Tasks To Be Performed:

1. Web tier: Launch an instance in a public subnet and that instance should allow HTTP and SSH from the internet.

2. Application tier: Launch an instance in a private subnet of the web tier and it should allow only SSH from the public subnet of Web Tier-3.

3. DB tier: Launch an RDS MYSQL instance in a private subnet and it should allow connection on port 3306 only from the private subnet of Application Tier-4.

4. Setup a Route 53 hosted zone and direct traffic to the EC2 instance.

You have been also asked to propose a solution so that:

1. 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.

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

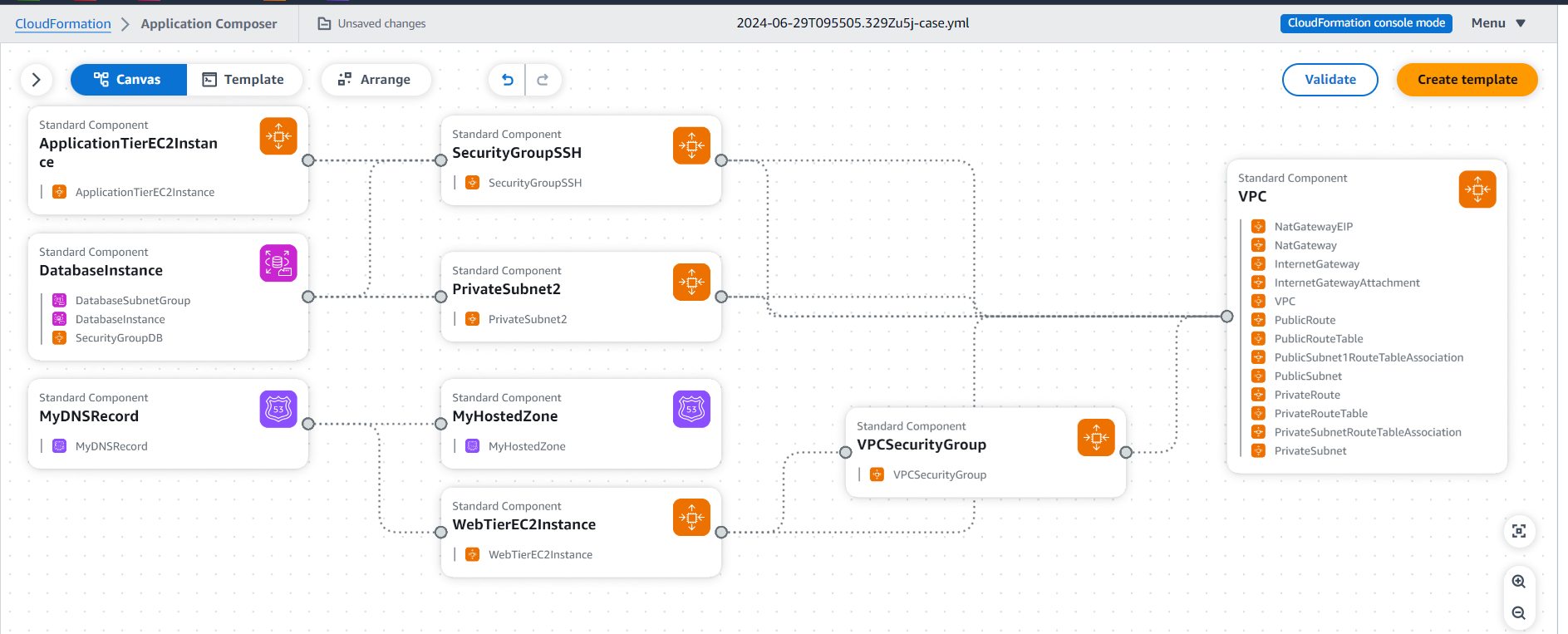
Create ymal template for above all 4 pints

1)WEB Tier 2) Application Tier 3) DB Tier and 4) Route 53

create ymal template

1. AWSTemplateFormatVersion: "2010-09-09"
2. Description : to implement case study

|  |  |  |
| --- | --- | --- |
| **4)     Parameters** | **5) Resources** | **6)OutPut** |
| ·       WebtierVpcCIDR | VPC | VPC |
| ·       WebtierPublicSubnetCIDR | InternetGateway | PublicSubnet |
| ·       WebtierprivateSubnet1CIDR | InternetGatewayAttachment | PrivateSubnet |
| ·       WebtierprivateSubnet2CIDR | PublicSubnet | PrivateSubnet2 |
| ·       myAMIid | PrivateSubnet | WebTierEC2Instance |
| ·       mykeypair | PublicRouteTable | ApplicationTierInstance |
| ·       EC2InstanceTpe | PublicRoute | VPCSecurityGroup |
| ·       DatabaseInstanceIdentifier | PublicSubnet1RouteTableAssociation | SecurityGroupSSH |
| ·       DatabaseName | NatGatewayEIP | SecurityGroupDB |
| ·       DatabaseUser | NatGateway | NatGateway |
| ·       DatabasePassword | PrivateRouteTable | InstancePublicIp |
| ·       DatabaseBackupRetentionPeriod | PrivateRoute |  |
| ·       DatabaseAllocatedStorage | PrivateSubnetRouteTableAssociation |  |
| ·       DatabaseInstanceClass | VPCSecurityGroup |  |
|  | SecurityGroupSSH |  |
|  | SecurityGroupDB |  |
|  | WebTierEC2Instance |  |
|  | ApplicationTierEC2Instance |  |
|  | DatabaseSubnetGroup |  |
|  | DatabaseInstance |  |
|  | MyHostedZone |  |
|  | MyDNSRecord |  |



AWSTemplateFormatVersion: "2010-09-09"

Description: to run casestudy

Parameters:

  WebtierVpcCIDR:

    Default: 10.0.0.0/16

    Description: pleae Enter VPC ip range

    Type: String

  WebtierPublicSubnetCIDR:

    Default: 10.0.5.0/24

    Description: pleae Enter PublicSubnet ip range

    Type: String

  WebtierprivateSubnet1CIDR:

    Default: 10.0.3.1/24

    Description: pleae Enter PrivateSubnet ip range

    Type: String

  WebtierprivateSubnet2CIDR:

    Default: 10.0.11.1/24

    Description: pleae Enter PrivateSubnet ip range

    Type: String

  mykeypair:

      Type: String

      Description: key pair

  myAMIid:

      Type: String

      Description: ami id

  EC2InstanceTpe:

      Description: Select Instance Type.

      Type: String

      Default: t2.micro

      AllowedValues: [t2.micro,t2.small,t3.medium]

  DatabaseInstanceIdentifier:

    AllowedPattern: '[a-zA-Z][a-zA-Z0-9]\*'

    ConstraintDescription: Must begin with a letter and contain only alphanumeric characters

    Default: myDB2024

    Description: Instance identifier name

    MaxLength: 60

    MinLength: 1

    Type: String

  DatabaseName:

    AllowedPattern: '[a-zA-Z][a-zA-Z0-9]\*'

    ConstraintDescription: Must begin with a letter and contain only alphanumeric characters

    Default: databasetier

    Description: MySQL database name

    MaxLength: 64

    MinLength: 1

    Type: String

  DatabaseUser:

    AllowedPattern: '[a-zA-Z][a-zA-Z0-9]\*'

    ConstraintDescription: Must begin with a letter and contain only alphanumeric characters

    Default: dbadmin

    Description: Username for MySQL database access

    MaxLength: 16

    MinLength: 1

    NoEcho: true

    Type: String

  DatabasePassword:

    AllowedPattern: '[a-zA-Z0-9]\*'

    ConstraintDescription: Must contain only alphanumeric characters

    Default: my278217

    Description: Password for MySQL database access

    MaxLength: 41

    MinLength: 8

    NoEcho: true

    Type: String

  DatabaseBackupRetentionPeriod:

    ConstraintDescription: Database backup retention period must be between 0 and 35 days

    Default: 0

    Description: The number of days for which automatic DB snapshots are retained

    MaxValue: 35

    MinValue: 0

    Type: Number

  DatabaseAllocatedStorage:

    ConstraintDescription: Must be between 5 and 1024Gb

    Default: 20

    Description: The size of the database (Gb)

    MaxValue: 65536

    MinValue: 5

    Type: Number

  DatabaseInstanceClass:

    AllowedValues:

      - db.t1.micro

      - db.t3.micro

      - db.m1.small

      - db.m1.medium

      - db.m1.large

    ConstraintDescription: Must select a valid database instance type

    Default: db.t3.micro

    Description: The database instance type

    Type: String

Resources:

   VPC:

      Type: AWS::EC2::VPC

      Properties:

        CidrBlock: !Ref WebtierVpcCIDR

        EnableDnsHostnames: true

        EnableDnsSupport: true

        InstanceTenancy: default

        Tags:

          - Key: Name

            Value: WebtierVpcCIDR

   InternetGateway:

      Type: AWS::EC2::InternetGateway

      Properties:

        Tags:

          - Key: Name

            Value: WebtierIGW

   InternetGatewayAttachment:

      Type: AWS::EC2::VPCGatewayAttachment

      Properties:

        InternetGatewayId: !Ref InternetGateway

        VpcId: !Ref VPC

   PublicSubnet:

      Type: AWS::EC2::Subnet

      Properties:

        AvailabilityZone: !Select [0, !GetAZs '']

        CidrBlock: !Ref WebtierPublicSubnetCIDR

        MapPublicIpOnLaunch: true

        Tags:

