```
AWSTemplateFormatVersion: "2010-09-09"
Description: "Secure deployment of EC2 with encrypted EBS, RDS (MariaDB in private subnet with multi-AZ support), S3 bucket, Secrets Manager, network segmentation, NACLs, SG rules, HTTPS enforcement, and strict S3 access control."

Parameters:
  KeyPairName:
    Description: "Name of an existing EC2 KeyPair to enable SSH access"
    Type: "AWS::EC2::KeyPair::KeyName"
    Default: test1

  InstanceName:
    Description: "Name of the EC2 instance"
    Type: String
    Default: "SecureEC2Instance"

  DBUser:
    Description: "Username for RDS database"
    Type: String
    Default: "admin"

  DBName:
    Description: "Name of the database"
    Type: String
    Default: "juiceshop"

  AMIId:
    Description: "The AMI ID to be used for the EC2 instance. Default is Amazon Linux 2023."
    Type: "AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>"
    Default: "/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-default-x86_64"

Resources:

  KMSKey:
    Type: "AWS::KMS::Key"
    Properties:
      Description: "KMS Key for encryption"
```

```
    KeyPolicy:
      Version: "2012-10-17"
      Statement:
        - Effect: Allow
          Principal:
            AWS: !Sub "arn:aws:iam::${AWS::AccountId}:root"
          Action: "kms:*"
          Resource: "*"

  SecretForDBPassword:
    Type: AWS::SecretsManager::Secret
    Properties:
      Name: "DBPasswordSecretv1"
      Description: "Auto-generated DB password for RDS"
      GenerateSecretString:
        SecretStringTemplate: "{}"
        GenerateStringKey: "password"
        PasswordLength: 16
        ExcludeCharacters: "\"@/"

  EC2S3AccessRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: EC2S3AccessRole
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Allow
            Principal:
              Service: ec2.amazonaws.com
            Action: sts:AssumeRole
      Policies:
        - PolicyName: AllowS3Access
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              - Effect: Allow
```

```
            Action:
              - s3:GetObject
              - s3:ListBucket
            Resource:
              - !Sub "${S3SecureBucket.Arn}"
              - !Sub "${S3SecureBucket.Arn}/*"

  EC2InstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Roles:
        - !Ref EC2S3AccessRole
      InstanceProfileName: EC2S3InstanceProfile

  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: "10.0.0.0/16"
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: "SecureVPC"

  PublicSubnetA:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      CidrBlock: "10.0.1.0/24"
      AvailabilityZone: !Select [0, !GetAZs ""]
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: "PublicSubnetA"

  PrivateSubnetA:
    Type: AWS::EC2::Subnet
```

```
  Properties:
    VpcId: !Ref VPC
    CidrBlock: "10.0.2.0/24"
    AvailabilityZone: !Select [0, !GetAZs ""]
    MapPublicIpOnLaunch: false
    Tags:
      - Key: Name
        Value: "PrivateSubnetA"

PrivateSubnetB:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref VPC
    CidrBlock: "10.0.3.0/24"
    AvailabilityZone: !Select [1, !GetAZs ""]
    MapPublicIpOnLaunch: false
    Tags:
      - Key: Name
        Value: "PrivateSubnetB"

DBSubnetGroup:
  Type: AWS::RDS::DBSubnetGroup
  Properties:
    DBSubnetGroupDescription: "Private subnets for RDS across 2 AZs"
    SubnetIds:
      - !Ref PrivateSubnetA
      - !Ref PrivateSubnetB
    Tags:
      - Key: Name
        Value: RDSPrivateSubnetGroup

RDSInstance:
  Type: AWS::RDS::DBInstance
  Properties:
    DBInstanceIdentifier: "secure-db"
    AllocatedStorage: 20
    DBInstanceClass: db.t3.micro
```

```
    Engine: mariadb
    MasterUsername: !Ref DBUser
    MasterUserPassword:
      Fn::Sub: "{{resolve:secretsmanager:${SecretForDBPassword}::password}}"
    DBName: !Ref DBName
    DBSubnetGroupName: !Ref DBSubnetGroup
    VPCSecurityGroups:
      - !Ref RDSSecurityGroup
    StorageEncrypted: true
    KmsKeyId: !Ref KMSKey
    BackupRetentionPeriod: 7
    PubliclyAccessible: false
    MultiAZ: true
    DeletionProtection: false
    Tags:
      - Key: Name
        Value: SecureRDSInstance

  RDSSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: "Allow access to RDS from EC2 only (ingress and egress)"
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 3306
          ToPort: 3306
          SourceSecurityGroupId: !Ref EC2SecurityGroup
      SecurityGroupEgress:
        - IpProtocol: tcp
          FromPort: 3306
          ToPort: 3306
          DestinationSecurityGroupId: !Ref EC2SecurityGroup
