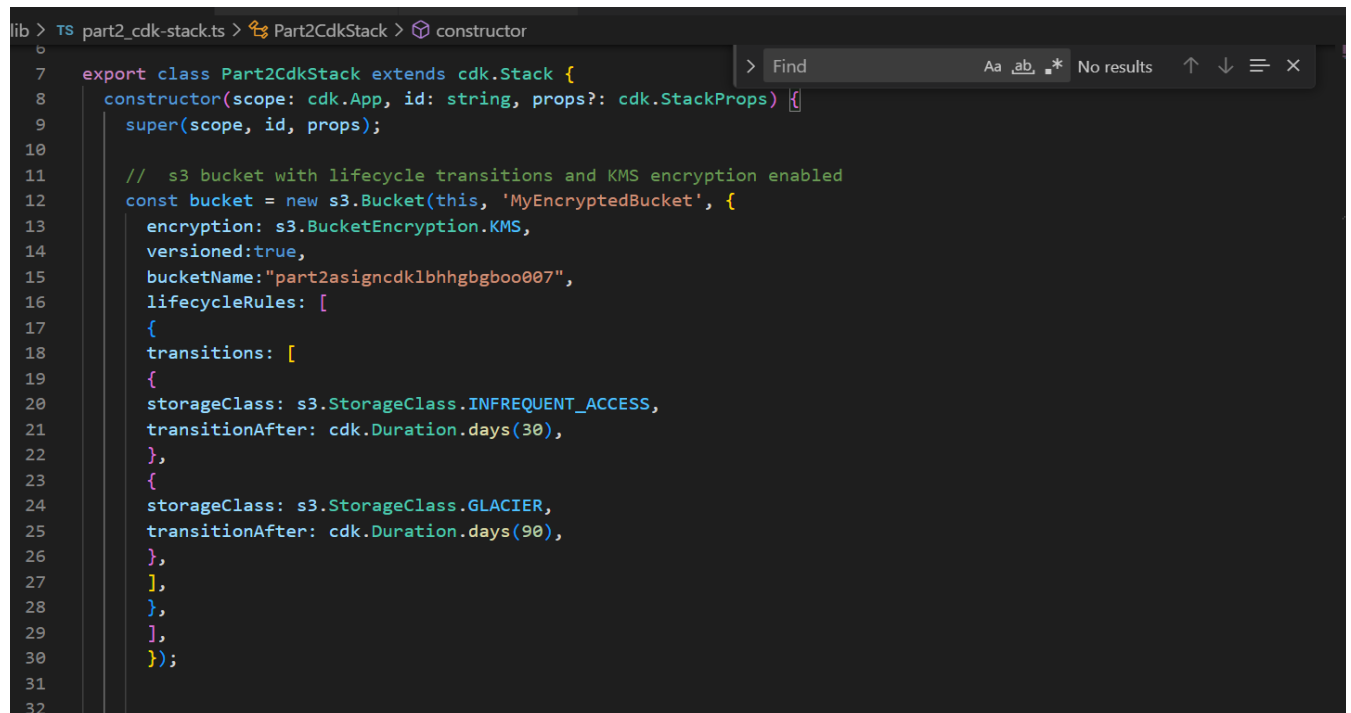


Use the CDK environment created from the Must Do section of Part 1 and create a sample CDK Project.



```
lib > TS part2_cdk-stack.ts > Part2CdkStack > constructor
6
7 export class Part2CdkStack extends cdk.Stack {
8   constructor(scope: cdk.App, id: string, props?: cdk.StackProps) {
9     super(scope, id, props);
10
11     // s3 bucket with lifecycle transitions and KMS encryption enabled
12     const bucket = new s3.Bucket(this, 'MyEncryptedBucket', {
13       encryption: s3.BucketEncryption.KMS,
14       versioned: true,
15       bucketName: "part2asigndck1bhghbgboo007",
16       lifecycleRules: [
17         {
18           transitions: [
19             {
20               storageClass: s3.StorageClass.INFREQUENT_ACCESS,
21               transitionAfter: cdk.Duration.days(30),
22             },
23             {
24               storageClass: s3.StorageClass.GLACIER,
25               transitionAfter: cdk.Duration.days(90),
26             },
27           ],
28         },
29       ],
30     });
31
32
```

Step 2: Create a CDK code to deploy an EC2 instance using Amazon Linux v2 AML.

->First created VPC where we will launch instance-

```
const vpc = new ec2.Vpc(this, 'my-cdk-vpc', {
  cidr: '10.0.0.0/16',
  natGateways: 0,
  subnetConfiguration: [
    {name: 'public', cidrMask: 24, subnetType: ec2.SubnetType.PUBLIC},
  ],
});
```

-> creating security groups

```
const webserverSG = new ec2.SecurityGroup(this, 'webserver-sg', {
  vpc,
  allowAllOutbound: true,
});

webserverSG.addIngressRule(
  ec2.Peer.anyIpv4(),
  ec2.Port.tcp(22),
  'allow SSH access from anywhere',
);

webserverSG.addIngressRule(
  ec2.Peer.anyIpv4(),
  ec2.Port.tcp(80),
  'allow HTTP traffic from anywhere',
);

webserverSG.addIngressRule(
  ec2.Peer.anyIpv4(),
  ec2.Port.tcp(443),
  'allow HTTPS traffic from anywhere',
);
```

