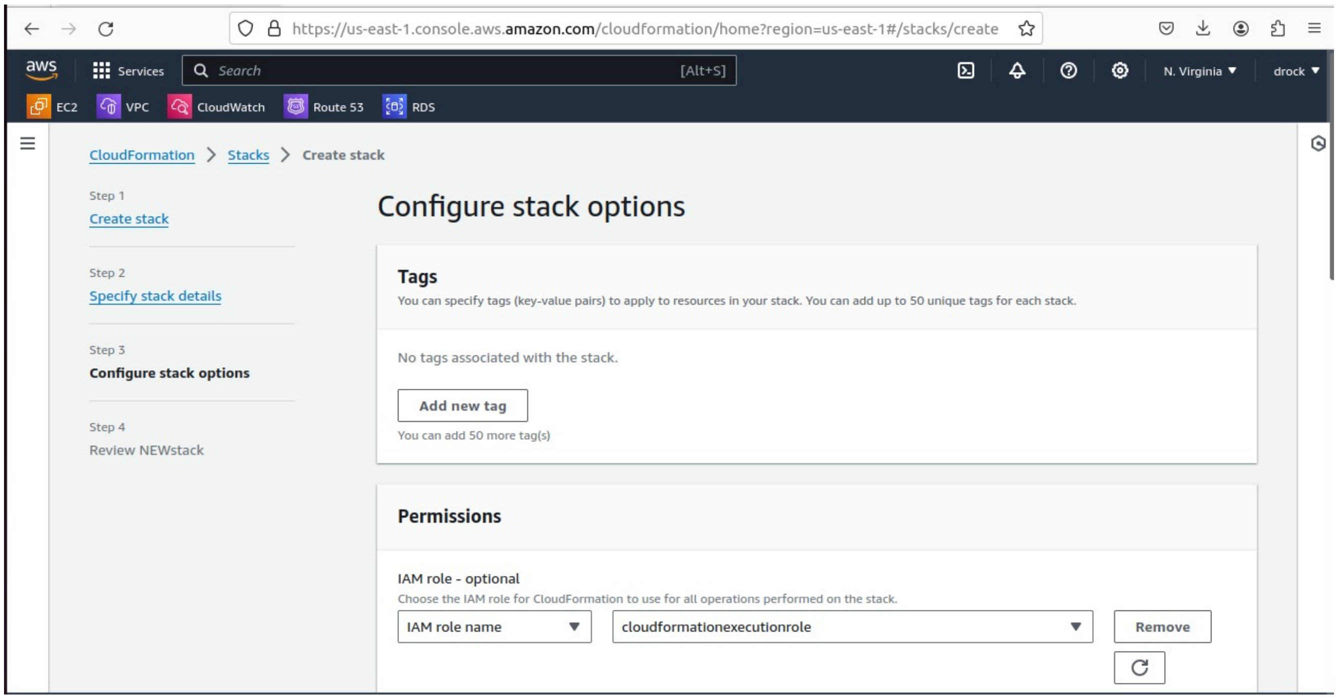
Prerequisite - Prepare template

Template
Template is ready

Template

Template URL
https://s3.us-east-1.amazonaws.com/cf-templates-hgqwn0dgyu4-us-east-1/2024-01-15T132034.942Zez4-new.template15012024

Stack description
-


https://us-east-1.console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/create

CloudFormation > Stacks > Create stack

Step 1
Create stack

Step 2
Specify stack details

Step 3
Configure stack options

Step 4
Review NEWstack

Configure stack options

Tags

You can specify tags (key-value pairs) to apply to resources in your stack. You can add up to 50 unique tags for each stack.

No tags associated with the stack.

Add new tag

You can add 50 more tag(s)

Permissions

IAM role - optional

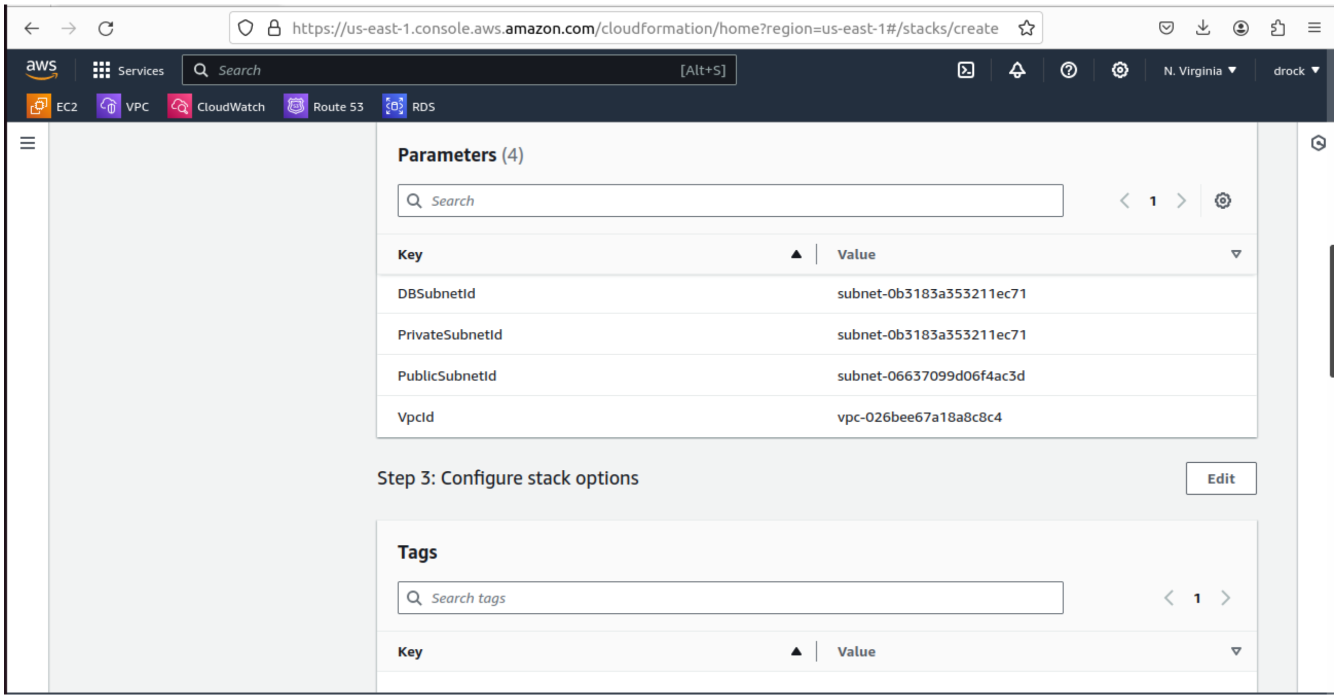
Choose the IAM role for CloudFormation to use for all operations performed on the stack.