          - Key: Name

            Value: test PublicSubnet | Web Tier

        VpcId: !Ref VPC

   PrivateSubnet:

      Type: AWS::EC2::Subnet

      Properties:

        AvailabilityZone: !Select [0, !GetAZs '']

        CidrBlock: !Ref WebtierprivateSubnet1CIDR

        MapPublicIpOnLaunch: true

        Tags:

          - Key: Name

            Value: test PrivateSubnet1 | Web Tier

        VpcId: !Ref VPC

   PrivateSubnet2:

      Type: AWS::EC2::Subnet

      Properties:

        AvailabilityZone: !Select [1, !GetAZs '']

        CidrBlock: !Ref WebtierprivateSubnet2CIDR

        MapPublicIpOnLaunch: true

        Tags:

          - Key: Name

            Value: test PrivateSubnet2 | Web Tier

        VpcId: !Ref VPC

   PublicRouteTable:

      Type: AWS::EC2::RouteTable

      Properties:

        Tags:

          - Key: Name

            Value: Test PublicRouteTable

        VpcId: !Ref VPC

   PublicRoute:

      Type: AWS::EC2::Route

      Properties:

        DestinationCidrBlock: 0.0.0.0/0

        GatewayId: !Ref InternetGateway

        RouteTableId: !Ref PublicRouteTable

   PublicSubnet1RouteTableAssociation:

      Type: AWS::EC2::SubnetRouteTableAssociation

      Properties:

        RouteTableId: !Ref PublicRouteTable

        SubnetId: !Ref PublicSubnet

   NatGatewayEIP:

      Type: AWS::EC2::EIP

      Properties:

        Domain: VPC

        Tags:

        - Key: NAME

          Value: EIP

   NatGateway:

      Type: AWS::EC2::NatGateway

      Properties:

        AllocationId: !GetAtt NatGatewayEIP.AllocationId

        SubnetId: !Ref PublicSubnet

        Tags:

        - Key: NAME

          Value: NAT gw for public Subnet

   PrivateRouteTable:

      Type: AWS::EC2::RouteTable

      Properties:

        Tags:

        - Key: Name

          Value: Private Route Table

        VpcId: !Ref VPC

   PrivateRoute:

    Type: AWS::EC2::Route

    Properties:

      DestinationCidrBlock: 0.0.0.0/0

      NatGatewayId: !Ref NatGateway

      RouteTableId: !Ref PrivateRouteTable

   PrivateSubnetRouteTableAssociation:

      Type: AWS::EC2::SubnetRouteTableAssociation

      Properties:

        RouteTableId: !Ref PrivateRouteTable

        SubnetId: !Ref PrivateSubnet

   VPCSecurityGroup:

      Type: 'AWS::EC2::SecurityGroup'

      Properties:

        GroupDescription: Enable HTTP/HTTP form port 22/80

        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

        Tags:

          - Key: Name

            Value: VPC SG

        VpcId: !Ref VPC

   SecurityGroupSSH:

      Type: 'AWS::EC2::SecurityGroup'

      Properties:

        GroupDescription: Enable SSH form port 22

        SecurityGroupIngress:

          - IpProtocol: tcp

            FromPort: 22

            ToPort: 22

            CidrIp: 0.0.0.0/0

          - IpProtocol: tcp

            FromPort: 3306

            ToPort: 3306

            CidrIp: 0.0.0.0/0

        Tags:

          - Key: Name

            Value: VPC SG

        VpcId: !Ref VPC

   SecurityGroupDB:

      Type: 'AWS::EC2::SecurityGroup'

      Properties:

        GroupDescription: Enable SSH form port 3306

        SecurityGroupIngress:

          - IpProtocol: tcp

            FromPort: 3306

            ToPort: 3306

            SourceSecurityGroupId: !Ref SecurityGroupSSH

        Tags:

          - Key: Name

            Value: VPC SG

        VpcId: !Ref VPC

   WebTierEC2Instance:

      Type: AWS::EC2::Instance

      Properties:

        KeyName: !Ref mykeypair

        DisableApiTermination: false

        ImageId: !Ref myAMIid

        AvailabilityZone: !Select [0, !GetAZs '']

        InstanceType: !Ref EC2InstanceTpe

        Monitoring: false

        NetworkInterfaces:

        - AssociatePublicIpAddress: "true"

          DeviceIndex: "0"

          GroupSet:

            - !Ref VPCSecurityGroup

          SubnetId: !Ref PublicSubnet

        Tags:

          - Key: Name

            Value: webTierInstance

   ApplicationTierEC2Instance:

      Type: AWS::EC2::Instance

      Properties:

        KeyName: !Ref mykeypair

        DisableApiTermination: false

        ImageId: !Ref myAMIid

        AvailabilityZone: !Select [0, !GetAZs '']

        InstanceType: !Ref EC2InstanceTpe

        Monitoring: false

        NetworkInterfaces:

        - AssociatePublicIpAddress: "False"

          DeviceIndex: "0"

          GroupSet:

            - !Ref SecurityGroupSSH

          SubnetId: !Ref PrivateSubnet

        Tags:

          - Key: Name

            Value: ApplicationTierInstance

   DatabaseSubnetGroup:

    Type: AWS::RDS::DBSubnetGroup

    Properties:

      DBSubnetGroupDescription: subnet group for RDS

      SubnetIds:

         - !Ref PrivateSubnet

         - !Ref PrivateSubnet2

      Tags:

        - Key: Name

          Value: DB Subnet

   DatabaseInstance:

    Type: AWS::RDS::DBInstance

    DeletionPolicy: Retain

    Properties:

      AllocatedStorage: !Ref DatabaseAllocatedStorage

      AvailabilityZone:  !Select [ 0, !GetAZs  '' ]

      BackupRetentionPeriod: !Ref DatabaseBackupRetentionPeriod

      DBInstanceClass: !Ref DatabaseInstanceClass

      DBInstanceIdentifier: !Ref DatabaseInstanceIdentifier

      DBName: !Ref DatabaseName

      DBSubnetGroupName: !Ref DatabaseSubnetGroup

      Engine: MySQL

      EngineVersion: 8.0.35

      MasterUsername: !Ref DatabaseUser

      MasterUserPassword: !Ref DatabasePassword

      VPCSecurityGroups:

        -  !Ref SecurityGroupDB

   MyHostedZone:

    Type: AWS::Route53::HostedZone

    Properties:

      Name: karaws.com

   MyDNSRecord:

    Type: AWS::Route53::RecordSet

    Properties:

      HostedZoneId: !Ref MyHostedZone

      Name: www.karaws.com

      Type: A

      TTL: '300'

      ResourceRecords:

        - !GetAtt WebTierEC2Instance.PublicIp

Outputs:

    VPC:

      Description: VPC ID

      Export:

        Name: !Sub ${AWS::StackName}-VPC

      Value: !Ref VPC

    PublicSubnet:

        Description: Public Subnet ID

        Export:

          Name: !Sub ${AWS::StackName}-PublicSubnet

        Value: !Ref PublicSubnet

    PrivateSubnet:

        Description: Public Subnet ID

        Export:

          Name: !Sub ${AWS::StackName}-PrivateSubnet

        Value: !Ref PrivateSubnet

    PrivateSubnet2:

        Description: Public Subnet ID

        Export:

          Name: !Sub ${AWS::StackName}-PrivateSubnet2

        Value: !Ref PrivateSubnet2

    WebTierEC2Instance:

        Description: ip of Ec2 instance

        Export:

          Name: !Sub ${AWS::StackName}-WebTierEC2Instance

        Value: !Ref WebTierEC2Instance

    ApplicationTierInstance:

        Description: ip of Ec2 instance

        Export:

          Name: !Sub ${AWS::StackName}-ApplicationTierInstance

        Value: !Ref ApplicationTierEC2Instance

    VPCSecurityGroup:

        Description: SG details

        Export:

          Name: !Sub ${AWS::StackName}-VPCSecurityGroup

        Value: !Ref VPCSecurityGroup

    SecurityGroupSSH:

        Description: SG details

        Export:

          Name: !Sub ${AWS::StackName}-SecurityGroupSSH

        Value: !Ref SecurityGroupSSH

    SecurityGroupDB:

        Description: SG details

        Export:

          Name: !Sub ${AWS::StackName}-SecurityGroupDB

        Value: !Ref SecurityGroupDB

    NatGateway:

        Description: NAT GW

        Export:

          Name: !Sub ${AWS::StackName}-NatGateway

        Value: !Ref NatGateway

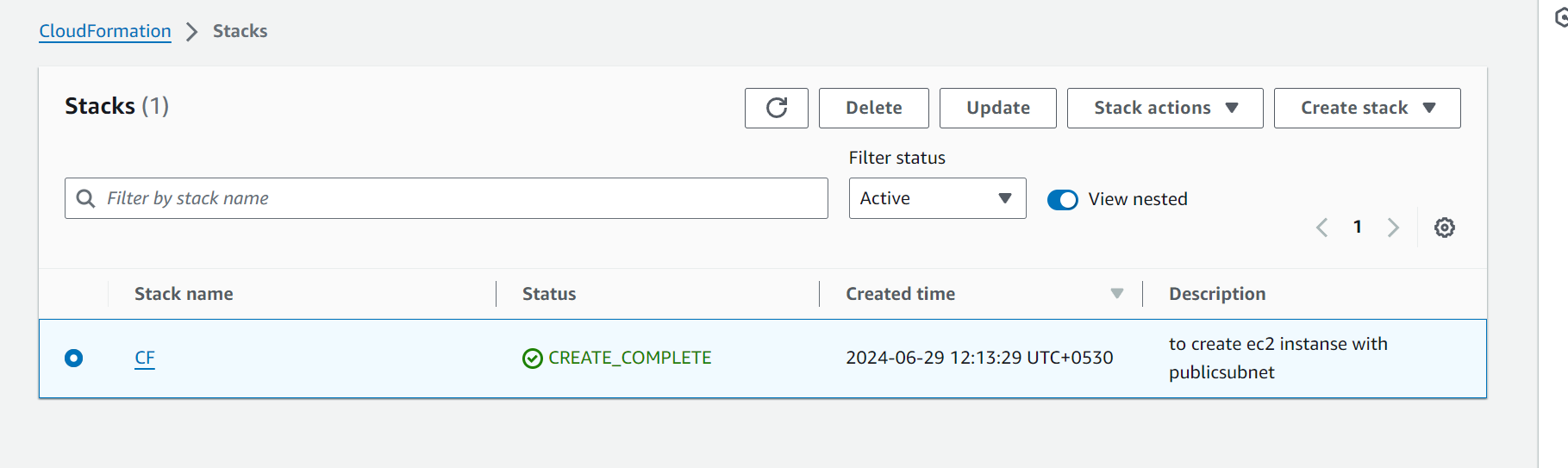
    InstancePublicIp:

       Description: Public IP address of the EC2 instance

       Value: !GetAtt WebTierEC2Instance.PublicIp

CloudFormation stack is created

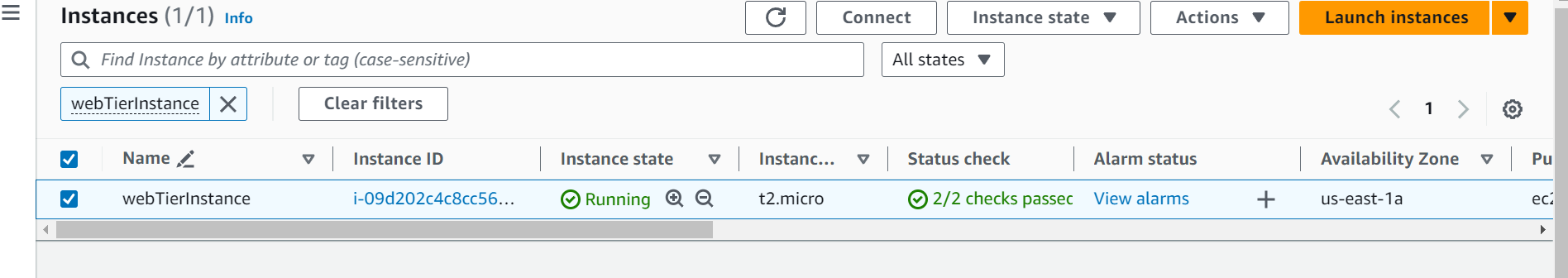
* Go to CloudFormation > Create stack
* Prepare template> Choose an existing template
* Specify template > Upload a template file
* Upload a template file > upload ymal file
* Stack name > CF



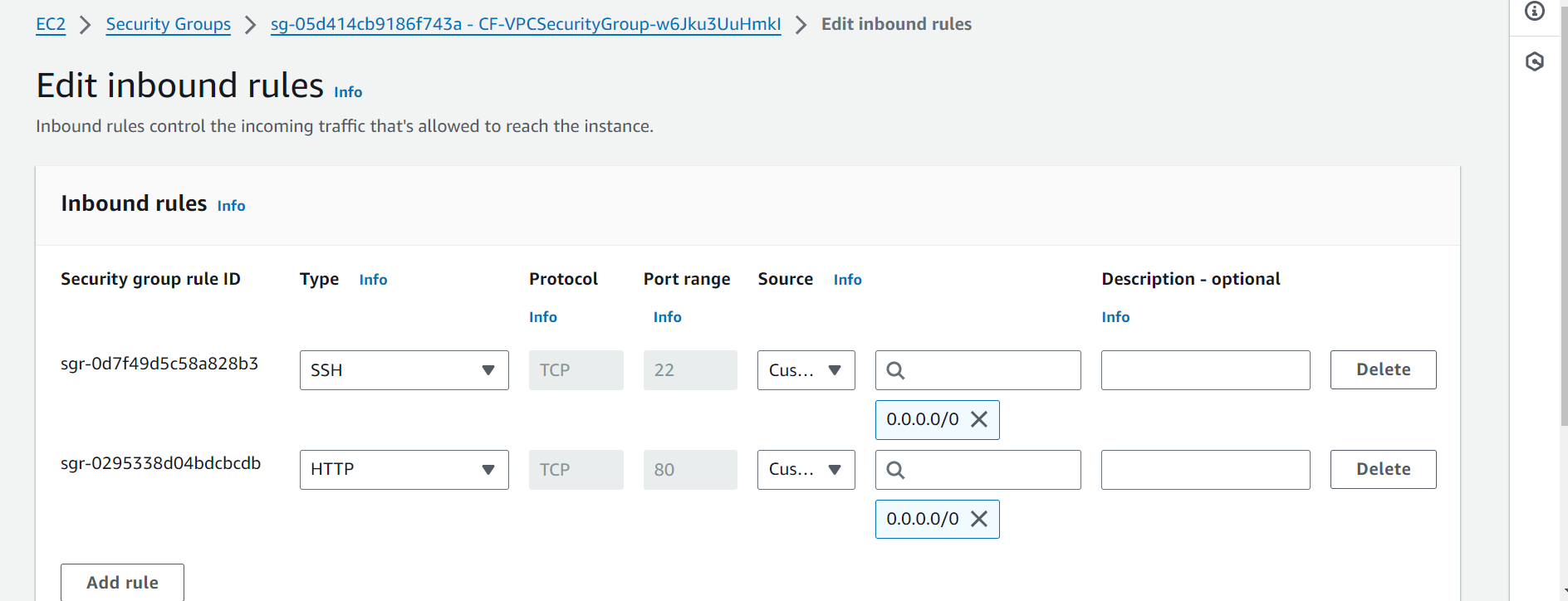
1. **Web tier: Launch an instance in a public subnet and that instance should allow HTTP and SSH from the internet.**

Web Tier instance created, with public subnet and allowing traffic HTTP n SSH from internet

