Lab Exercise: Deploying a 3-Tier Web Application on AWS Using CloudFormation

Objective:

Learn how to create a CloudFormation template in YAML to set up a VPC with public and private subnets, internet gateway, NAT gateway, security groups, and deploy a 3-tier application with EC2 instances for the web and application layers, and RDS for the database layer. Make sure to parameterize your templates as much as possible.

Prerequisites:

- An AWS account
- Basic understanding of AWS services: VPC, EC2, RDS, subnets, security groups, ELB, and NAT gateway
- Basic knowledge of YAML syntax
- An SSH key pair for EC2 instances

Templates:

1. **Network Resources Template**
    - VPC
    - Subnets (public and private)
    - Internet Gateway
    - NAT Gateway
    - Route Tables
    - Security Groups
2. **EC2 Resources Template**
    - Web Server EC2 Instance
    - Application Server EC2 Instance
    - Application Load Balancer
3. **Database Resources Template**
    - RDS Instance
    - DB Subnet Group

Step 1: Network Resources

Network Resources Template name (`network-resources.yaml`):

Resources:

- VPC
- Public Subnets
  - PublicSubnet1
  - PublicSubnet2
- Private Subnets
  - PrivateSubnet1
  - PrivateSubnet2
- Internet Gateway
  - InternetGateway
- NAT Gateway
  - NatGateway
  - NatGatewayEIP
- Route Tables and Routes
  - PublicRouteTable
    - DefaultRoute
    - PublicSubnet1RouteTableAssociation
    - PublicSubnet2RouteTableAssociation
  - PrivateRouteTable
    - PrivateRoute
    - PrivateSubnet1RouteTableAssociation
    - PrivateSubnet2RouteTableAssociation
- Security Groups
  - WebServerSecurityGroup
  - AppServerSecurityGroup
  - DatabaseSecurityGroup

All resource IDs should be outputted

Outputs:
VPCId
PublicSubnet1Id
PublicSubnet2Id
PrivateSubnet1Id
PrivateSubnet2Id
WebServerSecurityGroupId
AppServerSecurityGroupId
DatabaseSecurityGroupId

Network Resources Template name (`network-resources.yaml`):

Task:

- Write the above YAML in a file named `network-resources.yaml`.
- Use the AWS CLI or Console to create a CloudFormation stack with this template.

Step 2: EC2 Resources

EC2 Resources Template (`ec2-resources.yaml`):

Make sure to create the following parameters and use them within your template

Parameters:

```
VPCId:
PublicSubnet1Id:
PublicSubnet2Id:
PrivateSubnet1Id:
WebServerSecurityGroupId:
AppServerSecurityGroupId:
KeyPairName:
```

Resources:

- Web Server EC2 Instance
  - Should be in PublicSubnet1 and WebServerSecurityGroup
- Application Server EC2 Instance
  - Should be in PrivateSubnet1 and AppServerSecurityGroup
- Application Load Balancer
  - Should use both PublicSubnet1 and PublicSubnet2 and WebServerSecurityGroup
  - MyTargetGroup
  - MyListener

Outputs:

```
WebServerInstanceId:
AppServerInstanceId:
LoadBalancerDNSName:
```

EC2 Resources Template (`ec2-resources.yaml`):

Task:

- Write the above YAML in a file named `ec2-resources.yaml`.

- Use the AWS CLI or Console to create a CloudFormation stack with this template, passing in the required parameters.

Step 3: Database Resources

Resources:

- RDS Instance
- DB Subnet Group

Database Resources Template (`database-resources.yaml`):

Task:

- Write the above YAML in a file named `database-resources.yaml`.
- Use the AWS CLI or Console to create a CloudFormation stack with this template, passing in the required parameters.

Final Task:

- Validate the entire template by updating the stack.
- Verify that all resources are created successfully by checking the AWS Management Console.
- Test the 3-tier application by accessing the Load Balancer DNS name.

Cleanup:

- After the exercise, make sure to delete the CloudFormation stack to avoid incurring unnecessary charges.

Summary:

This lab exercise walks you through creating a VPC with public and private subnets, internet gateway, NAT gateway, security groups, an EC2 instance for the web and application layers, an

RDS instance for the database layer, and an Application Load Balancer. By following these steps, learners will gain hands-on experience with CloudFormation and a comprehensive understanding of deploying a 3-tier application on AWS.