IAM role name: cloudformationexecutionrole

Remove

IAM role was created for the resource, with adequate permission.

CHOSEN PARAMETERS


https://us-east-1.console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/create

Parameters (4)

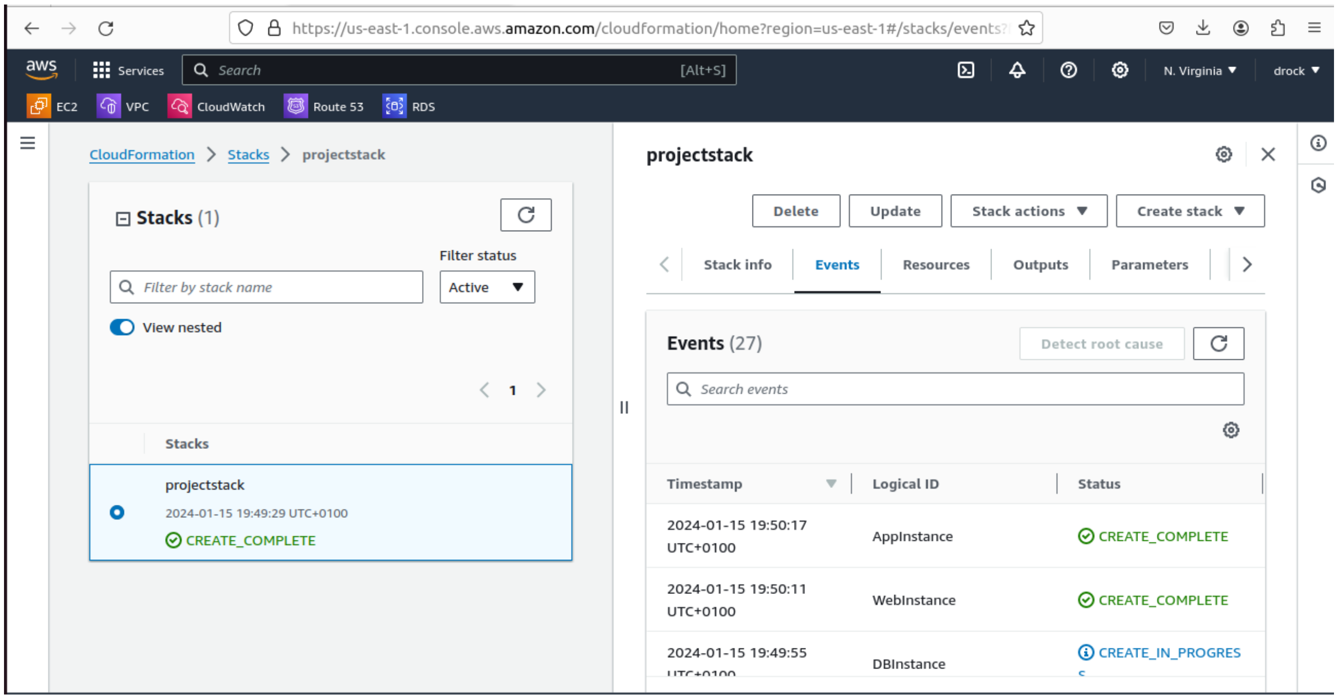
Key	Value
DBSubnetId	subnet-0b3183a353211ec71
PrivateSubnetId	subnet-0b3183a353211ec71
PublicSubnetId	subnet-06637099d06f4ac3d
VpcId	vpc-026bee67a18a8c8c4

Step 3: Configure stack options

Edit

Tags

Key	Value


https://us-east-1.console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/events?

CloudFormation > Stacks > projectstack

Stacks (1)

Filter status: Active

View nested

	Stacks
●	projectstack 2024-01-15 19:49:29 UTC+0100 CREATE_COMPLETE

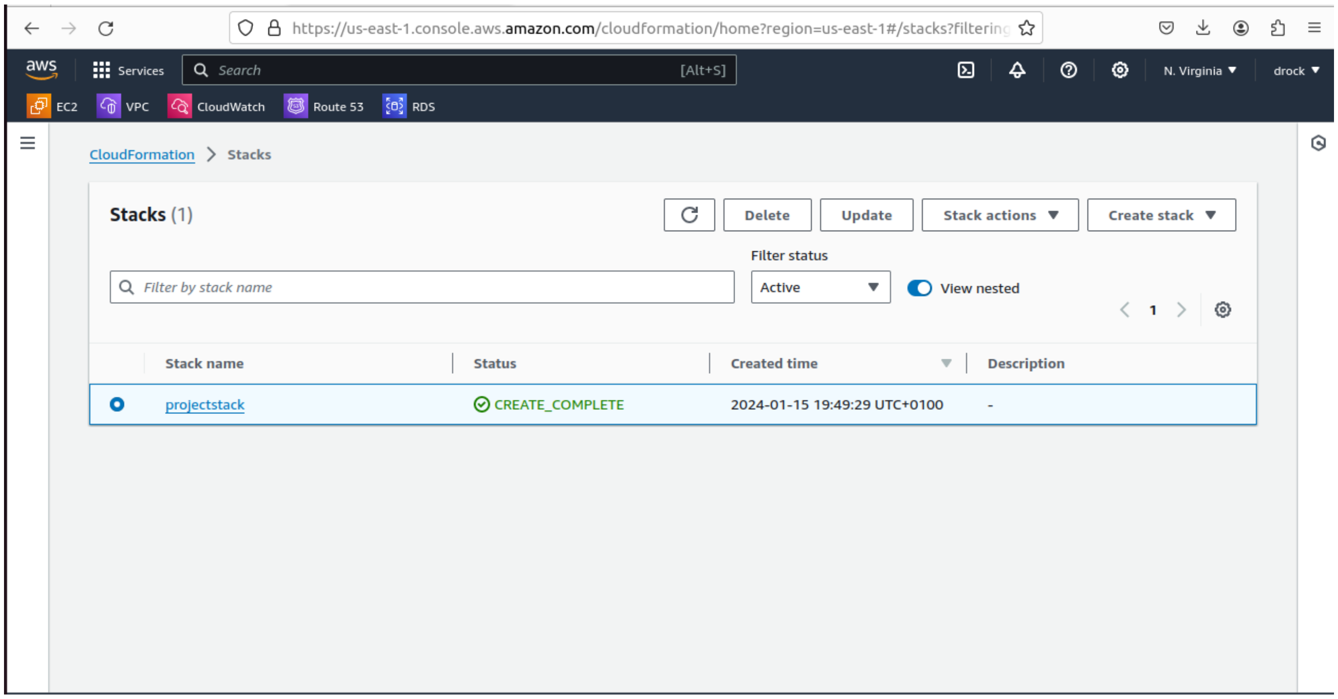
projectstack

Delete | Update | Stack actions | Create stack

Stack info | Events | Resources | Outputs | Parameters

Events (27)

