CI/CD Pipeline to Build, Test and Upload to ECR

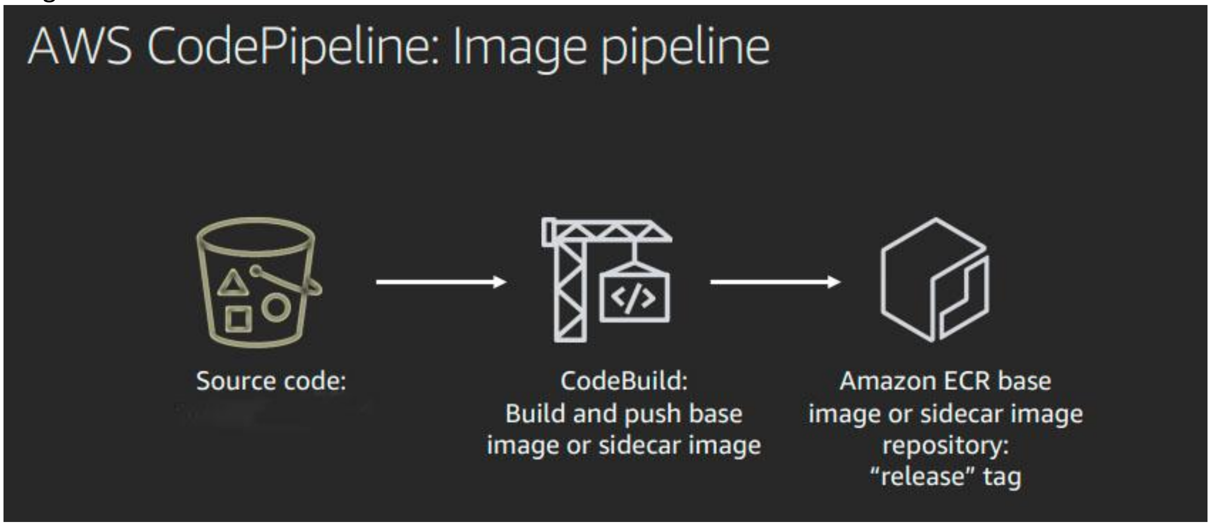
Project - To build Test and Upload to ECR.
Project Reference Git Repo: https://github.com/pypa/sampleproject

Infra-as-code : AWS CDK
Services Created : AWS Code Pipeline, S3, ECR,Code Build

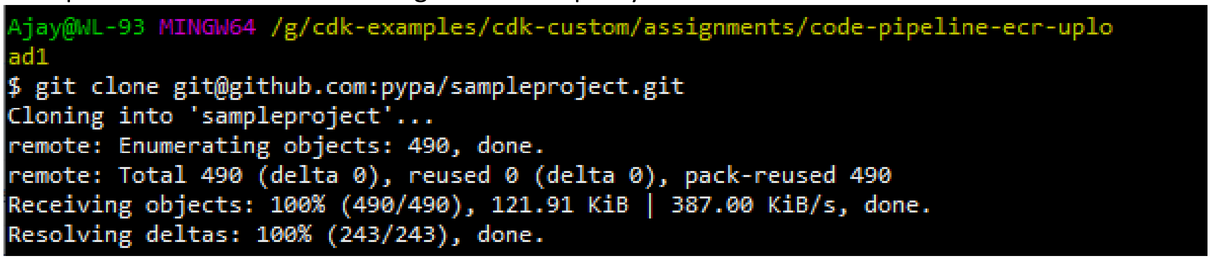
This can be achieved via two methods.
(1) Keeping S3 as Source
(2) Keeping CodeCommit as Source

Method -1

Diagram-



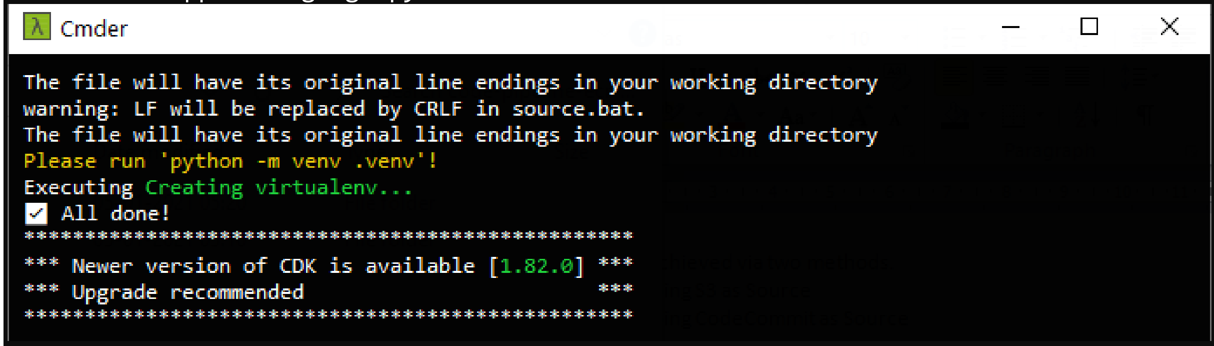
Temporary Pulling of Code
Let's pull the reference code from github in temporay folder



```
Ajay@WL-93 MINGW64 /g/cdk-examples/cdk-custom/assignments/code-pipeline-ecr-uplo
ad1
$ git clone git@github.com:pypa/sampleproject.git
Cloning into 'sampleproject'...
remote: Enumerating objects: 490, done.
remote: Total 490 (delta 0), reused 0 (delta 0), pack-reused 490
Receiving objects: 100% (490/490), 121.91 KiB | 387.00 KiB/s, done.
Resolving deltas: 100% (243/243), done.
```