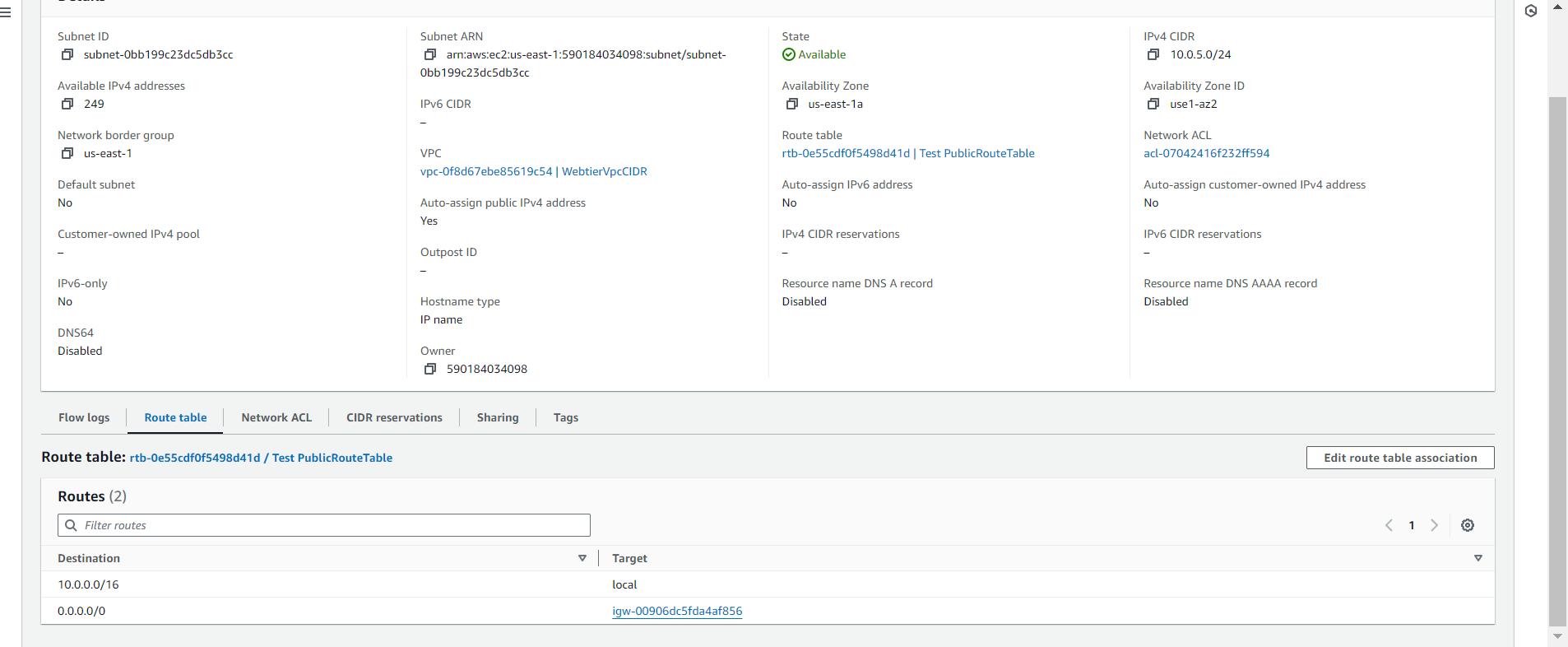
Web Tier instance



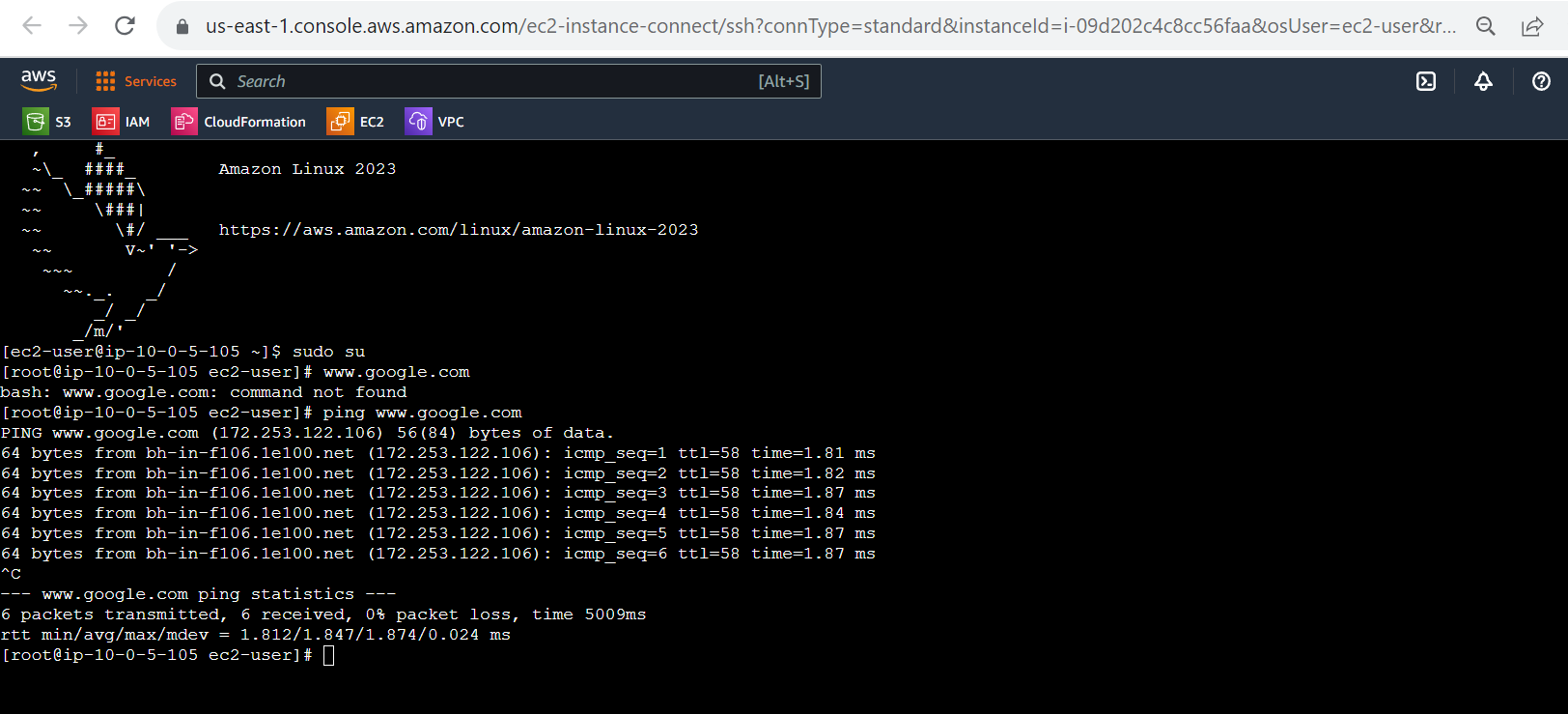
SG

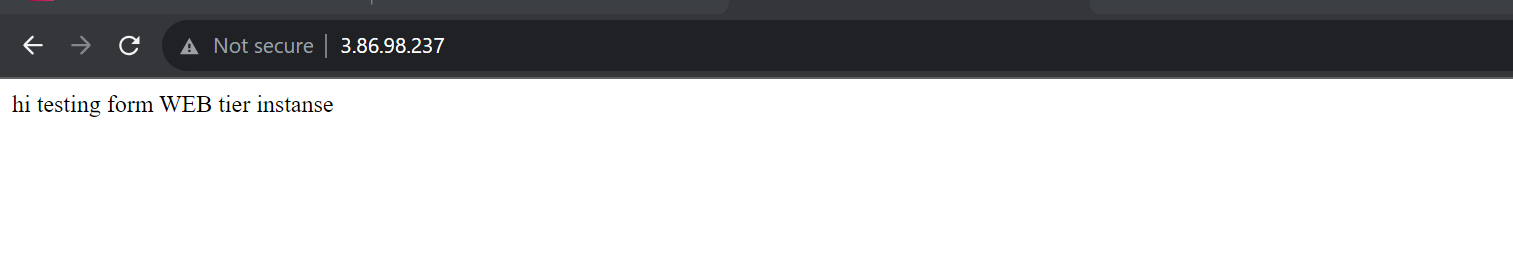


subnet



Able to SSH n http from the above instance vio ssh n http

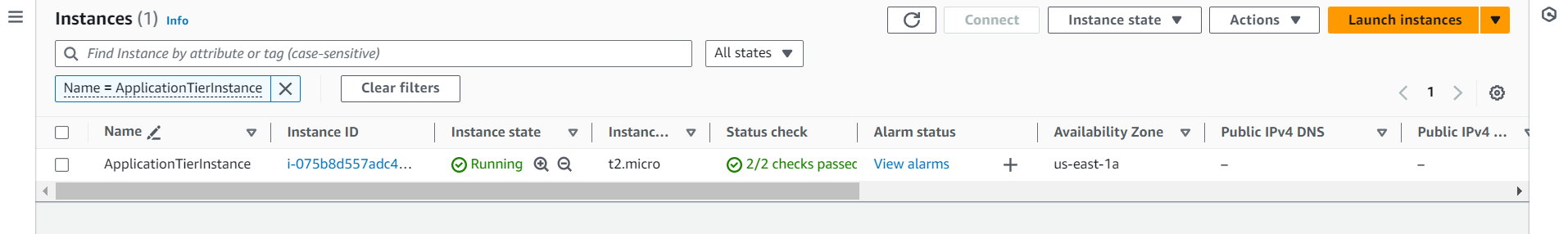




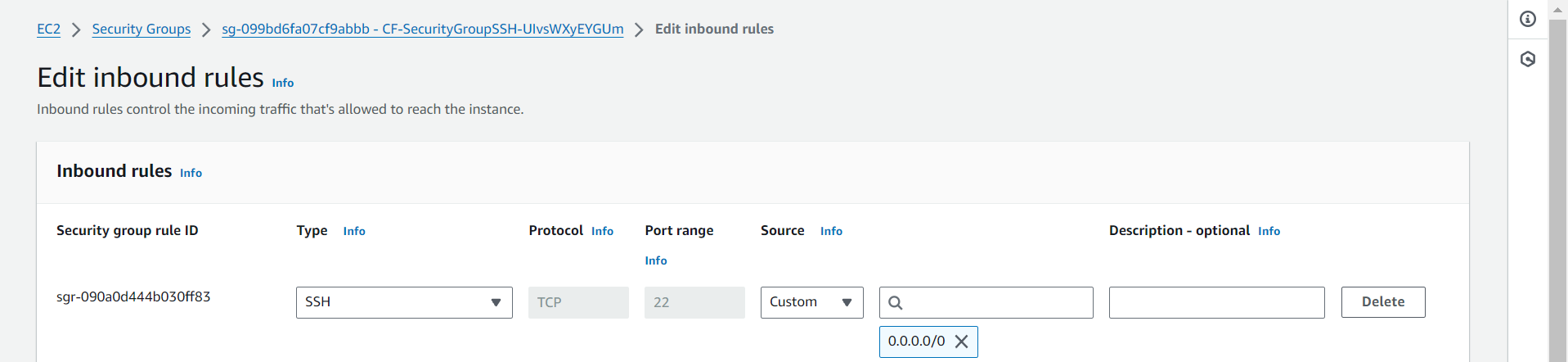
1. **Application tier: Launch an instance in a private subnet of the web tier and it should allow only SSH from the public subnet of Web Tier-3.**

Application tier created with pvt subnet of web tier and it will allow to ssh from web tier instance with the help of NAT GW (created natgw in ymal templete)