Timestamp	Logical ID	Status
2024-01-15 19:50:17 UTC+0100	AppInstance	CREATE_COMPLETE
2024-01-15 19:50:11 UTC+0100	WebInstance	CREATE_COMPLETE
2024-01-15 19:49:55 UTC+0100	DBInstance	CREATE_IN_PROGRESS


https://us-east-1.console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks?filtering

CloudFormation > Stacks

Stacks (1)

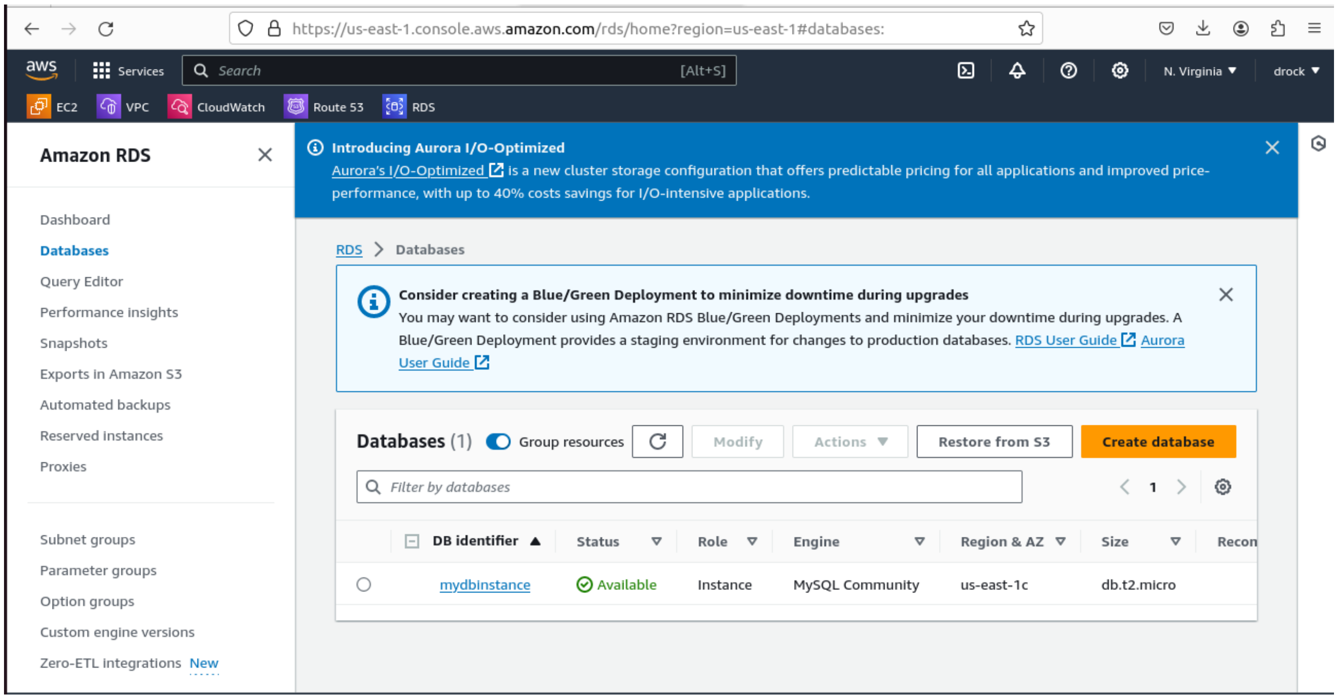
Delete | Update | Stack actions | Create stack

Filter status: Active

View nested

	Stack name	Status	Created time	Description
●	projectstack	CREATE_COMPLETE	2024-01-15 19:49:29 UTC+0100	-

RDS CREATED



https://us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#databases:

Amazon RDS

Introducing Aurora I/O-Optimized
Aurora's I/O-Optimized is a new cluster storage configuration that offers predictable pricing for all applications and improved price-performance, with up to 40% costs savings for I/O-intensive applications.