Create a new folder and initialize CDK

```
λ mkdir ci-ecr-upload
λ cd ci-ecr-upload\
λ cdk init app --language python
```



```
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in source.bat.
The file will have its original line endings in your working directory
Please run 'python -m venv .venv'!
Executing Creating virtualenv...
✓ All done!
***************************************************
*** Newer version of CDK is available [1.82.0] ***
*** Upgrade recommended                         ***
***************************************************
```

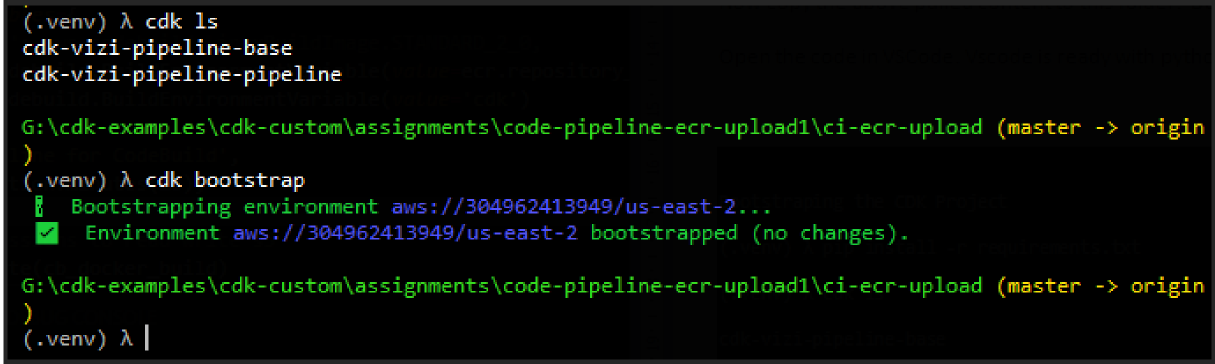
```
λ .venv\Scripts\activate.bat

(.venv) λ ls
app.py*  cdk.json  ci_ecr_upload/  README.md  requirements.txt  setup.py  source.bat

(.venv) λ python --version
Python 3.8.3

(.venv) λ pip --version
pip 19.2.3 from g:\cdk-examples\cdk-custom\assignments\code-pipeline-ecr-upload1\ci-ecr-
upload\.venv\lib\site-packages\pip (python 3.8)

(.venv) λ npm --version
6.14.9

(.venv) λ git --version
git version 2.22.0.windows.1

(.venv) λ git config --global user.name "Ajay Kumar"
(.venv) λ git config --global user.email "ajay011.sharma@hotmail.com"

# In this example cdk is already installed if not there, we can install this from npm
install -g aws-cdk


(.venv) λ pip list
Package    Version
---------- -------
pip        19.2.3
setuptools 41.2.0

#Installing awscli and aws_cdk.core
(.venv) λ pip install awscli aws_cdk.core

 (.venv) λ cdk --version
1.81.0 (build 6ef67c7)

(.venv) λ aws --version
aws-cli/1.18.208 Python/3.8.3 Windows/10 botocore/1.19.48

(.venv) λ aws configure
```

```
(.venv) λ aws configure list
      Name                    Value             Type    Location
      ----                    -----             ----    --------
   profile                <not set>             None    None
access_key     ****************MKWB shared-credentials-file
secret_key     ****************bBlN shared-credentials-file
    region                us-east-2      config-file    ~/.aws/config
```

Now copy the above pulled content to this folder. Do not keep unwanted files like .git etc.

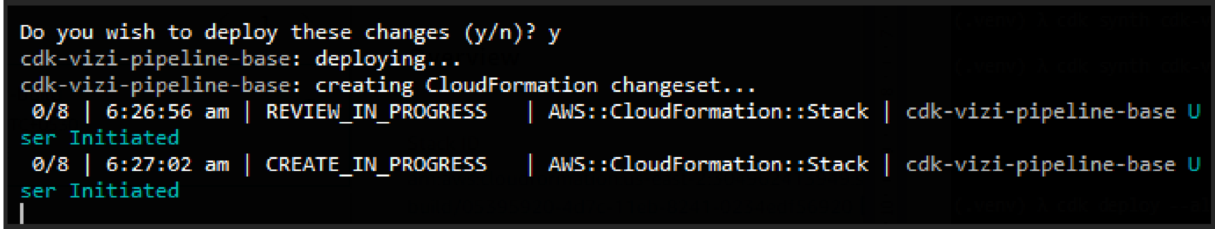
Open the code in VSCode. Vscode is ready with python extension.

```
Bootstraping the CDK Project
(.venv) λ pip install -r requirements.txt
(.venv) λ cdk ls
cdk-vizi-pipeline-base
cdk-vizi-pipeline-pipeline
(.venv) λ cdk bootstrap
```