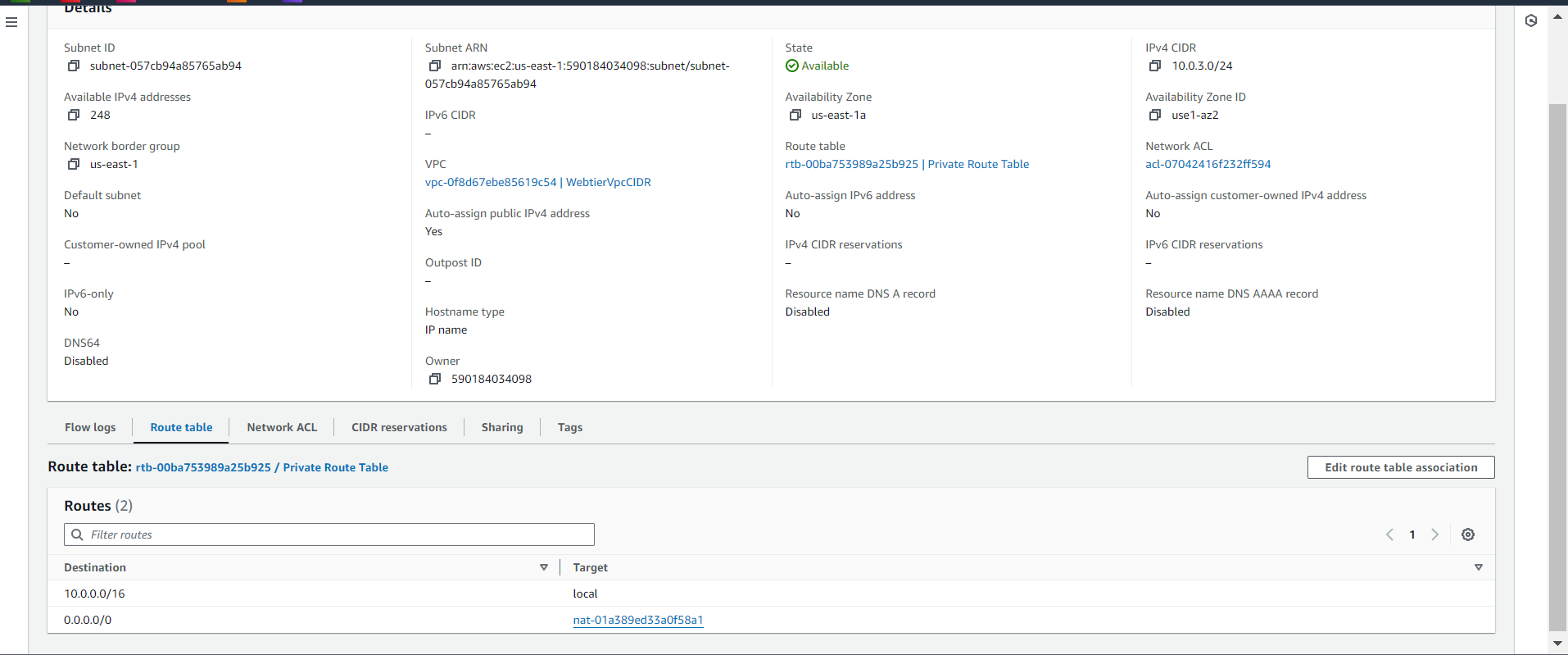
Application tier Instance



SG



Pvt subnet



Connected to application instance via ssh through web instance

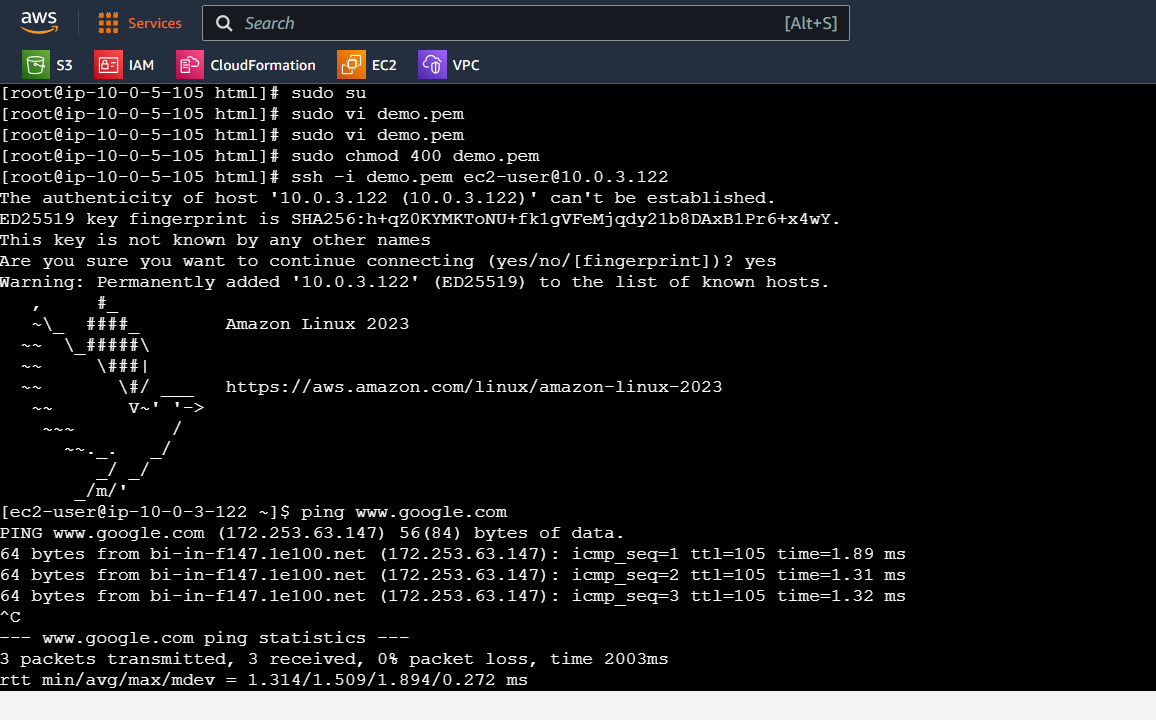
root@ip-10-0-5-105 html]# sudo su

[root@ip-10-0-5-105 html]# sudo vi demo.pem

[root@ip-10-0-5-105 html]# sudo vi demo.pem

[root@ip-10-0-5-105 html]# sudo chmod 400 demo.pem

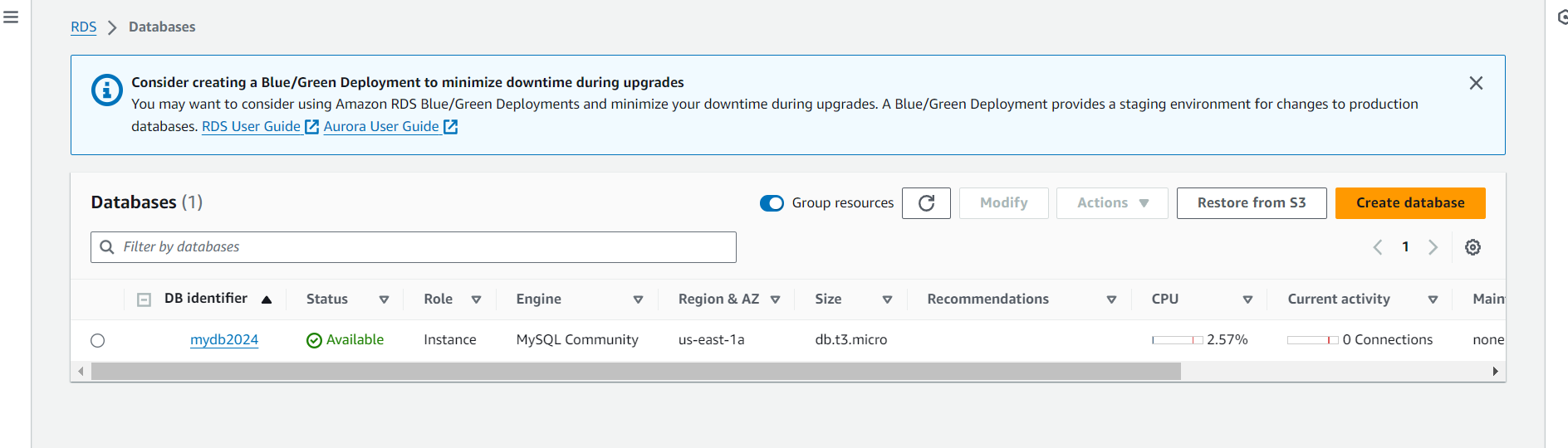
[root@ip-10-0-5-105 html]# ssh -i demo.pem [ec2-user@10.0.3.122](mailto:ec2-user@10.0.3.122)