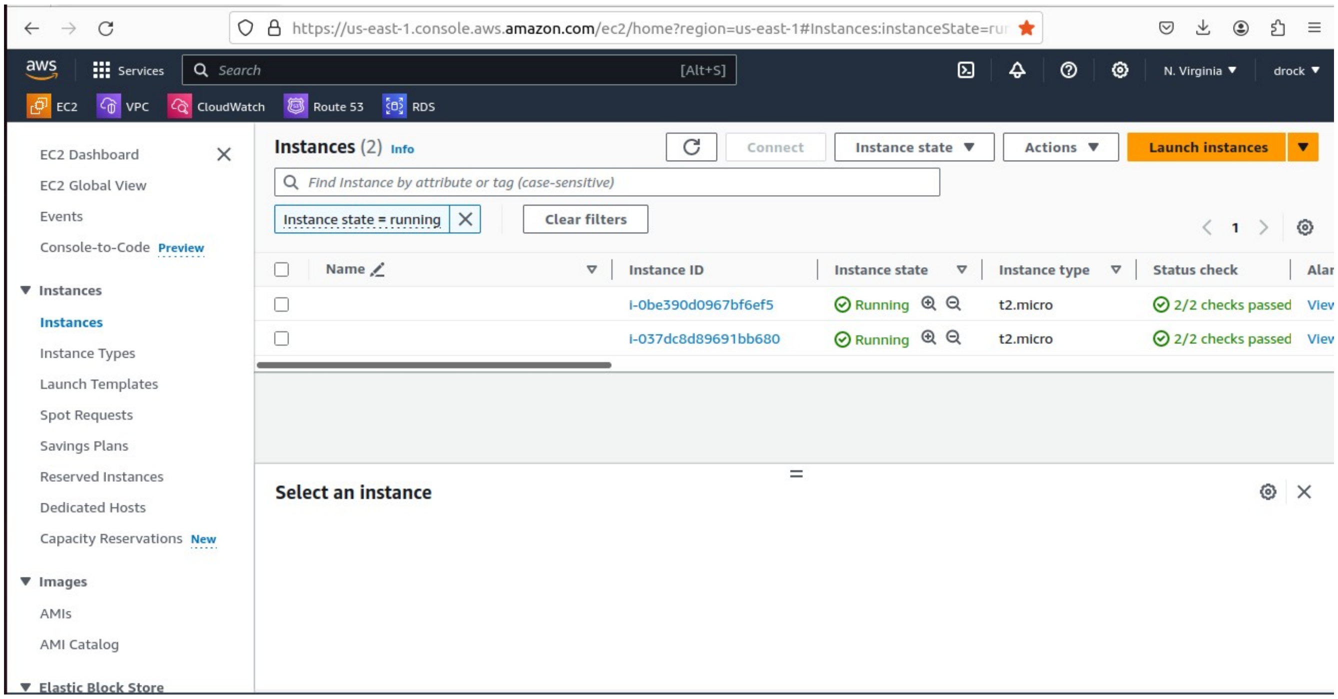
RDS > Databases

Consider creating a Blue/Green Deployment to minimize downtime during upgrades
You may want to consider using Amazon RDS Blue/Green Deployments and minimize your downtime during upgrades. A Blue/Green Deployment provides a staging environment for changes to production databases. RDS User Guide Aurora User Guide

Databases (1)

DB identifier	Status	Role	Engine	Region & AZ	Size	Recon
mydbinstance	Available	Instance	MySQL Community	us-east-1c	db.t2.micro	

WEB AND APPLICATION INSTANCE CREATION



https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:instanceState=ru

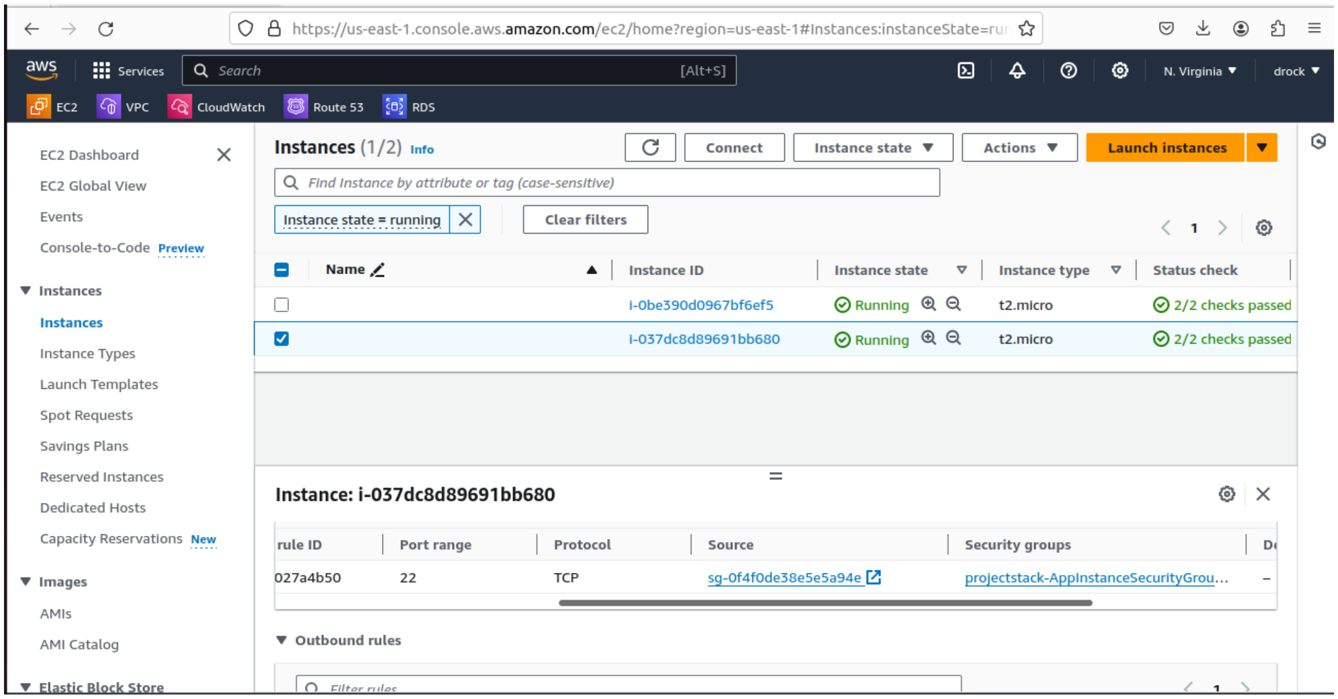
Instances (2)

Instance state = running

Name	Instance ID	Instance state	Instance type	Status check	Alar
	i-0be390d0967bf6ef5	Running	t2.micro	2/2 checks passed	View
	i-037dc8d89691bb680	Running	t2.micro	2/2 checks passed	View

Select an instance

APPLICATION SERVER; WITH APPLICATION SECURITY SPECIFICATION



Instances (1/2) Info

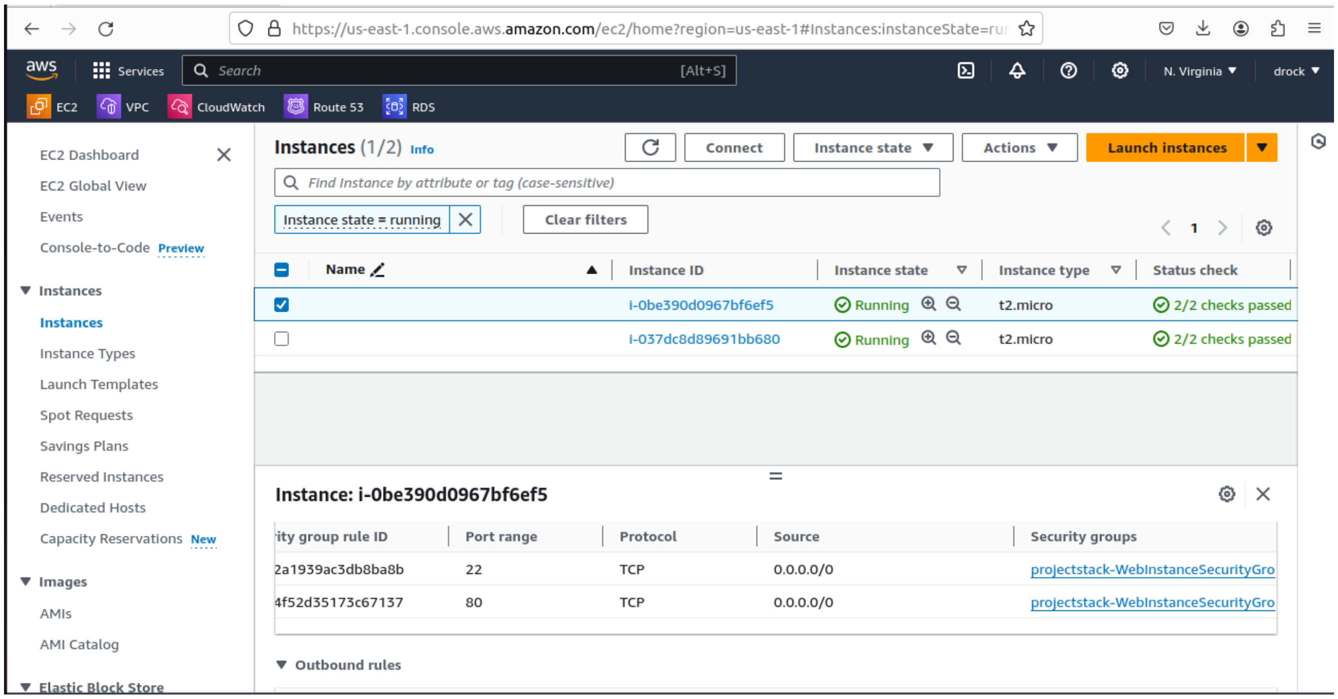
	Name	Instance ID	Instance state	Instance type	Status check
☐		i-0be390d0967bf6ef5	Running	t2.micro	2/2 checks passed
☑		i-037dc8d89691bb680	Running	t2.micro	2/2 checks passed