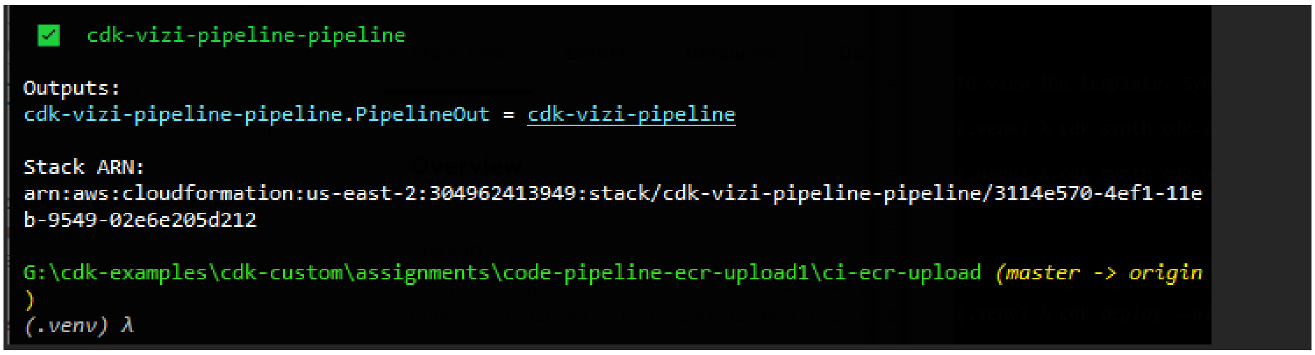
```
(.venv) λ cdk ls
cdk-vizi-pipeline-base
cdk-vizi-pipeline-pipeline

G:\cdk-examples\cdk-custom\assignments\code-pipeline-ecr-upload1\ci-ecr-upload (master -> origin
)
(.venv) λ cdk bootstrap
 ⏳  Bootstrapping environment aws://304962413949/us-east-2...
 ✅  Environment aws://304962413949/us-east-2 bootstrapped (no changes).

G:\cdk-examples\cdk-custom\assignments\code-pipeline-ecr-upload1\ci-ecr-upload (master -> origin
)
(.venv) λ |
```

```
To view the template. Synth is automatically done from app.py
(.venv) λ cdk synth cdk-vizi-pipeline-base
(.venv) λ cdk synth cdk-vizi-pipeline-pipeline


(.venv) λ cdk deploy --all

Confirm the changes.
```



```
Do you wish to deploy these changes (y/n)? y
cdk-vizi-pipeline-base: deploying...
cdk-vizi-pipeline-base: creating CloudFormation changeset...
 0/8 | 6:26:56 am | REVIEW_IN_PROGRESS   | AWS::CloudFormation::Stack | cdk-vizi-pipeline-base U
ser Initiated
 0/8 | 6:27:02 am | CREATE_IN_PROGRESS   | AWS::CloudFormation::Stack | cdk-vizi-pipeline-base U
ser Initiated
|
```

```
Infra - Deployment is finished
```

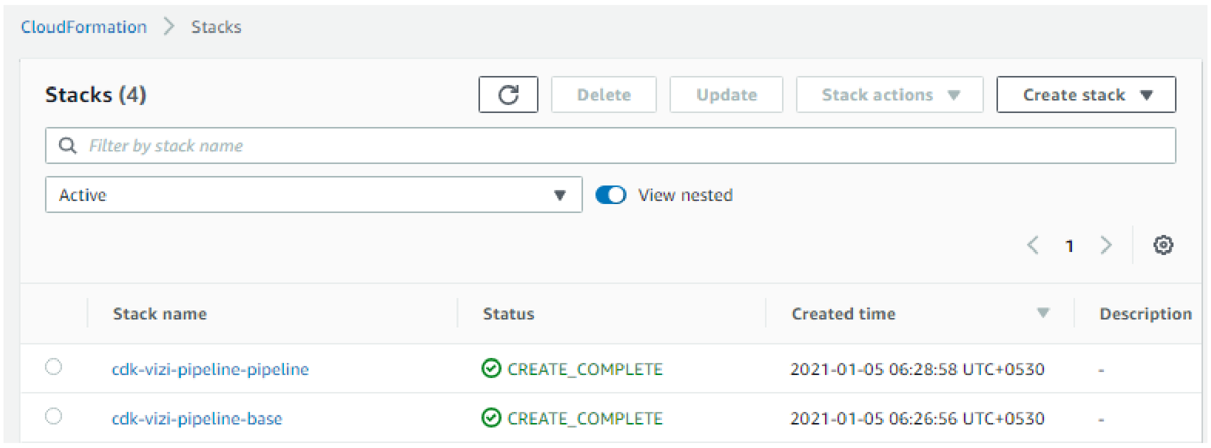


```
✅  cdk-vizi-pipeline-pipeline

Outputs:
cdk-vizi-pipeline-pipeline.PipelineOut = cdk-vizi-pipeline

Stack ARN:
arn:aws:cloudformation:us-east-2:304962413949:stack/cdk-vizi-pipeline-pipeline/3114e570-4ef1-11e
b-9549-02e6e205d212

G:\cdk-examples\cdk-custom\assignments\code-pipeline-ecr-upload1\ci-ecr-upload (master -> origin
)
(.venv) λ
```