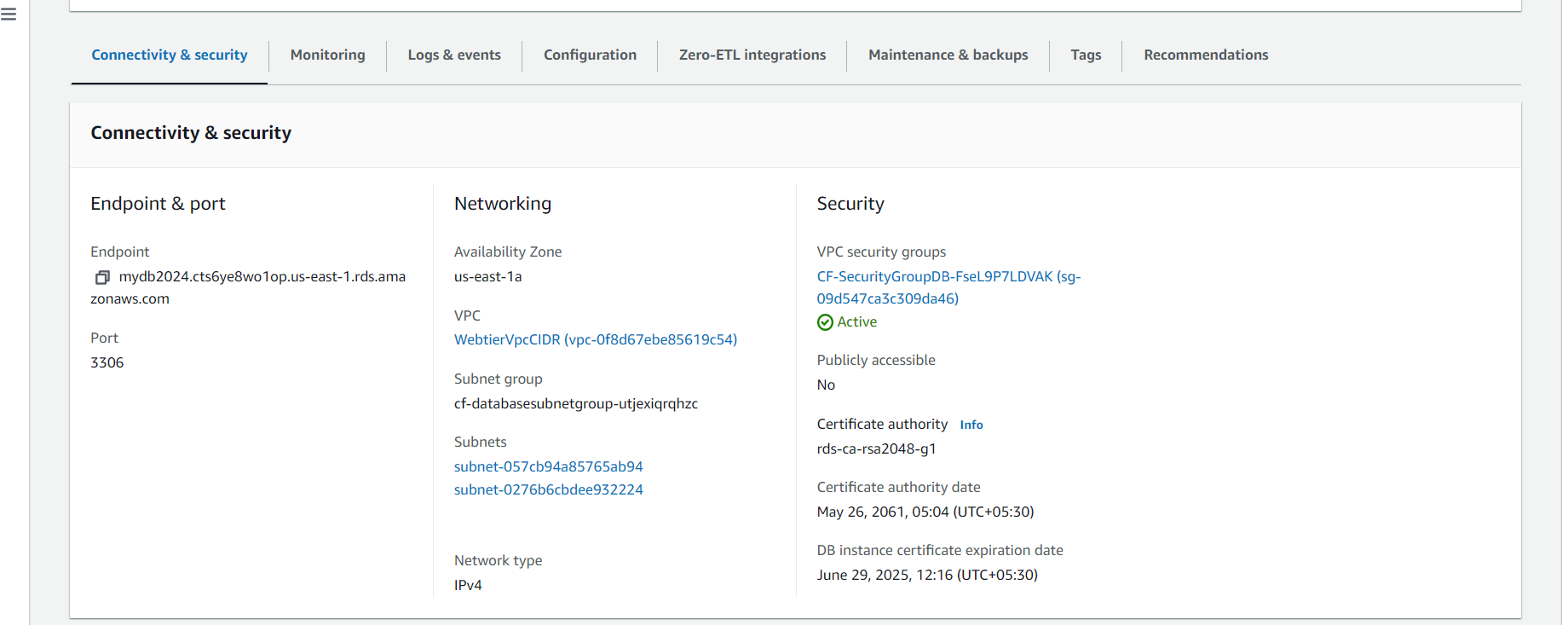


**3. DB tier: Launch an RDS MYSQL instance in a private subnet and it should allow connection on port 3306 only from the private subnet of Application Tier-4.**

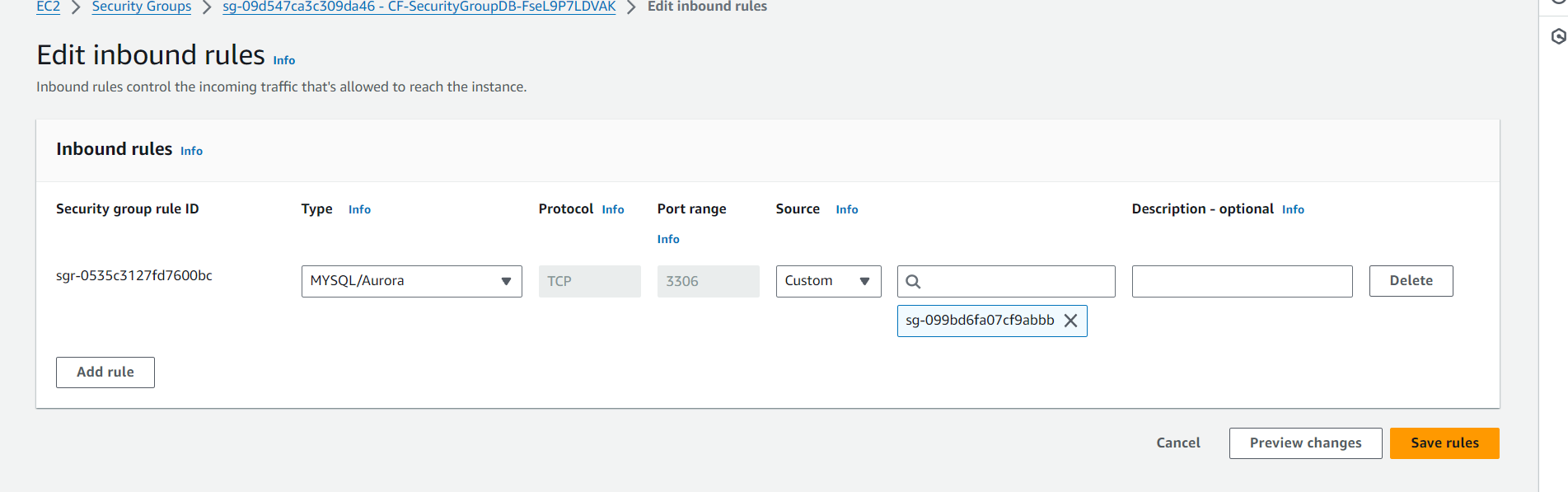
Created DB tier instance to launch RDS MYSQL instance in pvt subnet will allow connection from 3306 via pvt subnet only

DB instance





Sg



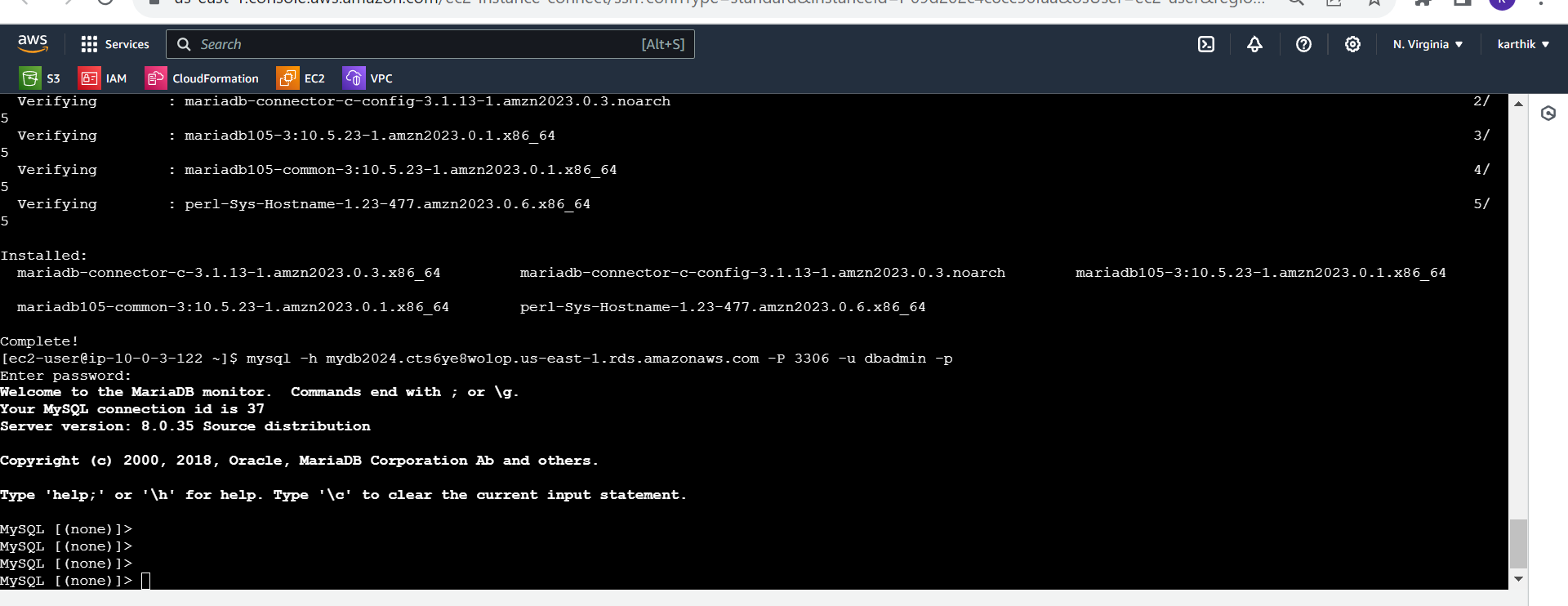
Connect db instance via pvt application instance

To connect db instance we need to install mysql Clint in pvt instance

To install > sudo dnf install mariadb105

To connect > mysql -h (dbendpoint) -P 3306 -u mymasteruser -p

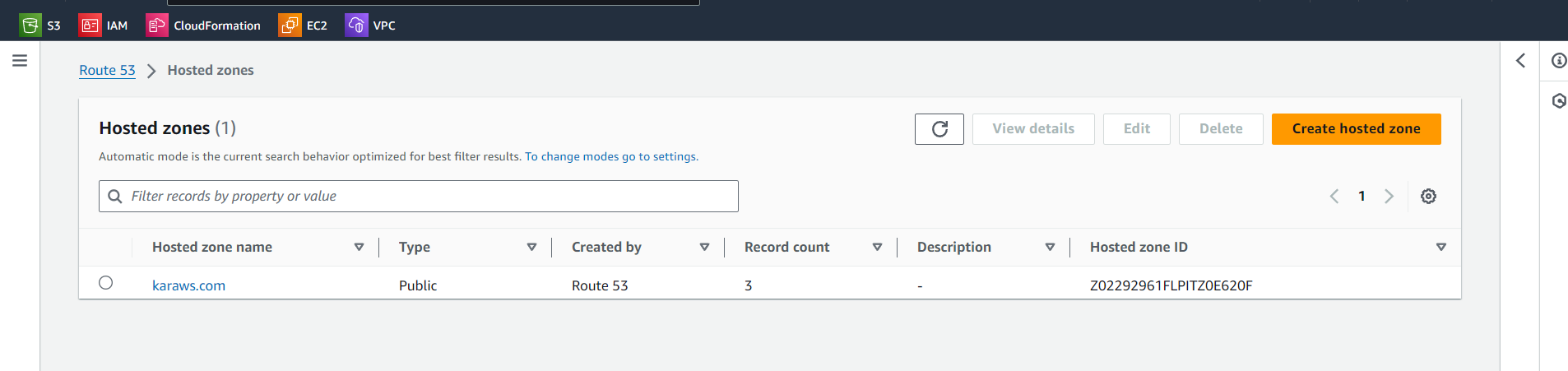
Connected to db instance via pvt application instance