Instance: i-037dc8d89691bb680

rule ID	Port range	Protocol	Source	Security groups	De
027a4b50	22	TCP	sg-0f4f0de38e5e5a94e	projectstack-AppInstanceSecurityGrou…	–

Outbound rules

WEB SERVER WITH WEB SERVER SECURITY SPECIFICATIONS



Instances (1/2) Info

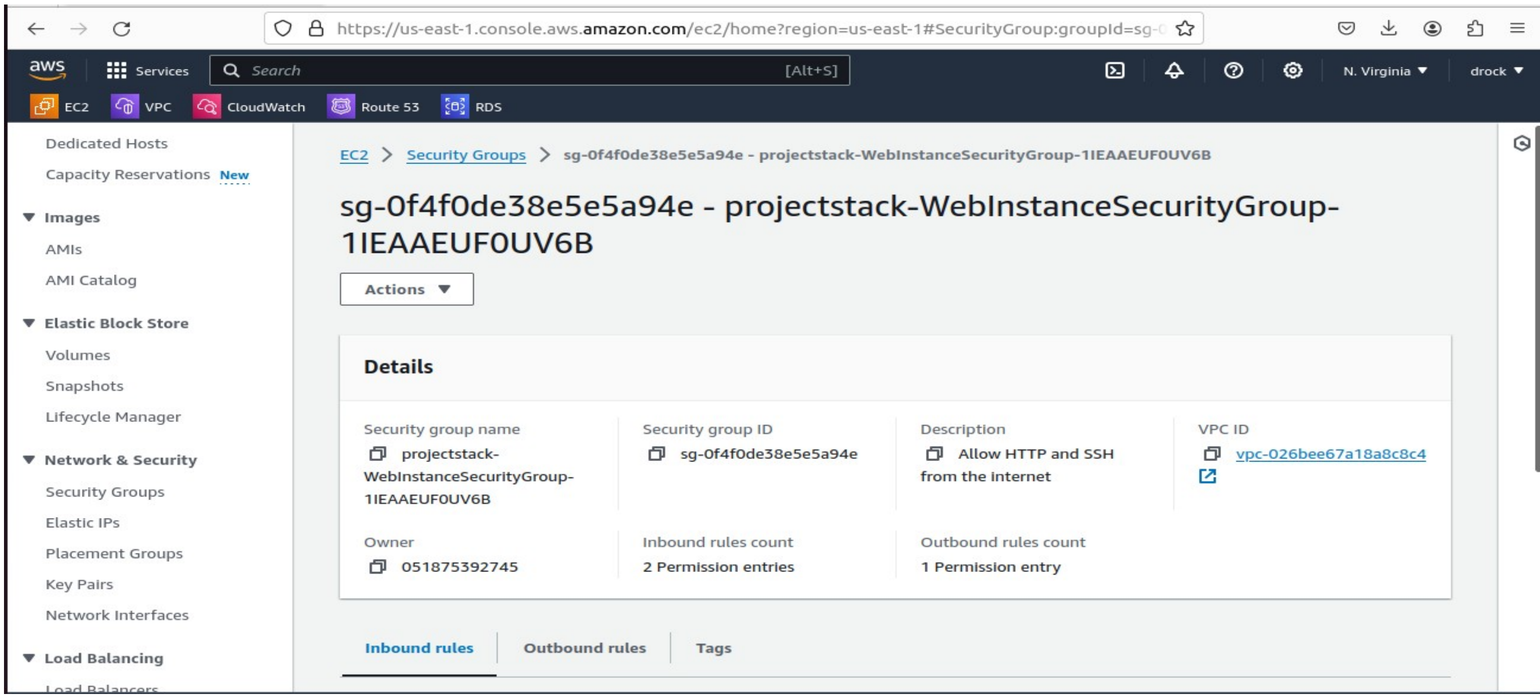
	Name	Instance ID	Instance state	Instance type	Status check
☑		i-0be390d0967bf6ef5	Running	t2.micro	2/2 checks passed
☐		i-037dc8d89691bb680	Running	t2.micro	2/2 checks passed

Instance: i-0be390d0967bf6ef5

ity group rule ID	Port range	Protocol	Source	Security groups
2a1939ac3db8ba8b	22	TCP	0.0.0.0/0	projectstack-WebInstanceSecurityGro
4f52d35173c67137	80	TCP	0.0.0.0/0	projectstack-WebInstanceSecurityGro

