Lab 5: Infrastructure as Code (IaC) with AWS CloudFormation

Lab Title:

Creating and Managing AWS Infrastructure using CloudFormation

Objective:

Introduce students to Infrastructure as Code by using AWS CloudFormation to define and deploy a complete VPC-based web architecture.

Duration:

2 hours

Pre-requisites:

- AWS Free Tier account
- Basic familiarity with AWS services (VPC, EC2, S3)
- Familiarity with JSON or YAML formats

Note: You may use any editor to create yaml files on your local computer, instead of using Cloudshell.

Part A: Introduction to CloudFormation (10 mins)

Key Concepts:

- **Stack**: A collection of AWS resources managed as a single unit
- **Template**: Defines AWS resources in YAML or JSON
- **Change Set**: Previews changes before applying them
- **Nested Stack**: A stack that uses other stack templates as building blocks

Explain: "CloudFormation allows us to model infrastructure in code and deploy reproducibly."

Part B: Launch CloudShell and Prepare Template (15 mins)

1. Open AWS CloudShell

```
cd ~
mkdir lab5-cloudformation && cd lab5-cloudformation
```

2. Create a Sample Template (YAML)

```
cat > simple-s3-stack.yaml
AWSTemplateFormatVersion: '2010-09-09'
Description: Simple S3 Bucket Stack
```

```
Resources:
  MyS3Bucket:
    Type: AWS::S3::Bucket
    Properties:
      BucketName: !Sub "student-demo-bucket-${AWS::AccountId}"
```

Part C: Deploy Stack Using Console (25 mins)

1. Go to AWS Console → CloudFormation

- Click **Create stack** → With new resources (standard)
- Upload your `simple-s3-stack.yaml`
- Stack name: `demo-s3-stack`
- Click **Next** through configuration
- Acknowledge and **Create stack**

2. Verify Resources

- Stack status: `CREATE_COMPLETE`
- Go to **S3 Console** → Confirm bucket is created

Part D: Update the Stack (20 mins)

1. Modify Template

```
cat >> simple-s3-stack.yaml


  MyS3BucketPolicy:
    Type: AWS::S3::BucketPolicy
    DependsOn: MyS3Bucket  # Ensures bucket is created first
    Properties:
      Bucket: !Ref MyS3Bucket
      PolicyDocument:
        Statement:
        - Effect: Allow
          Principal: '*'
          Action: s3:GetObject
          Resource: !Sub "arn:aws:s3:::${MyS3Bucket}/*"
```

2. Update Stack in Console

- Go to CloudFormation → `demo-s3-stack`
- Click **Update** → Replace current template
- Upload the modified file
- Click **Next** and finish

3. Test Public Access

- Upload a file to the bucket
- Test public access via browser (if allowed)

Part E: Additional Stack - EC2 Instance (30 mins)

1. Create EC2 Template (YAML)

```
cat > simple-ec2-stack.yaml
AWSTemplateFormatVersion: '2010-09-09'
Description: Launch a basic EC2 instance in a specific VPC and subnet
Parameters:
  VpcId:
    Type: AWS::EC2::VPC::Id
    Description: Select a VPC for the instance
  SubnetId:
    Type: AWS::EC2::Subnet::Id
    Description: Select a subnet for the instance
Resources:
  MySecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Allow SSH access
      VpcId: !Ref VpcId
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: 0.0.0.0/0  # Not secure in production, for demo only
  MyEC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: ami-0d03cb826412c6b0f  # Amazon Linux 2 in ap-south-1
      InstanceType: t2.micro
      SubnetId: !Ref SubnetId
      SecurityGroupIds:
        - !Ref MySecurityGroup
      Tags:
        - Key: Name
          Value: DemoInstance
```

2. Deploy EC2 Stack via Console

- Go to CloudFormation → **Create stack**
- Upload `simple-ec2-stack.yaml`
- Stack name: `demo-ec2-stack`
- Click **Next** and follow through

3. Verify in EC2 Console

- Go to EC2 → Instances
- Confirm `DemoInstance` is running

⚠ *Note:* Make sure your region supports the AMI and t2.micro type

---

Part F: Nested Stack Example (30 mins)

1. Create Child Template File

```
cat > nested-child-s3.yaml
```

```
AWSTemplateFormatVersion: '2010-09-09'
Description: Child template to create an S3 bucket

Resources:
  ChildS3Bucket:
    Type: AWS::S3::Bucket
    Properties:
      BucketName: !Sub "nested-demo-bucket-${AWS::AccountId}-${AWS::Region}"
```

2. Upload Child Template to S3

```
aws s3 mb s3://my-nested-stack-templates
aws s3 cp nested-child-s3.yaml s3://my-nested-stack-templates/
```

(Note: Replace `my-nested-stack-templates` with a unique bucket name)

3. Create Parent Template File

```
cat > nested-parent.yaml
AWSTemplateFormatVersion: '2010-09-09'
Description: Parent template to call nested S3 template

Resources:
  NestedStack:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: "https://my-nested-stack-templates-3456.s3.ap-south-1.amazonaws.com/nested-child-s3.yaml"
```

4. Deploy Nested Stack

- Go to CloudFormation → **Create stack**
- Upload `nested-parent.yaml`
- Stack name: `nested-s3-parent`
- Click **Next** and finish

5. Verify S3 Bucket Creation

- Go to **S3 Console** → Look for `nested-demo-bucket-<account-id>`

Part G: Delete Stacks (10 mins)

To clean up all resources:

1. Go to CloudFormation
2. Select `demo-s3-stack`, `demo-ec2-stack`, and `nested-s3-parent`
3. Click **Delete**
4. Wait for `DELETE_COMPLETE`

EC2 instance, S3 buckets, and nested stack resources will be removed.