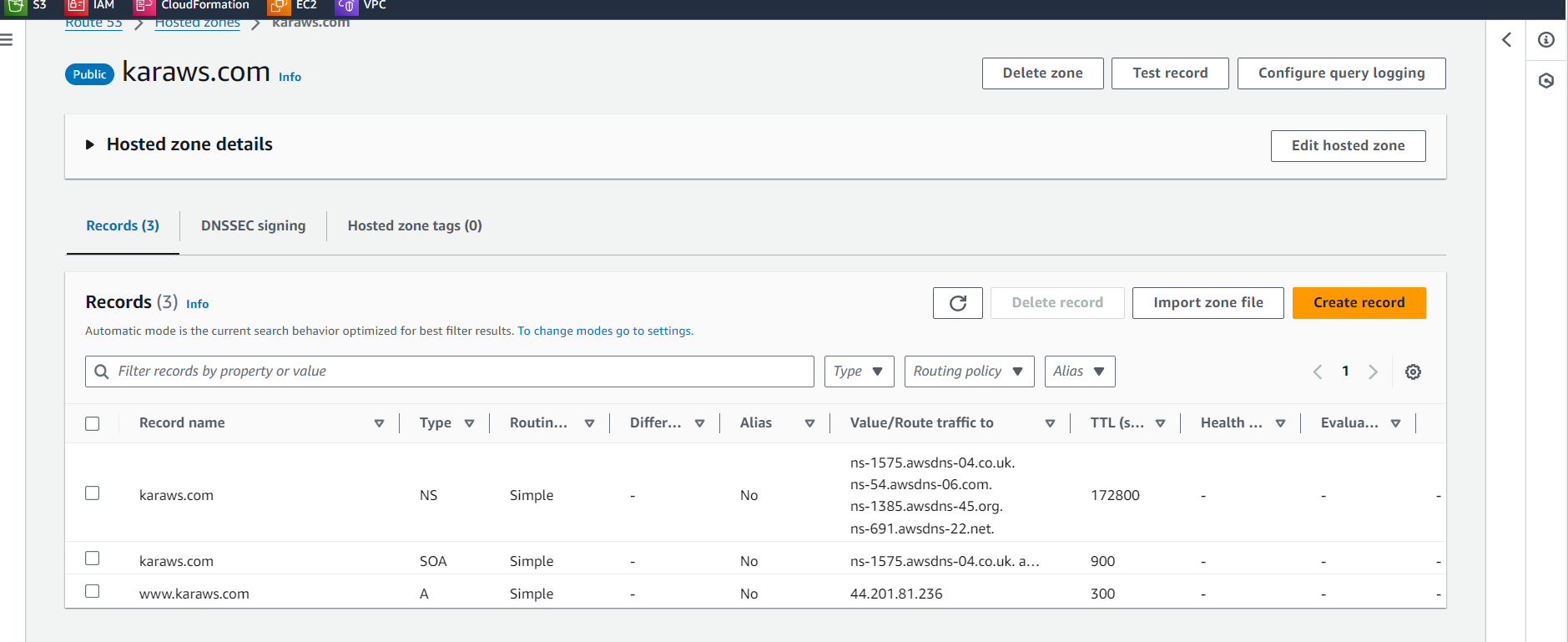
**4.Setup a Route 53 hosted zone and direct traffic to the EC2 instance.**

Created hosted zone and directed traffic to EC2 instance

**Note: steps are recorded due to not having domain name unable to show output**



Created a record to route the traffic to EC2 instance



You have been also asked to propose a solution so that:

1. 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.

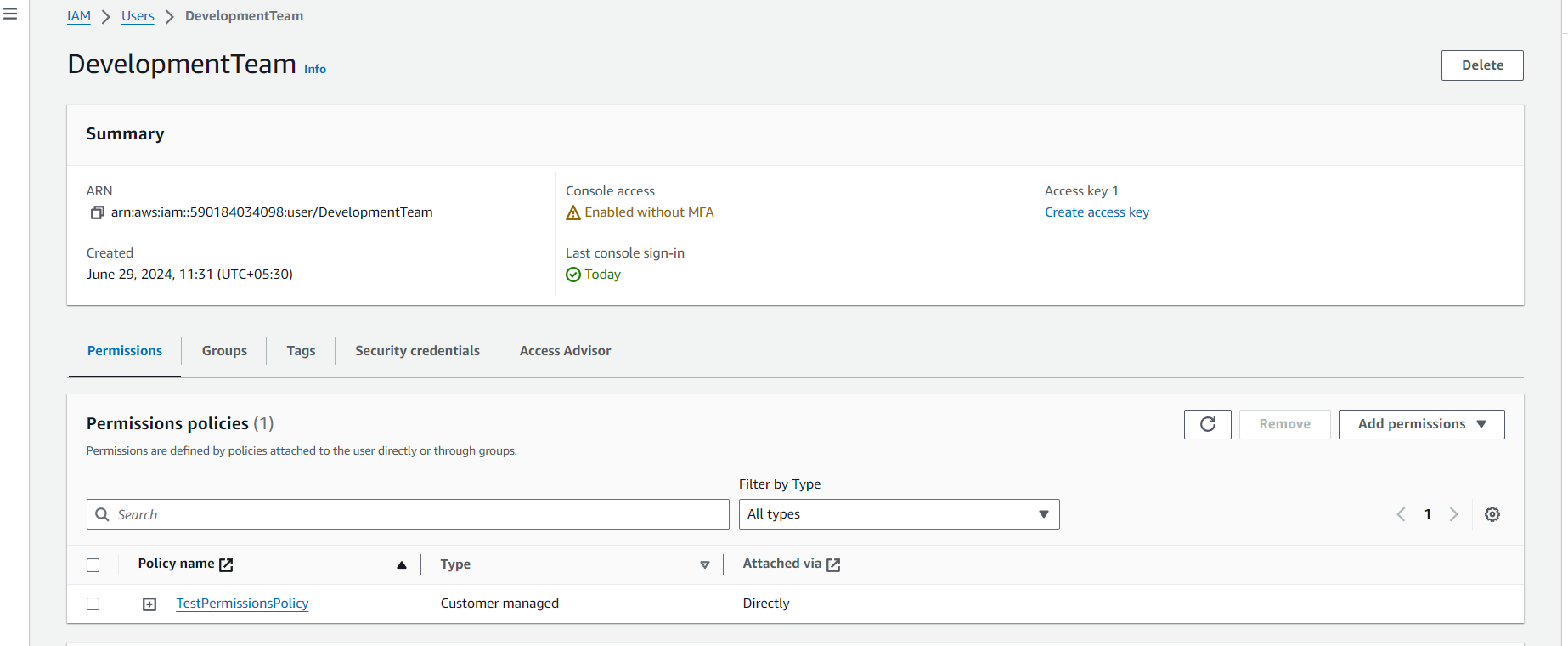
2. Make sure when the development team deletes the stack, RDS DB instances should not be deleted.

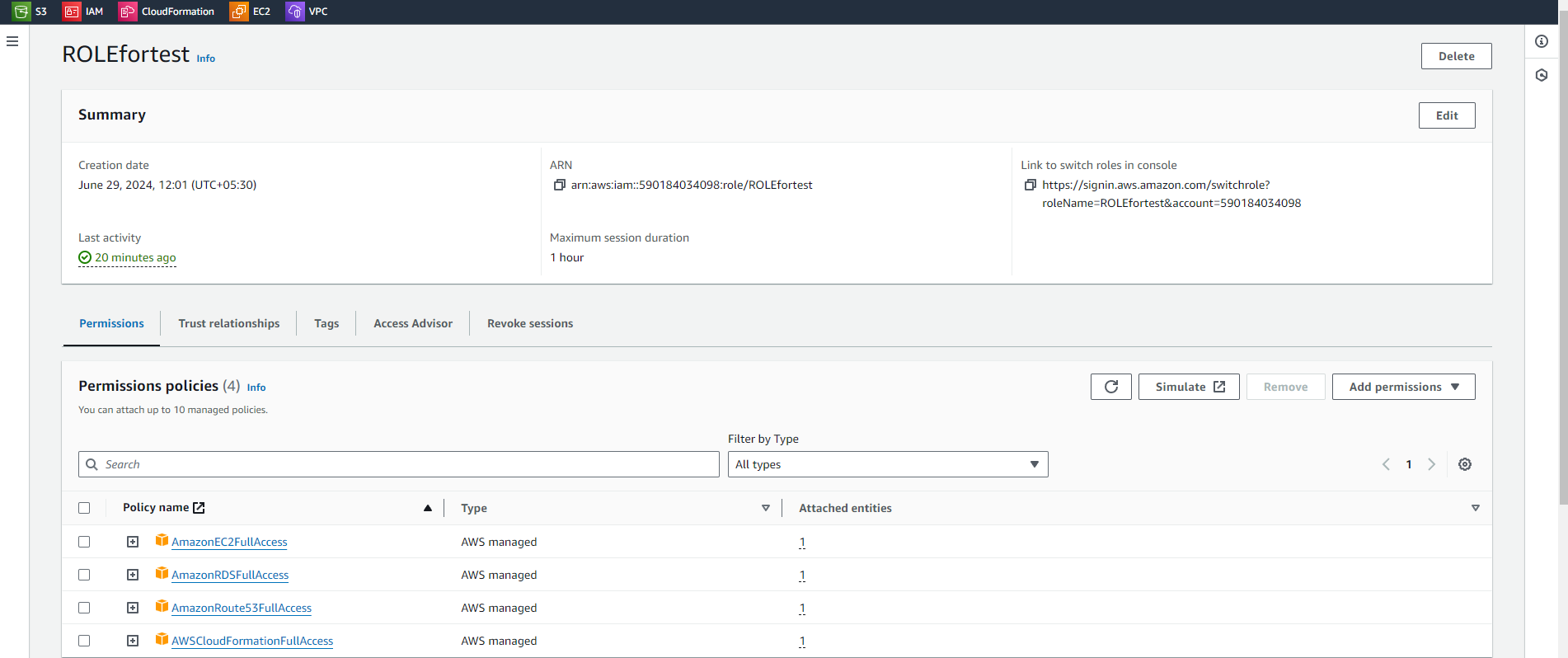
1. **Development team can test their code without having to involve the system admins and can invest their time in testing the code rather than provisioning, configuring and updating the resources needed to test the code.**