In this exampleweare creating following resources

- New S3 Bucket
- New ECR
- Code Build
- Code Pipeline

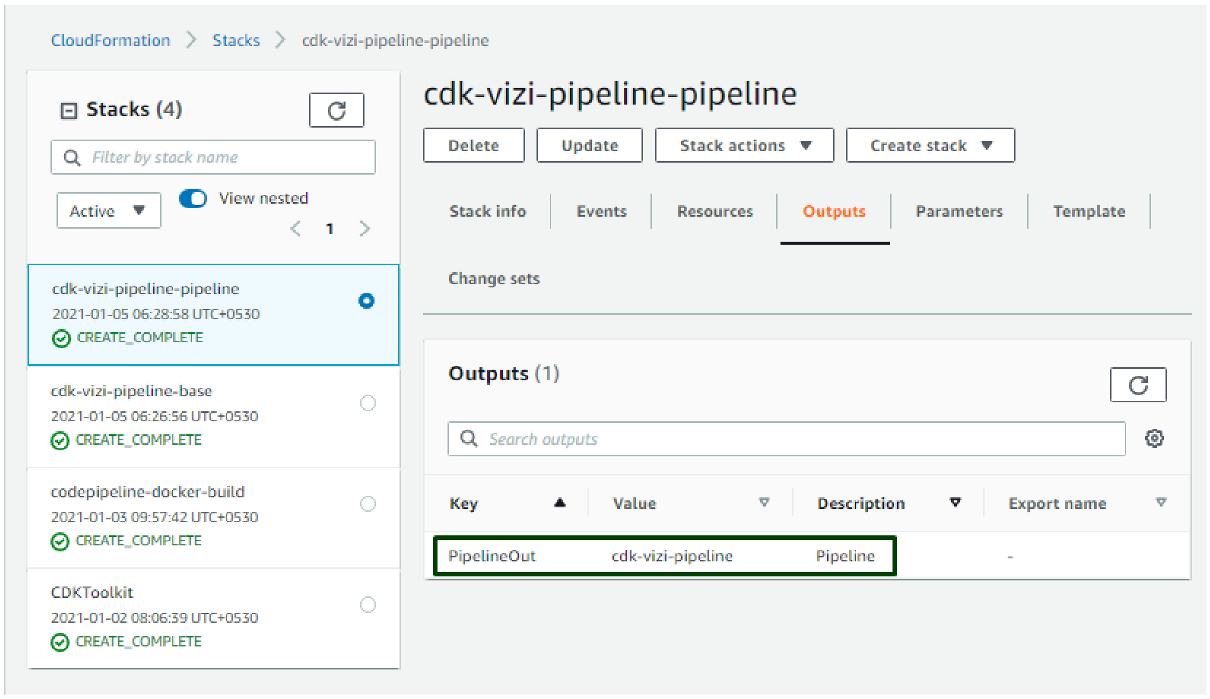
Stacks on Console



CloudFormation > Stacks

Stacks (4)

	Stack name	Status	Created time	Description
○	cdk-vizi-pipeline-pipeline	✓ CREATE_COMPLETE	2021-01-05 06:28:58 UTC+0530	-
○	cdk-vizi-pipeline-base	✓ CREATE_COMPLETE	2021-01-05 06:26:56 UTC+0530	-



CloudFormation > Stacks > cdk-vizi-pipeline-pipeline

Stacks (4)

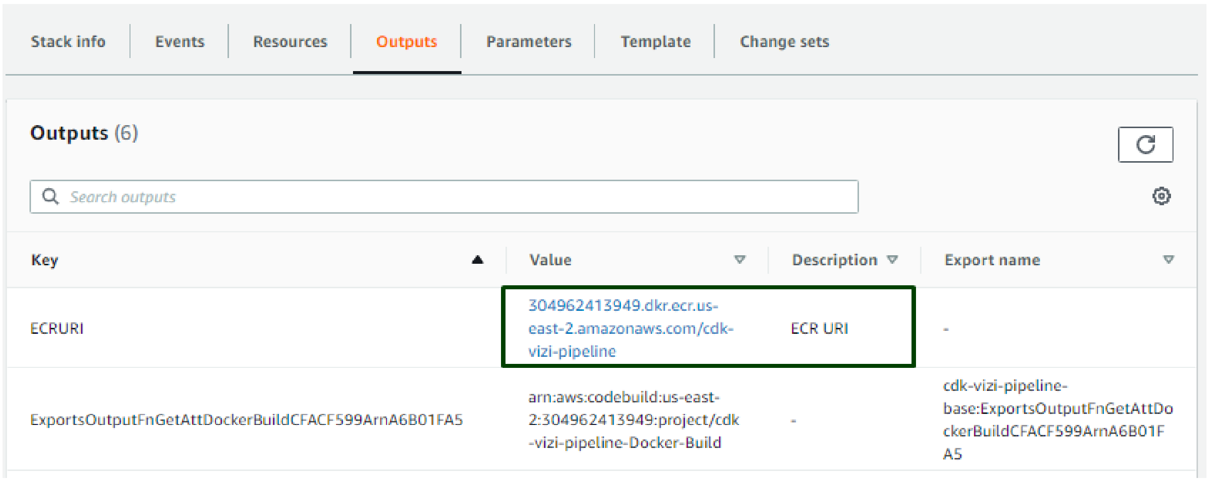
cdk-vizi-pipeline-pipeline	2021-01-05 06:28:58 UTC+0530	CREATE_COMPLETE
cdk-vizi-pipeline-base	2021-01-05 06:26:56 UTC+0530	CREATE_COMPLETE
codepipeline-docker-build	2021-01-03 09:57:42 UTC+0530	CREATE_COMPLETE
CDKToolkit	2021-01-02 08:06:39 UTC+0530	CREATE_COMPLETE

cdk-vizi-pipeline-pipeline

Outputs (1)