      Tags:
        - Key: Name
          Value: RDSSecurityGroup
```

```yaml
EC2SecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: "Allow SSH and HTTP access to EC2 instance"
    VpcId: !Ref VPC
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: 100.15.116.115
      - IpProtocol: tcp
        FromPort: 80
        ToPort: 80
        CidrIp: 0.0.0.0/0
    Tags:
      - Key: Name
        Value: EC2SecurityGroup

S3SecureBucket:
  Type: AWS::S3::Bucket
  Properties:
    BucketEncryption:
      ServerSideEncryptionConfiguration:
        - ServerSideEncryptionByDefault:
            SSEAlgorithm: aws:kms
            KMSMasterKeyID: !Ref KMSKey
    AccessControl: Private
    Tags:
      - Key: Name
        Value: EncryptedSnapshotStorage

S3BucketPolicy:
  Type: AWS::S3::BucketPolicy
  Properties:
    Bucket: !Ref S3SecureBucket
    PolicyDocument:
      Version: "2012-10-17"
```

```
      Statement:
        - Sid: AllowEC2RoleAccess
          Effect: Allow
          Principal: "*"
          Action:
            - s3:GetObject
            - s3:ListBucket
          Resource:
            - !Sub "${S3SecureBucket.Arn}"
            - !Sub "${S3SecureBucket.Arn}/*"
          Condition:
            StringEquals:
              aws:PrincipalArn: !Sub "arn:aws:iam::${AWS::AccountId}:role/EC2S3AccessRole"
        - Sid: DenyAllOthers
          Effect: Deny
          Principal: "*"
          Action: "s3:*"
          Resource:
            - !Sub "${S3SecureBucket.Arn}"
            - !Sub "${S3SecureBucket.Arn}/*"
          Condition:
            StringNotEquals:
              aws:PrincipalArn: !Sub "arn:aws:iam::${AWS::AccountId}:role/EC2S3AccessRole"

  EC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: t2.micro
      UserData:
        Fn::Base64: !Sub |
          #!/bin/bash
          yum update -y
          yum install -y mariadb
          cd /home/ec2-user
          curl https://truststore.pki.rds.amazonaws.com/global/global-bundle.pem -o global-bundle.pem
          cat > /home/ec2-user/.my.cnf <<EOF
          [client]
```

```
        user=${DBUser}
        ssl-ca=/home/ec2-user/global-bundle.pem
        ssl-mode=REQUIRED
        EOF
        chown ec2-user:ec2-user /home/ec2-user/.my.cnf
        chmod 600 /home/ec2-user/.my.cnf
    KeyName: !Ref KeyPairName
    ImageId: !Ref AMIId
    SubnetId: !Ref PublicSubnetA
    IamInstanceProfile: !Ref EC2InstanceProfile
    SecurityGroupIds:
     - !Ref EC2SecurityGroup
    BlockDeviceMappings:
     - DeviceName: "/dev/xvda"
       Ebs:
         VolumeSize: 8
         VolumeType: gp2
         Encrypted: true
         KmsKeyId: !Ref KMSKey
    Tags:
     - Key: Name
       Value: !Ref InstanceName

PublicNACL:
  Type: AWS::EC2::NetworkAcl
  Properties:
    VpcId: !Ref VPC
    Tags:
     - Key: Name
       Value: PublicNACL

PrivateNACL:
  Type: AWS::EC2::NetworkAcl
  Properties:
    VpcId: !Ref VPC
    Tags:
     - Key: Name
```

```
      Value: PrivateNACL

  PublicNACLIngress:
    Type: AWS::EC2::NetworkAclEntry
    Properties:
      NetworkAclId: !Ref PublicNACL
      RuleNumber: 100
      Protocol: 6
      RuleAction: allow
      Egress: false
      CidrBlock: 0.0.0.0/0
      PortRange:
        From: 80
        To: 80

  PublicNACLEgress:
    Type: AWS::EC2::NetworkAclEntry
    Properties:
      NetworkAclId: !Ref PublicNACL
      RuleNumber: 100
      Protocol: 6
      RuleAction: allow
      Egress: true
      CidrBlock: 0.0.0.0/0
      PortRange:
        From: 1024
        To: 65535

  PublicSubnetNACLAssociation:
    Type: AWS::EC2::SubnetNetworkAclAssociation
    Properties:
      SubnetId: !Ref PublicSubnetA
      NetworkAclId: !Ref PublicNACL

  PrivateNACLIngress:
    Type: AWS::EC2::NetworkAclEntry
    Properties:
```

```
      NetworkAclId: !Ref PrivateNACL
      RuleNumber: 100
      Protocol: 6
      RuleAction: allow
      Egress: false
      CidrBlock: 10.0.1.0/24
      PortRange:
        From: 3306
        To: 3306

  PrivateNACLEgress:
    Type: AWS::EC2::NetworkAclEntry
    Properties:
      NetworkAclId: !Ref PrivateNACL
      RuleNumber: 100
      Protocol: 6
      RuleAction: allow
      Egress: true
      CidrBlock: 10.0.1.0/24
      PortRange:
        From: 1024
        To: 65535

  PrivateSubnetNACLAssociationA:
    Type: AWS::EC2::SubnetNetworkAclAssociation
    Properties:
      SubnetId: !Ref PrivateSubnetA
      NetworkAclId: !Ref PrivateNACL

  PrivateSubnetNACLAssociationB:
    Type: AWS::EC2::SubnetNetworkAclAssociation
    Properties:
      SubnetId: !Ref PrivateSubnetB
      NetworkAclId: !Ref PrivateNACL
```