To check the code directly for Development team IAM role is created while creating stack with minimum permissions so that they can easily test code without involving in

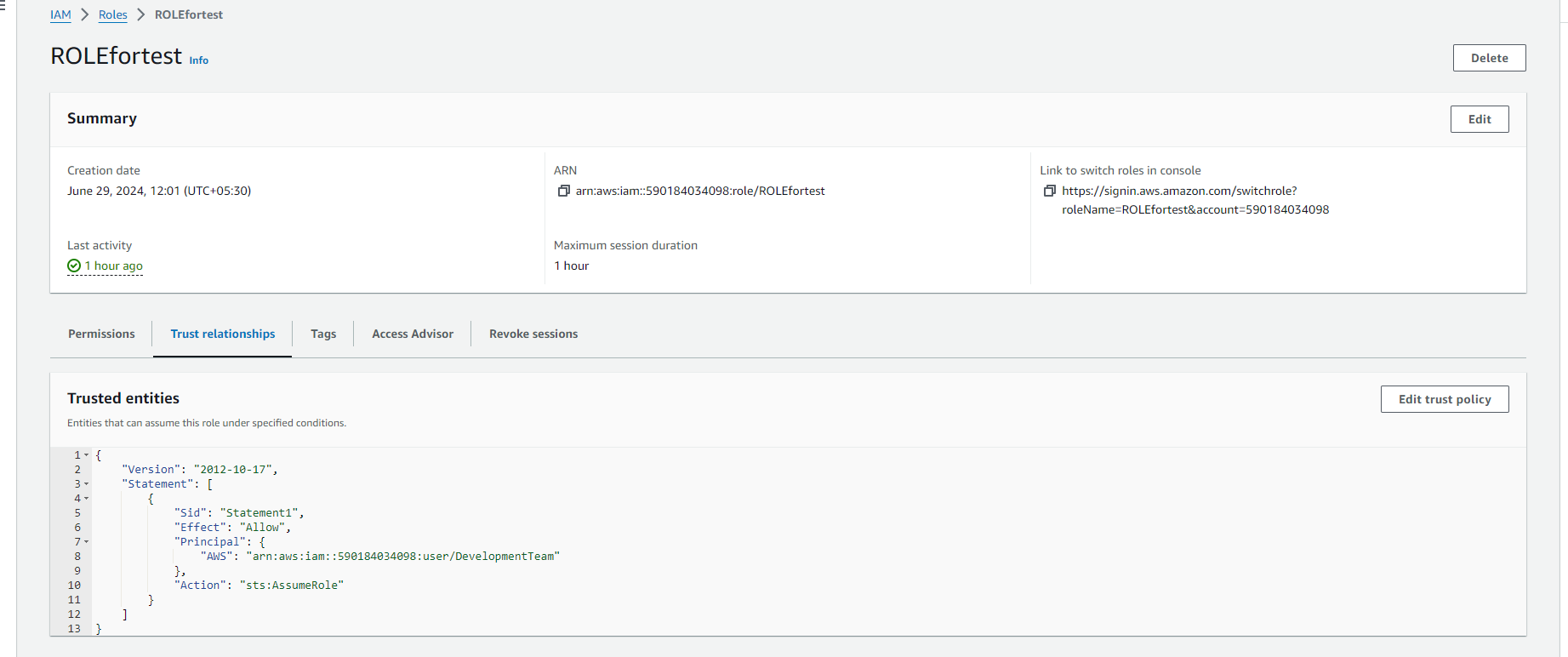
Provisioning n configuring

Created IAM user with name Development Team with minimum access, created policy



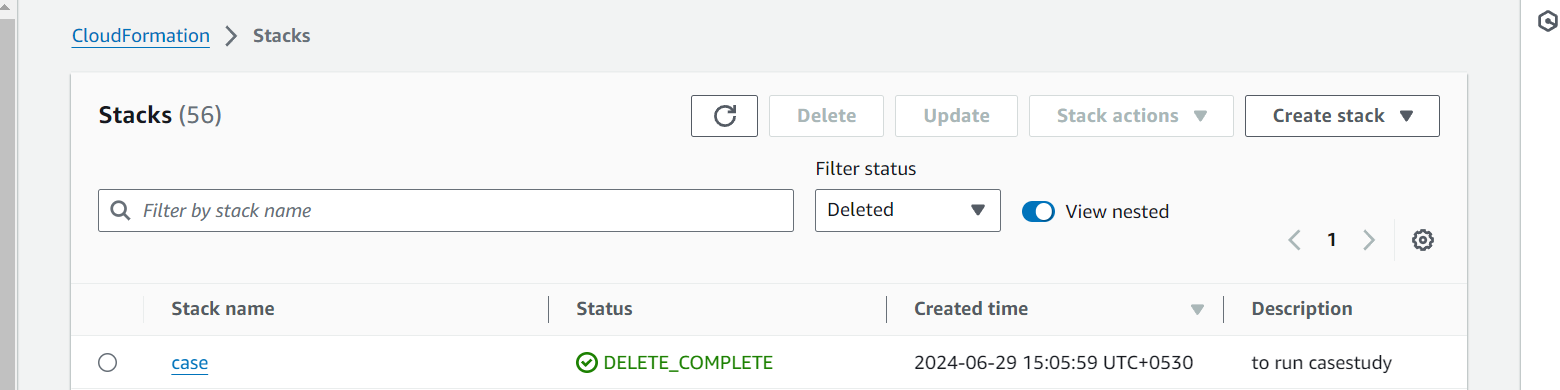


And created IAM role for with trust relation

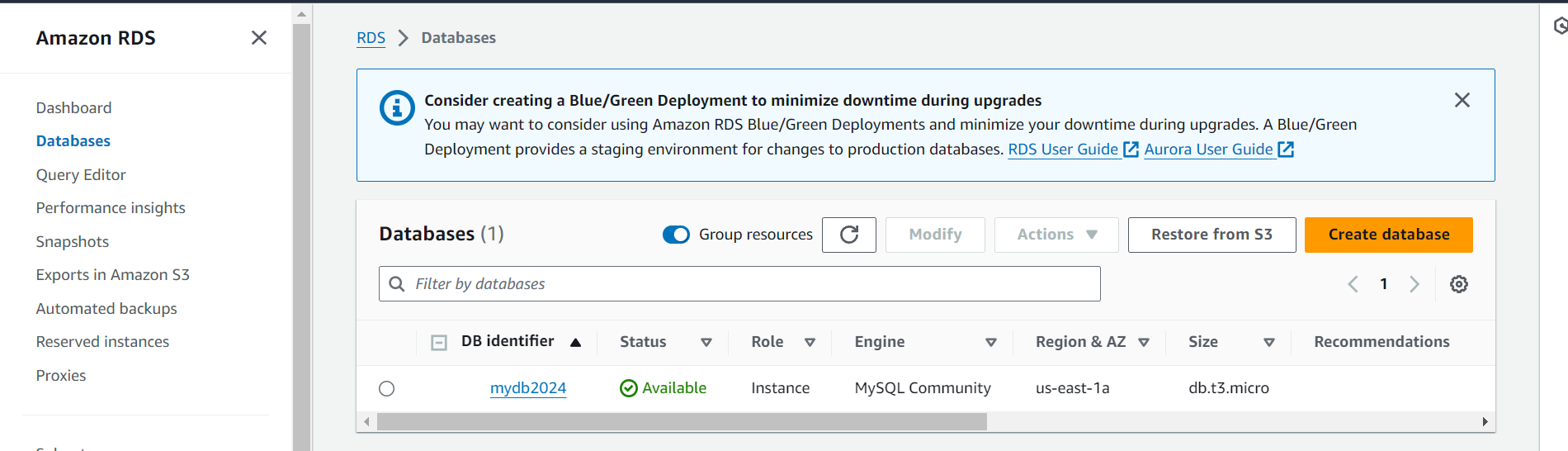


2. Make sure when the development team deletes the stack, RDS DB instances should not be deleted.

Deleted stack by development team , RDS not deleted ( Deletion policy : Retained while creating db instance )



DB instance not deleted



**Stack ymal file attached for refence .**