Key	Value	Description	Export name
PipelineOut	cdk-vizi-pipeline	Pipeline	-

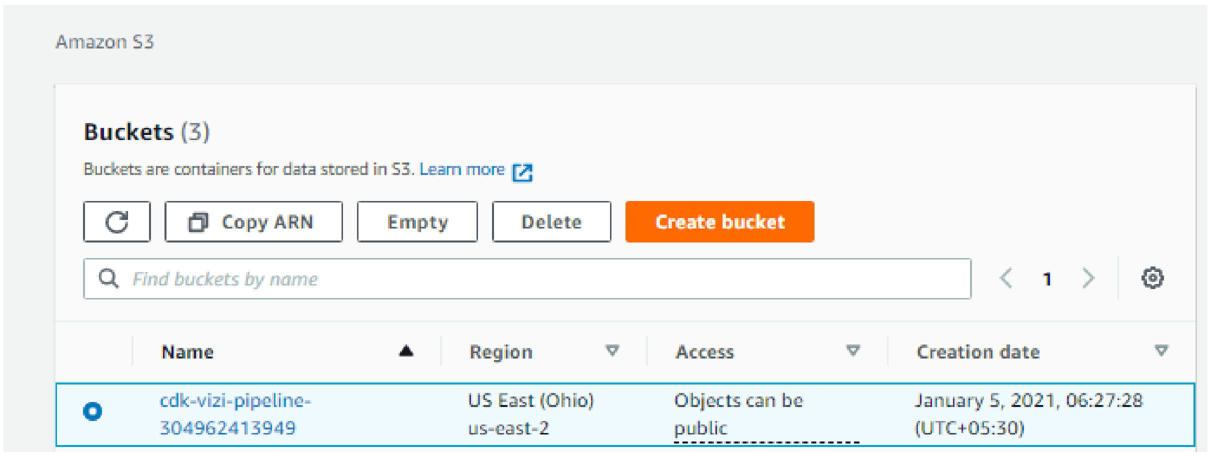
Outputs



Outputs (6)

Key	Value	Description	Export name
ECRURI	304962413949.dkr.ecr.us-east-2.amazonaws.com/cdk-vizi-pipeline	ECR URI	-
ExportsOutputFnGetAttDockerBuildCFACF599ArnA6B01FA5	arn:aws:codebuild:us-east-2:304962413949:project/cdk-vizi-pipeline-Docker-Build	-	cdk-vizi-pipeline-base:ExportsOutputFnGetAttDockerBuildCFACF599ArnA6B01FA5

Bucket created to upload zip source later.


Amazon S3

Buckets (3)

Buckets are containers for data stored in S3. Learn more

	Name	Region	Access	Creation date
●	cdk-vizi-pipeline-304962413949	US East (Ohio) us-east-2	Objects can be public	January 5, 2021, 06:27:28 (UTC+05:30)

