[Task] Automate Static website hosting using AWS CloudFormation

The purpose of this lab is to familiarise yourself with how AWS CloudFormation works

Prerequisites

- Access and download the template (kdr-s3website-cloudfront.yml) to edit

To automate Static website hosting using AWS CloudFormation lab:

In this lab, we will use AWS CloudFormation to create an S3 bucket and CloudFront distribution. For every step please follow the commands on the given template to add resources

1. Create an S3 bucket

```
  rS3WebsiteBucket:
    Type: AWS::S3::Bucket
    DeletionPolicy: Delete
    Properties:
      BucketName: !Sub ${pDomainName}-${AWS::AccountId}
      BucketEncryption:
        ServerSideEncryptionConfiguration:
          - ServerSideEncryptionByDefault:
              SSEAlgorithm: 'AES256'
      LoggingConfiguration:
        DestinationBucketName: !Sub ${pDomainName}-${AWS::AccountId}-
logging
        LogFilePrefix: !Sub ${pDomainName}-${AWS::AccountId}
      Tags:
        - Key: Name
          Value: !Sub ${pDomainName}-${AWS::AccountId}
```

2. Create an S3 bucket policy to allow CloudFront to access objects from the S3 bucket

```
rS3WebsiteBucketPolicy:
  Type: AWS::S3::BucketPolicy
  Properties:
    Bucket: !Ref rS3WebsiteBucket
    PolicyDocument:
      Version: '2012-10-17'
```

```
        Id: 'PolicyForCloudFrontPrivateContent'
        Statement:
          - Action: 's3:GetObject'
            Effect: Allow
            Resource: !Sub 'arn:aws:s3:::${pDomainName}-${AWS::AccountId}/*'
            Principal:
              AWS: !Sub 'arn:aws:iam::cloudfront:user/CloudFront Origin
Access Identity ${rCloudFrontOriginIdentity}'
```

3. Create a logging bucket to store and trace steps/actions made on the bucket

```
  rS3LoggingBucket:
    Type: AWS::S3::Bucket
    DeletionPolicy: Delete
    Properties:
      AccessControl: LogDeliveryWrite
      BucketName: !Sub ${pDomainName}-${AWS::AccountId}-logging
      BucketEncryption:
        ServerSideEncryptionConfiguration:
          - ServerSideEncryptionByDefault:
              SSEAlgorithm: 'AES256'
      LoggingConfiguration:Create
        DestinationBucketName: !Sub ${pDomainName}-${AWS::AccountId}-logging
        LogFilePrefix: !Sub ${pDomainName}-${AWS::AccountId}-logging
      Tags:
        - Key: Name
          Value: !Sub ${pDomainName}-${AWS::AccountId}-logging
```

4. Create a CloudFront origin identity

```
  rCloudFrontOriginIdentity:
    Type: AWS::CloudFront::CloudFrontOriginAccessIdentity
    Properties:
      CloudFrontOriginAccessIdentityConfig:
        Comment: !Sub '${pDomainName} access-identity'
```

5. Create a CloudFront Distribution

```
  rPublicDistribution:
    DependsOn: rCloudFrontOriginIdentity
    Type: AWS::CloudFront::Distribution
    Properties:
      DistributionConfig:
        Comment: !Ref pDomainName
        DefaultCacheBehavior:
          AllowedMethods:
            - GET
            - HEAD
          TargetOriginId: !GetAtt rS3WebsiteBucket.DomainName
          ForwardedValues:
            QueryString: false
            Cookies:
              Forward: none
          ViewerProtocolPolicy: redirect-to-https
        DefaultRootObject: 'index.html'
```

```
        Enabled: true
        HttpVersion: http2
        Logging:
          Bucket: !GetAtt rS3LoggingBucket.DomainName
          IncludeCookies: false
          Prefix: !Sub '${pDomainName}/cloudfront'
        Origins:
          - DomainName: !GetAtt rS3WebsiteBucket.DomainName
            Id: !GetAtt rS3WebsiteBucket.DomainName
            S3OriginConfig:
              OriginAccessIdentity: !Sub 'origin-access-
identity/cloudfront/${rCloudFrontOriginIdentity}'
        ViewerCertificate:
          CloudFrontDefaultCertificate: true
```

Task results

From your **Outputs** on you will access your CloudFrontDomainName value to access your website

Additional resources

- [Concept] AWS ramp-up guide