Outbound rules

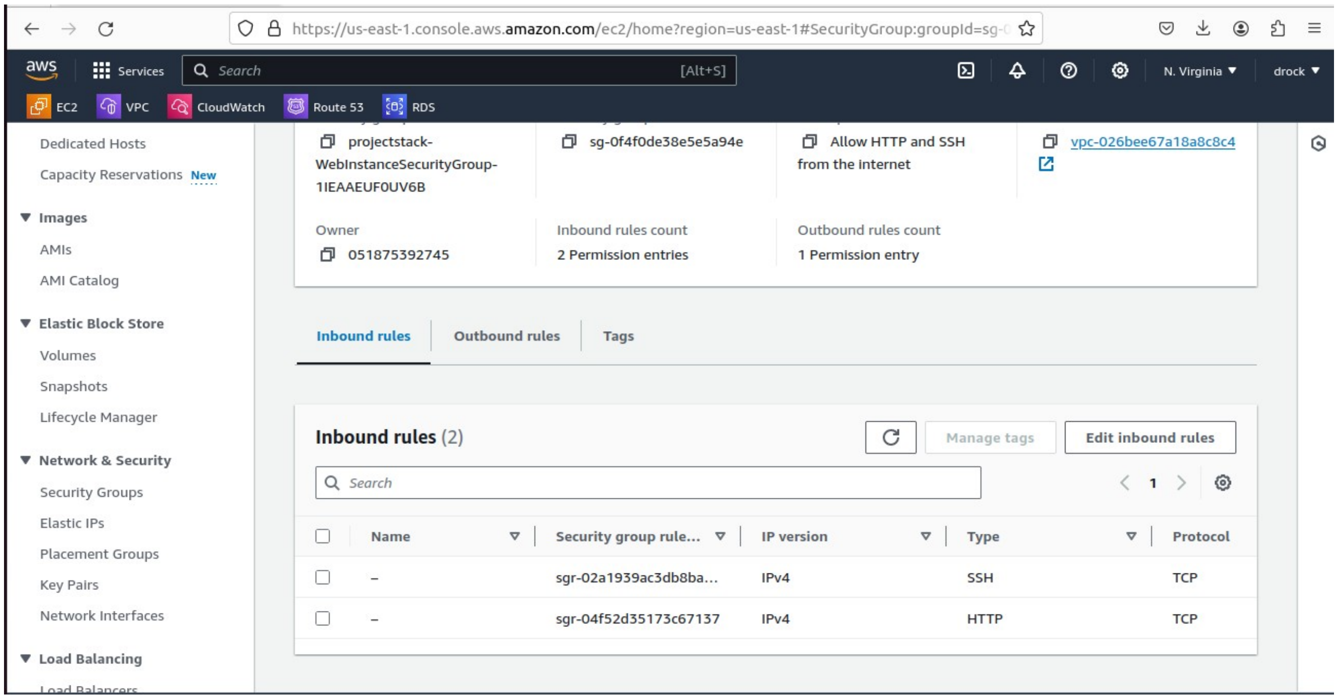
DETAILS OF WEBINSTANCE SECURITY GROUP ID



sg-0f4f0de38e5e5a94e - projectstack-WebInstanceSecurityGroup-1IEAAEUF0UV6B

Details

Security group name	Security group ID	Description	VPC ID
projectstack-WebInstanceSecurityGroup-1IEAAEUF0UV6B	sg-0f4f0de38e5e5a94e	Allow HTTP and SSH from the internet	vpc-026bee67a18a8c8c4
Owner	Inbound rules count	Outbound rules count	
051875392745	2 Permission entries	1 Permission entry	