Code Build in Code Pipeline

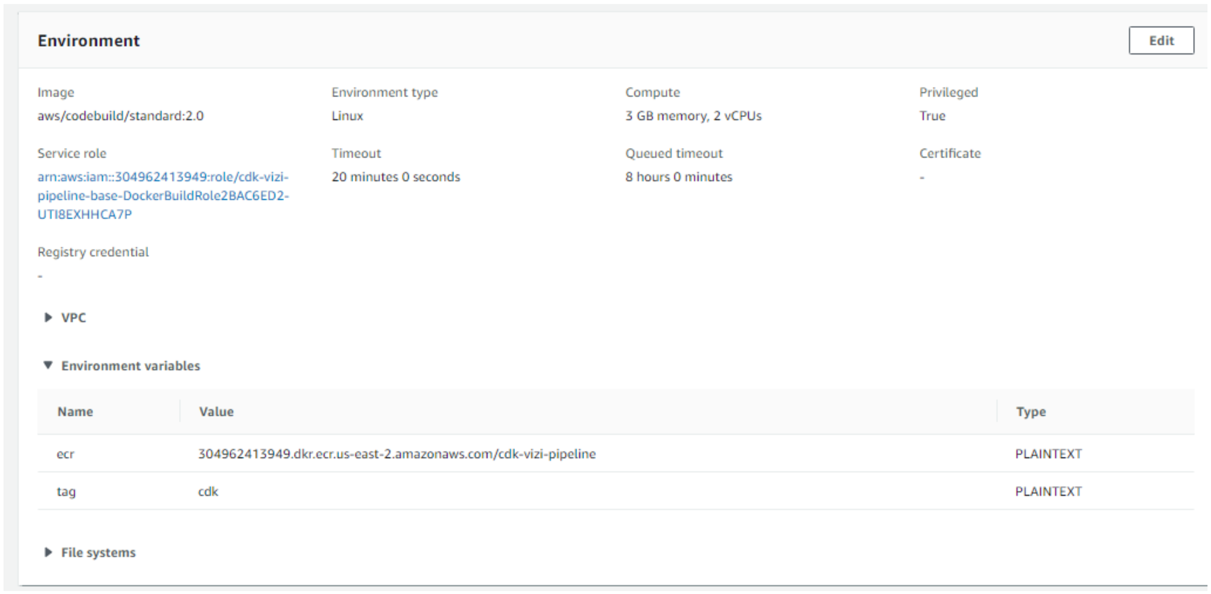

Environment

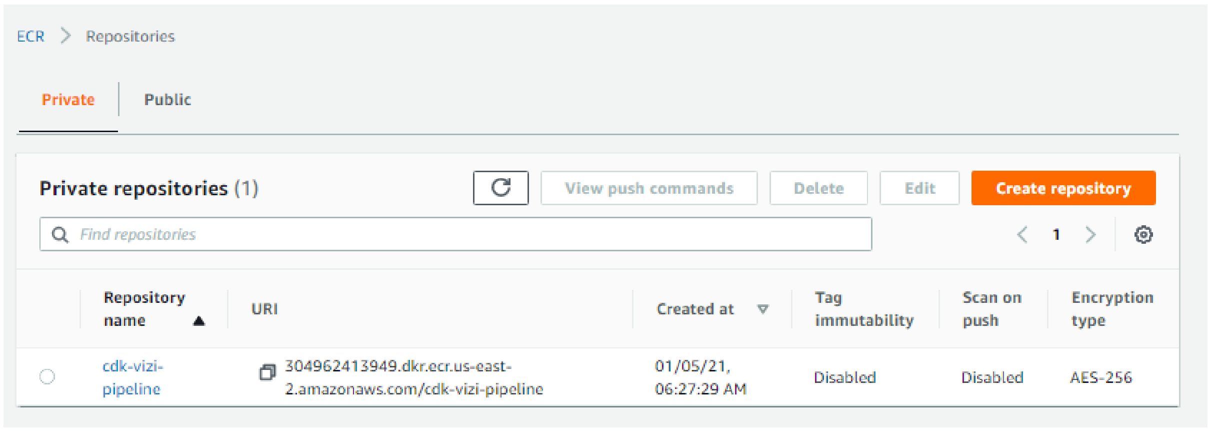
Image aws/codebuild/standard:2.0	Environment type Linux	Compute 3 GB memory, 2 vCPUs	Privileged True
Service role arn:aws:iam::304962413949:role/cdk-vizi-pipeline-base-DockerBuildRole2BAC6ED2-UTI8EXHHCA7P	Timeout 20 minutes 0 seconds	Queued timeout 8 hours 0 minutes	Certificate -
Registry credential -			

▶ VPC

▼ Environment variables

Name	Value	Type
ecr	304962413949.dkr.ecr.us-east-2.amazonaws.com/cdk-vizi-pipeline	PLAINTEXT
tag	cdk	PLAINTEXT

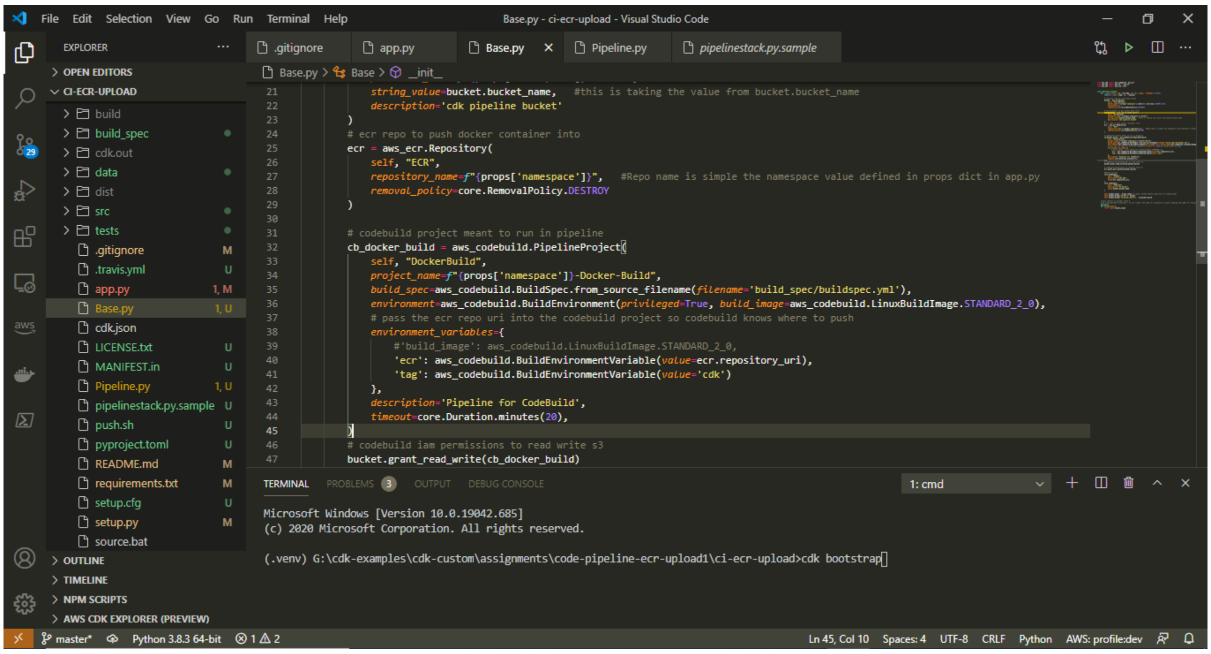
▶ File systems


ECR > Repositories

Private repositories (1)

