Setting up CDK
->first install node.js,then npm
->After installing Node.js, install the AWS CDK Toolkit(npm install -g aws-cdk)
->npm install -g typescript

Creating a project
->mkdir newcdk
->cd newcdk
->cdk init app --language typescript

Commands

`cdk list` (`ls`)	Lists the stacks in the app
`cdk synthesize` (`synth`)	Synthesizes and prints the CloudFormation template for one or more specified stacks
`cdk bootstrap`	Deploys the CDK Toolkit staging stack; see Bootstrapping
`cdk deploy`	Deploys one or more specified stacks
`cdk destroy`	Destroys one or more specified stacks
`cdk diff`	Compares the specified stack and its dependencies with the deployed stacks

Create a CDK code using TS to deploy an S3 Bucket with versioning, Encryption enabled (KMS)

->created a new stack named -NewcdkStack on starting point of CDK, newcdk.ts.

```
const app = new cdk.App();
new NewcdkStack(app, 'NewcdkStack', {});
```

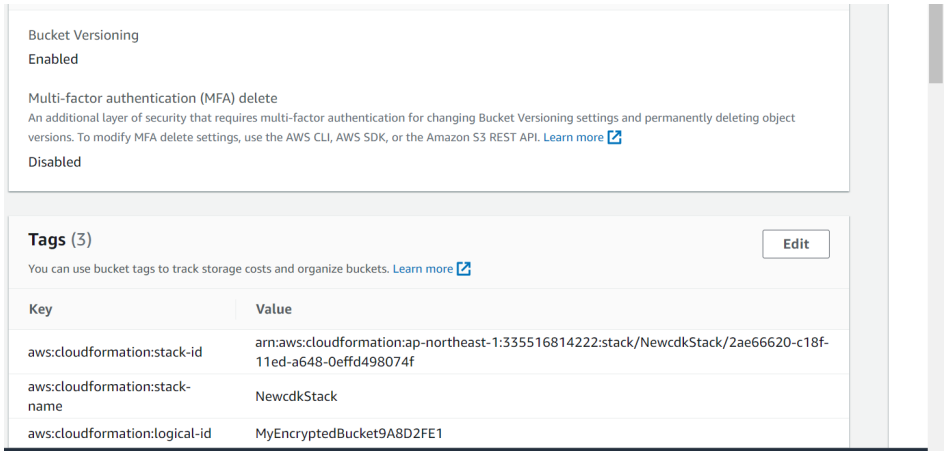
-> then cdk ls to check the stack names

PS C:\Users\Rraks\try1\newcdk> cdk ls
NewcdkStack

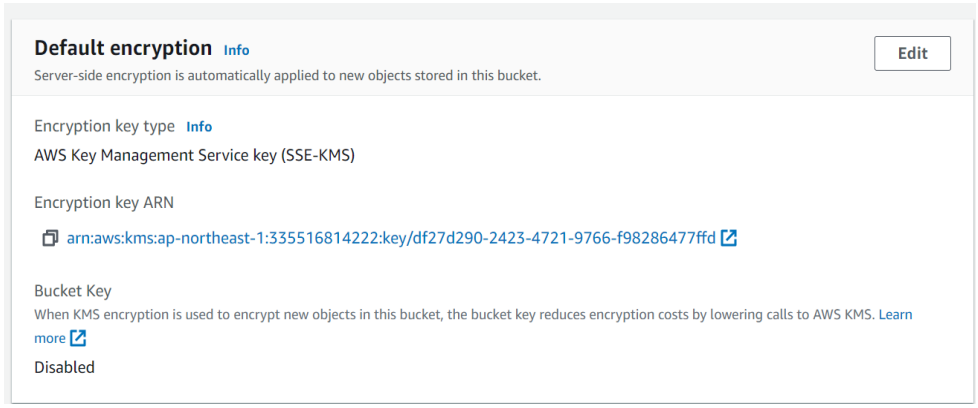
-> then on the stack-newcdk-stack.ts- created a bucket- with bucketname-yurnkiyu787, With versioning enabled , with kms encryption enabled, with lifecycyle rules-rules for Infrequent Access transition after 30 days and Glacier after 90 Days.

```
const bucket = new s3.Bucket(this, 'MyEncryptedBucket', {
      encryption: s3.BucketEncryption.KMS,
      versioned:true,
      bucketName:"yurnkiyu787",
      lifecycleRules: [
        {
            transitions: [
                {
                    storageClass: s3.StorageClass.INFREQUENT_ACCESS,
                    transitionAfter: cdk.Duration.days(30),
                },
                {
                    storageClass: s3.StorageClass.GLACIER,
                    transitionAfter: cdk.Duration.days(90),
                },
            ],
        },
    ],
    });
```

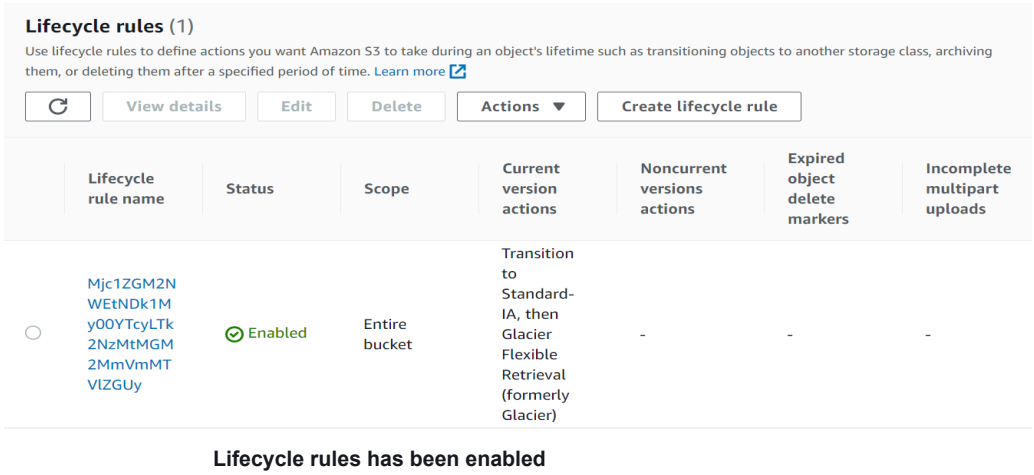
-> then cdk deploy NewcdkStack



Versioning has been enabled on the bucket



Encryption has been enabled



Lifecycle rules has been enabled