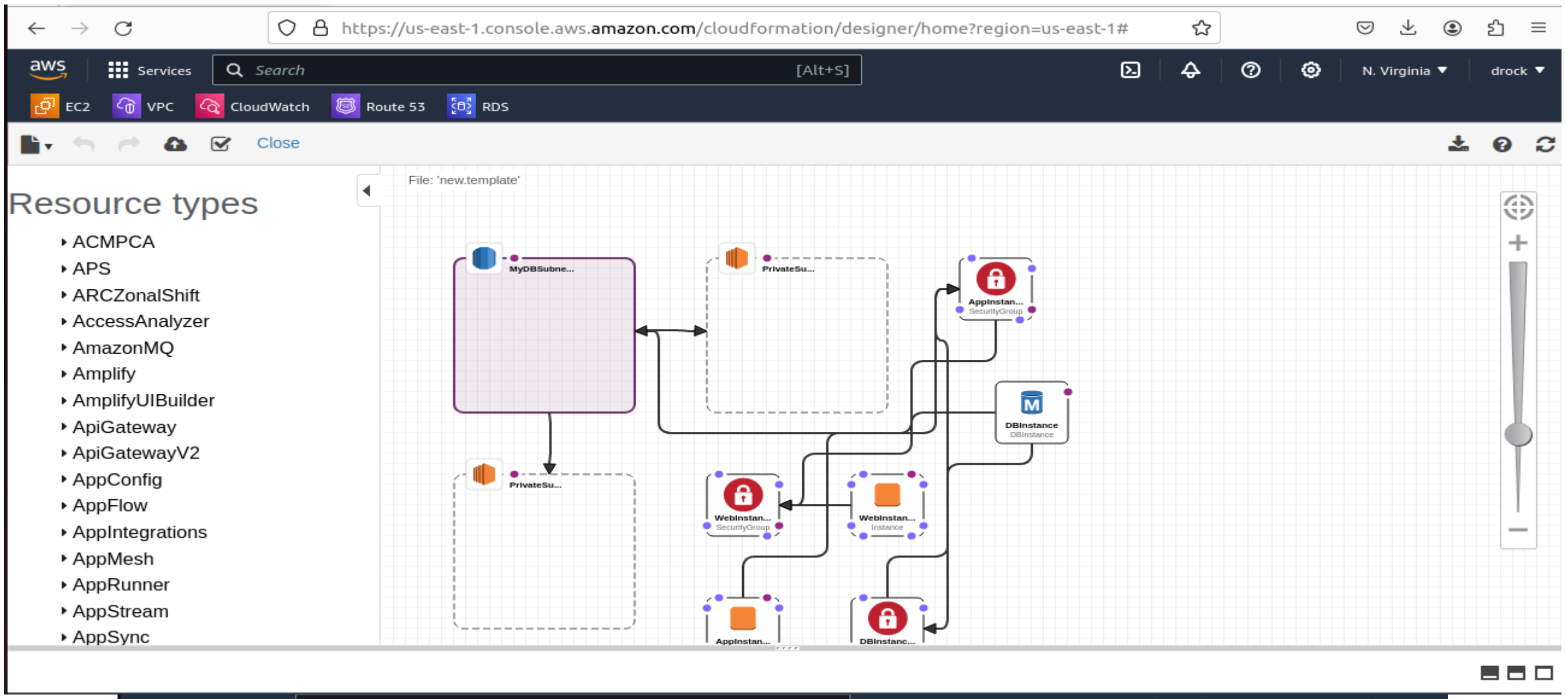


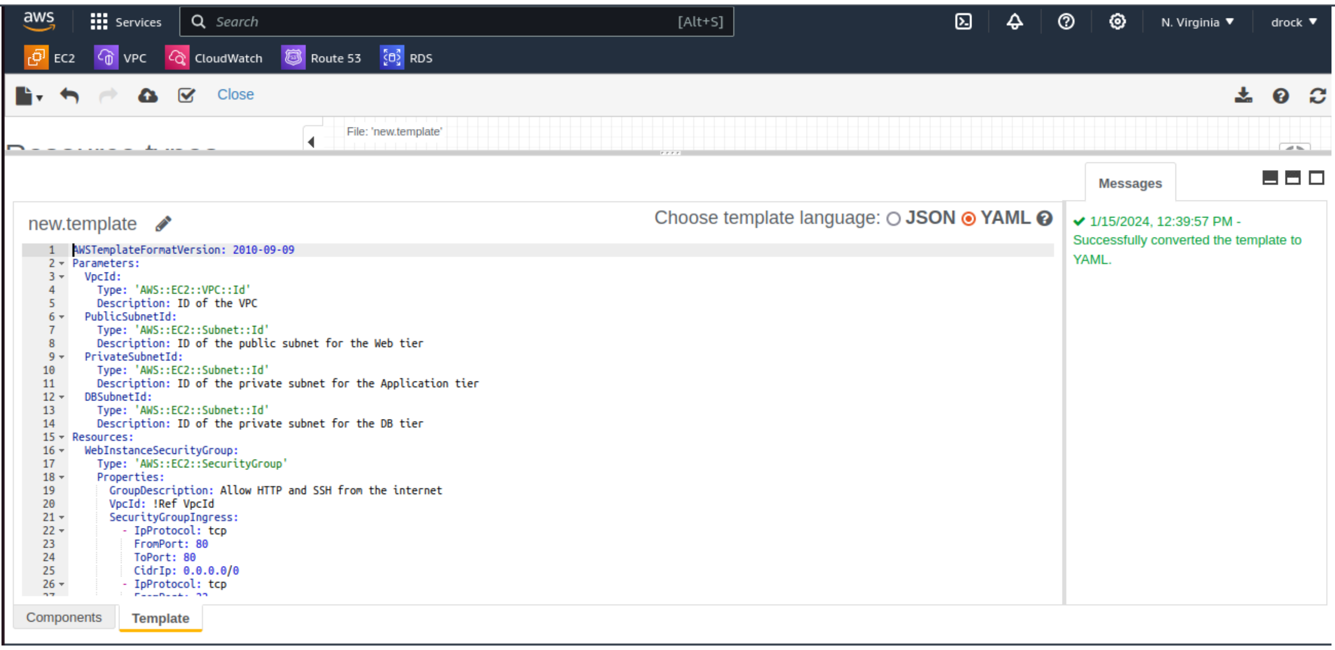
projectstack-WebInstanceSecurityGroup-1IEAAEUF0UV6B	sg-0f4f0de38e5e5a94e	Allow HTTP and SSH from the internet	vpc-026bee67a18a8c8c4
Owner	Inbound rules count	Outbound rules count	
051875392745	2 Permission entries	1 Permission entry	

Inbound rules (2)

Name	Security group rule...	IP version	Type	Protocol
–	sgr-02a1939ac3db8ba...	IPv4	SSH	TCP
–	sgr-04f52d35173c67137	IPv4	HTTP	TCP

STACK DETAILS





new.template

Choose template language: JSON [✓ YAML]

```
AWSTemplateFormatVersion: 2010-09-09
Parameters:
  VpcId:
    Type: 'AWS::EC2::VPC::Id'
    Description: ID of the VPC
  PublicSubnetId:
    Type: 'AWS::EC2::Subnet::Id'
    Description: ID of the public subnet for the Web tier
  PrivateSubnetId:
    Type: 'AWS::EC2::Subnet::Id'
    Description: ID of the private subnet for the Application tier
  DBSubnetId:
    Type: 'AWS::EC2::Subnet::Id'
    Description: ID of the private subnet for the DB tier
Resources:
  WebInstanceSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Allow HTTP and SSH from the internet
      VpcId: !Ref VpcId
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
```

✓ 1/15/2024, 12:39:57 PM - Successfully converted the template to YAML.

DETAILS OF THE YAML FILE

```
AWSTemplateFormatVersion: 2010-09-09
Parameters:
  VpcId:
    Type: 'AWS::EC2::VPC::Id'
    Description: ID of the VPC
  PublicSubnetId:
    Type: 'AWS::EC2::Subnet::Id'
    Description: ID of the public subnet for the Web tier
  PrivateSubnetId:
    Type: 'AWS::EC2::Subnet::Id'
    Description: ID of the private subnet for the Application tier
  DBSubnetId:
    Type: 'AWS::EC2::Subnet::Id'
    Description: ID of the private subnet for the DB tier
Resources:
  WebInstanceSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Allow HTTP and SSH from the internet
      VpcId: !Ref VpcId
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: 0.0.0.0/0
  AppInstanceSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Allow only SSH from the public subnet of Web Tier-3
      VpcId: !Ref VpcId
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          SourceSecurityGroupId: !Ref WebInstanceSecurityGroup
  DBInstanceSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: >-
        Allow connection on port 3306 only from the private subnet of
        Application Tier-4
      VpcId: !Ref VpcId
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 3306
          ToPort: 3306
          SourceSecurityGroupId: !Ref AppInstanceSecurityGroup
  WebInstance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ImageId: ami-0005e0cfe09cc9050
      InstanceType: t2.micro
      KeyName: newkey-virginia
      SubnetId: !Ref PublicSubnetId
      SecurityGroupIds:
        - !Ref WebInstanceSecurityGroup
      UserData: !Base64
        'Fn::Sub': |
          #!/bin/bash
          # Your initialization script here
  AppInstance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ImageId: ami-0005e0cfe09cc9050
      InstanceType: t2.micro
      KeyName: newkey-virginia
      SubnetId: !Ref PrivateSubnetId
      SecurityGroupIds:
        - !Ref AppInstanceSecurityGroup
      UserData: !Base64
        'Fn::Sub': |
          #!/bin/bash
          # Your initialization script here
  PrivateSubnet1:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref VpcId
      CidrBlock: '172.31.96.0/24'
      # Add other necessary properties

  PrivateSubnet2:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref VpcId
      CidrBlock: '172.31.112.0/24'
      # Add other necessary properties

  MyDBSubnetGroup:
    Type: 'AWS::RDS::DBSubnetGroup'
    Properties:
      DBSubnetGroupDescription: My DB Subnet Group
      SubnetIds:
        - !Ref PrivateSubnet1
        - !Ref PrivateSubnet2
        # Add more subnets if needed
  DBInstance:
    Type: 'AWS::RDS::DBInstance'
    Properties:
      Engine: mysql
      DBInstanceIdentifier: MyDBInstance
      MasterUsername: admin
      MasterUserPassword: adminpassword
      AllocatedStorage: 20
      DBInstanceClass: db.t2.micro
      VPCSecurityGroups:
        - !Ref DBInstanceSecurityGroup
      MultiAZ: false
      StorageType: gp2
      DBSubnetGroupName: !Ref MyDBSubnetGroup
Outputs:
  WebInstance:
    Description: Public IP of the EC2 instance in the Web tier
    Value: !GetAtt
      - WebInstance
      - PublicIp
  AppInstance:
    Description: Private IP of the EC2 instance in the Application tier
    Value: !GetAtt
      - AppInstance
      - PrivateIp
  DBInstance:
    Description: DB Instance Endpoint of the RDS MySQL instance in the DB tier
    Value: !GetAtt
      - DBInstance
      - Endpoint.Address
```