	Repository name	URI	Created at	Tag immutability	Scan on push	Encryption type
○	cdk-vizi-pipeline	304962413949.dkr.ecr.us-east-2.amazonaws.com/cdk-vizi-pipeline	01/05/21, 06:27:29 AM	Disabled	Disabled	AES-256

VS Code View



```
            string_value=bucket.bucket_name,   #this is taking the value from bucket.bucket_name
            description='cdk pipeline bucket'
        )
        # ecr repo to push docker container into
        ecr = aws_ecr.Repository(
            self, "ECR",
            repository_name=f"{props['namespace']}",   #Repo name is simple the namespace value defined in props dict in app.py
            removal_policy=core.RemovalPolicy.DESTROY
        )

        # codebuild project meant to run in pipeline
        cb_docker_build = aws_codebuild.PipelineProject(
            self, "DockerBuild",
            project_name=f"{props['namespace']}-Docker-Build",
            build_spec=aws_codebuild.BuildSpec.from_source_filename(filename='build_spec/buildspec.yml'),
            environment=aws_codebuild.BuildEnvironment(privileged=True, build_image=aws_codebuild.LinuxBuildImage.STANDARD_2_0),
            # pass the ecr repo uri into the codebuild project so codebuild knows where to push
            environment_variables={
                #'build_image': aws_codebuild.LinuxBuildImage.STANDARD_2_0,
                'ecr': aws_codebuild.BuildEnvironmentVariable(value=ecr.repository_uri),
                'tag': aws_codebuild.BuildEnvironmentVariable(value='cdk')
            },
            description='Pipeline for CodeBuild',
            timeout=core.Duration.minutes(20),
        )
        # codebuild iam permissions to read write s3
        bucket.grant_read_write(cb_docker_build)
```

```
Microsoft Windows [Version 10.0.19042.685]
(c) 2020 Microsoft Corporation. All rights reserved.

(.venv) G:\cdk-examples\cdk-custom\assignments\code-pipeline-ecr-upload1\ci-ecr-upload>cdk bootstrap
```

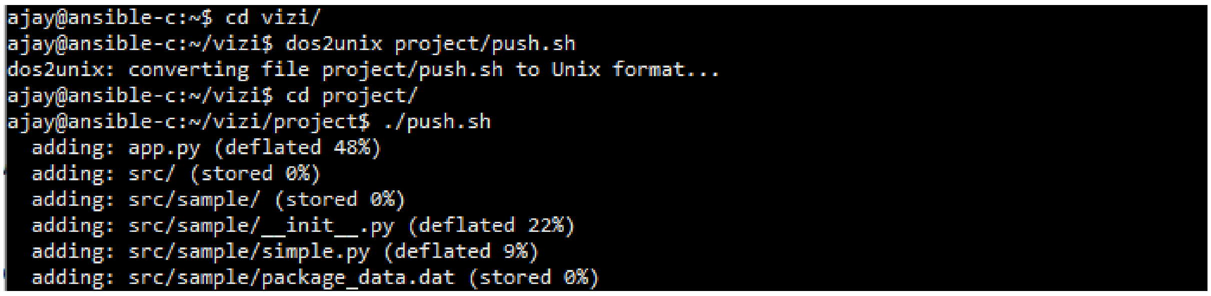
Deployment Process

Let's upload code to S3 bucket.

For this we would use Linux machine and run push.sh. Here is the content in the list.We can exclude the folders as per our requirement in Zip command.

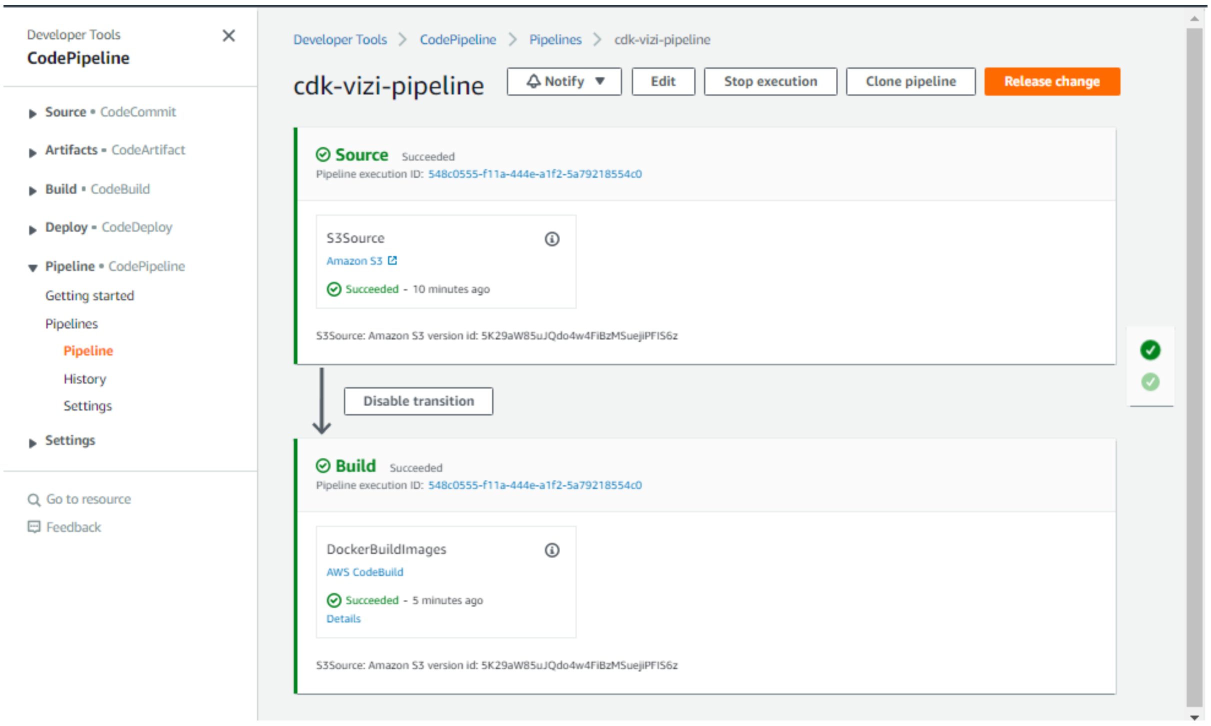
```
#!/usr/bin/env bash

export account_id=$(aws sts get-caller-identity | jq -r .Account)
export source_bucket=$(aws ssm get-parameter --name 'cdk-vizi-pipeline-bucket' | jq -
r .Parameter.Value)
export pipeline_name=$(aws ssm get-parameter --name 'cdk-vizi-pipeline-pipeline' | jq -
r .Parameter.Value)
export REGION='us-east-2'

zip -r source.zip .
aws s3 cp source.zip s3://${source_bucket}/source.zip
#aws codepipeline start-pipeline-execution --name ${pipeline_name}
```