-> role for EC2 instance:

```
const webserverRole = new iam.Role(this, 'webserver-role', {
  assumedBy: new iam.ServicePrincipal('ec2.amazonaws.com'),
  managedPolicies: [
    iam.ManagedPolicy.fromAwsManagedPolicyName('AmazonS3ReadOnlyAccess'),
  ],
});
```

->EC2 instance with using Amazon linux

```
const ec2Instance = new ec2.Instance(this, 'ec2-instance', {
  vpc,
  vpcSubnets: {
    subnetType: ec2.SubnetType.PUBLIC,
  },
  role: webserverRole,
  securityGroup: webserverSG,
  instanceType: ec2.InstanceType.of(
    ec2.InstanceClass.BURSTABLE2,
    ec2.InstanceSize.MICRO,
  ),
  machineImage: new ec2.AmazonLinuxImage({
    generation: ec2.AmazonLinuxGeneration.AMAZON_LINUX_2,
  }),
  keyName: 'assignment',
});
```

Step 3: Using CDK code create a custom config set and have a config to install nginx server on the Instance and also added config to update the host name of the Instance.

```
#!/bin/bash
yum update -y
sudo su

amazon-linux-extras install -y nginx1
systemctl start nginx
systemctl enable nginx
sudo hostnamectl set-hostname part2assign4
sudo systemctl restart systemd-hostnamed
```

Step 4: Use the s3 Bucket from the Must Do Section of CDK Part 1 and set up an optional Integration by granting read-write permissions to the Bucket (Once we log into the Instance, we should be able to access the objects, and also able to upload the objects to the Bucket)

```
bucket.grantReadWrite(ec2Instance);
```

->Instance launched

EC2 > Instances > i-04bb294a44bd30c3b

Instance summary for i-04bb294a44bd30c3b (Part2CdkStack/ec2-instance) [Info](#)

Updated less than a minute ago

[Refresh](#) [Connect](#) [Instance state ▼](#) [Actions ▼](#)

Instance ID i-04bb294a44bd30c3b (Part2CdkStack/ec2-instance)	Public IPv4 address 13.115.84.5 open address	Private IPv4 addresses 10.0.0.161
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-13-115-84-5.ap-northeast-1.compute.amazonaws.com open address
Hostname type IP name: ip-10-0-0-161.ap-northeast-1.compute.internal	Private IP DNS name (IPv4 only) ip-10-0-0-161.ap-northeast-1.compute.internal	
Answer private resource DNS name -	Instance type t2.micro	Elastic IP addresses -
Auto-assigned IP address 13.115.84.5 [Public IP]	VPC ID vpc-09a4c7b57b0f9122a (Part2CdkStack/my-cdk-vpc) open address	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. Learn more
IAM Role Part2CdkStack-webserverroleDB0308B6-SA7QJREYP9LQ open address	Subnet ID subnet-0539229c5772bf0f8 (Part2CdkStack/my-cdk-vpc/publicSubnet1) open address	Auto Scaling Group name -
IMDSv2 Optional		

-> nginx server

```
[ec2-user@part2assign4 ~]$ sudo systemctl statis
Unknown operation 'statis'.
[ec2-user@part2assign4 ~]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; vendor preset: disabled)
   Active: active (running) since Fri 2023-03-31 20:30:05 UTC; 2 days ago
     Main PID: 3256 (nginx)
    CGroup: /system.slice/nginx.service
            └─3256 nginx: master process /usr/sbin/nginx
               └─3258 nginx: worker process

Mar 31 20:30:04 part2assign4 systemd[1]: Starting The nginx HTTP and reverse proxy server...
Mar 31 20:30:05 part2assign4 systemd[1]: Started The nginx HTTP and reverse proxy server.
Mar 31 20:30:07 part2assign4 nginx[3248]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Mar 31 20:30:07 part2assign4 nginx[3248]: nginx: configuration file /etc/nginx/nginx.conf test is successful
[ec2-user@part2assign4 ~]$
```

i-04bb294a44bd30c3b (Part2CdkStack/ec2-instance)

PublicIPs: 13.115.84.5 PrivateIPs: 10.0.0.161

-> granted read-write permissions to the Bucket, we can view the objects in the bucket

```
Last login: Fri Mar 31 20:32:49 2023 from ec2-3-112-23-3.ap-northeast-1.compute.amazonaws.com
```

```
  _ | _ | _ )  
  _ | ( _ /   Amazon Linux 2 AMI  
  _ |\_|_|_|
```

```
https://aws.amazon.com/amazon-linux-2/
```

```
[ec2-user@part2assign4 ~]$ aws s3 ls s3://part2assigncdklbhhggbgboo007
```

```
[ec2-user@part2assign4 ~]$ aws s3 ls s3://part2assigncdklbhhggbgboo007
```

```
2023-03-31 20:58:33      574838 JIO UPLOAD.pdf
```

```
2023-03-31 20:58:34      46392 generic-food.csv
```

```
[ec2-user@part2assign4 ~]$
```

i-04bb294a44bd30c3b (Part2CdkStack/ec2-instance)

PublicIPs: 13.115.84.5 PrivateIPs: 10.0.0.161