2.
Make sure when the development team deletes the stack, RDS DB instances should not be deleted.

Here Deletion Protection under RDS is set to TRUE

The cloudformation Template :RDS DB INSTANCE (Deletion protection) is copied below:

```
AWSTemplateFormatVersion: 2010-09-09
Parameters:
  VpcId:
    Type: 'AWS::EC2::VPC::Id'
    Description: ID of the VPC
  PublicSubnetId:
    Type: 'AWS::EC2::Subnet::Id'
    Description: ID of the public subnet for the Web tier
  PrivateSubnetId:
    Type: 'AWS::EC2::Subnet::Id'
    Description: ID of the private subnet for the Application tier
  DBSubnetId:
    Type: 'AWS::EC2::Subnet::Id'
    Description: ID of the private subnet for the DB tier
Resources:
  WebInstanceSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Allow HTTP and SSH from the internet
      VpcId: !Ref VpcId
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: 0.0.0.0/0
  AppInstanceSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Allow only SSH from the public subnet of Web Tier-3
      VpcId: !Ref VpcId
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          SourceSecurityGroupId: !Ref WebInstanceSecurityGroup
  DBInstanceSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: >-
        Allow connection on port 3306 only from the private subnet of
        Application Tier-4
      VpcId: !Ref VpcId
      SecurityGroupIngress:
```

```
      - IpProtocol: tcp
        FromPort: 3306
        ToPort: 3306
        SourceSecurityGroupId: !Ref AppInstanceSecurityGroup
WebInstance:
  Type: 'AWS::EC2::Instance'
  Properties:
    ImageId: ami-0005e0cfe09cc9050
    InstanceType: t2.micro
    KeyName: newkey-virginia
    SubnetId: !Ref PublicSubnetId
    SecurityGroupIds:
      - !Ref WebInstanceSecurityGroup
    UserData: !Base64
      'Fn::Sub': |
        #!/bin/bash
        # Your initialization script here
AppInstance:
  Type: 'AWS::EC2::Instance'
  Properties:
    ImageId: ami-0005e0cfe09cc9050
    InstanceType: t2.micro
    KeyName: newkey-virginia
    SubnetId: !Ref PrivateSubnetId
    SecurityGroupIds:
      - !Ref AppInstanceSecurityGroup
    UserData: !Base64
      'Fn::Sub': |
        #!/bin/bash
        # Your initialization script here
PrivateSubnet1:
  Type: 'AWS::EC2::Subnet'
  Properties:
    VpcId: !Ref VpcId
    CidrBlock: '172.31.96.0/24'
    # Add other necessary properties

PrivateSubnet2:
  Type: 'AWS::EC2::Subnet'
  Properties:
    VpcId: !Ref VpcId
    CidrBlock: '172.31.112.0/24'
    # Add other necessary properties

MyDBSubnetGroup:
  Type: 'AWS::RDS::DBSubnetGroup'
  Properties:
    DBSubnetGroupDescription: My DB Subnet Group
    SubnetIds:
      - !Ref PrivateSubnet1
      - !Ref PrivateSubnet2
      # Add more subnets if needed
DBInstance:
  Type: 'AWS::RDS::DBInstance'
  Properties:
    Engine: mysql
    DBInstanceIdentifier: MyDBInstance
    MasterUsername: admin
```

```
      MasterUserPassword: adminpassword
      AllocatedStorage: 20
      DBInstanceClass: db.t2.micro
      VPCSecurityGroups:
        - !Ref DBInstanceSecurityGroup
      MultiAZ: false
      StorageType: gp2
      DBSubnetGroupName: !Ref MyDBSubnetGroup
      DeletionProtection: true
Outputs:
  WebInstance:
    Description: Public IP of the EC2 instance in the Web tier
    Value: !GetAtt
      - WebInstance
      - PublicIp
  AppInstance:
    Description: Private IP of the EC2 instance in the Application tier
    Value: !GetAtt
      - AppInstance
      - PrivateIp
  DBInstance:
    Description: DB Instance Endpoint of the RDS MySQL instance in the DB tier
    Value: !GetAtt
      - DBInstance
      - Endpoint.Address
```

Propose a solution so that:
Development team can test their code without having to involve the system admins and can invest their time in testing the code rather than provisioning, configuring and updating the resources needed to test the code.

To enable the development team to test their code without involving system administrators and to streamline the process, you can use AWS CloudFormation to automate the provisioning and management of resources. Here's a solution:

AWS CloudFormation Template:

Create an AWS CloudFormation template that defines the infrastructure needed for the web-based application. This includes instances, security groups, RDS instances, and any other required resources.

Launch Stack:

Provide the development team with a pre-configured AWS CloudFormation template. They can use the AWS Management Console, AWS CLI, or SDKs to launch the stack without having to manually provision resources.

Parameterize Key Configuration:

Use parameters in the CloudFormation template for configurable values such as instance types, AMI IDs, and database credentials. This allows the development team to customize the deployment without modifying the template.

Scripted Deployment:

Encourage the development team to use scripts or automation tools that leverage AWS CloudFormation. This can be integrated into their continuous integration/continuous deployment (CI/CD) pipelines, making it easy to deploy and test code changes.

Version Control:

Store the CloudFormation template in version control (e.g., Git). This enables the development team to track changes, collaborate, and roll back to previous versions if needed.