On Linux Shell



```
ajay@ansible-c:~$ cd vizi/
ajay@ansible-c:~/vizi$ dos2unix project/push.sh
dos2unix: converting file project/push.sh to Unix format...
ajay@ansible-c:~/vizi$ cd project/
ajay@ansible-c:~/vizi/project$ ./push.sh
  adding: app.py (deflated 48%)
  adding: src/ (stored 0%)
  adding: src/sample/ (stored 0%)
  adding: src/sample/__init__.py (deflated 22%)
  adding: src/sample/simple.py (deflated 9%)
  adding: src/sample/package_data.dat (stored 0%)
```

Once the code is pushed we can see the deployment executed and Code build get success status.



Developer Tools > CodePipeline > Pipelines > cdk-vizi-pipeline

cdk-vizi-pipeline

Source Succeeded
Pipeline execution ID: 548c0555-f11a-444e-a1f2-5a79218554c0

S3Source
Amazon S3
Succeeded - 10 minutes ago

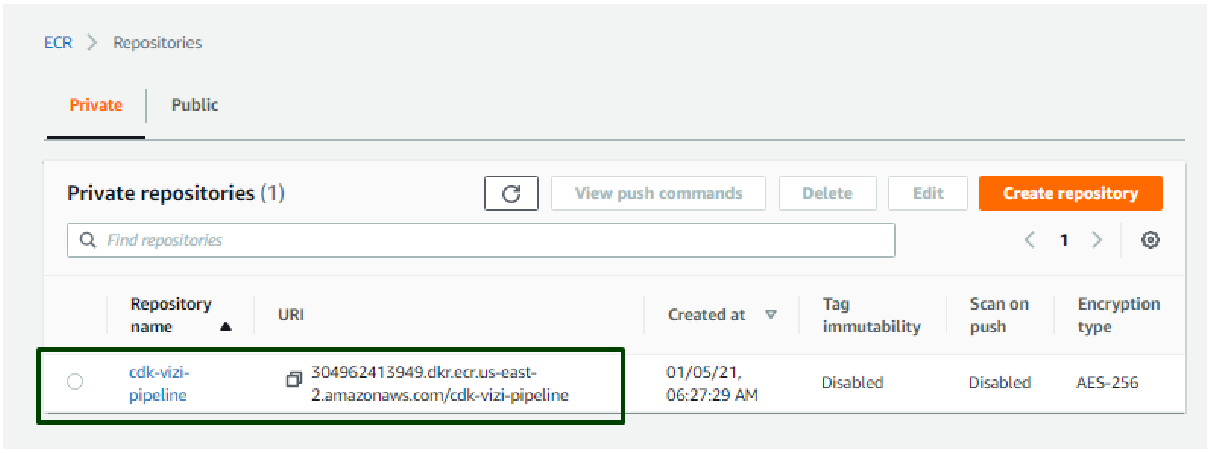
S3Source: Amazon S3 version id: 5K29aW85uJQdo4wFiBzMSuejiPFIS6z

Build Succeeded
Pipeline execution ID: 548c0555-f11a-444e-a1f2-5a79218554c0

DockerBuildImages
AWS CodeBuild
Succeeded - 5 minutes ago
Details

S3Source: Amazon S3 version id: 5K29aW85uJQdo4wFiBzMSuejiPFIS6z

We have successfully deployed the application. Let's check the ECR



ECR > Repositories

Private repositories (1)

Repository name	URI	Created at	Tag immutability	Scan on push	Encryption type
cdk-vizi-pipeline	304962413949.dkr.ecr.us-east-2.amazonaws.com/cdk-vizi-pipeline	01/05/21, 06:27:29 AM	Disabled	Disabled	AES-256

Explanation of Docker file

We have simply take a base Image of python and copy the contents to the src directory and finally we run setup commands 'pip install .'. In this example we do not require toExpose any port or define any Entrypoint for pyton.

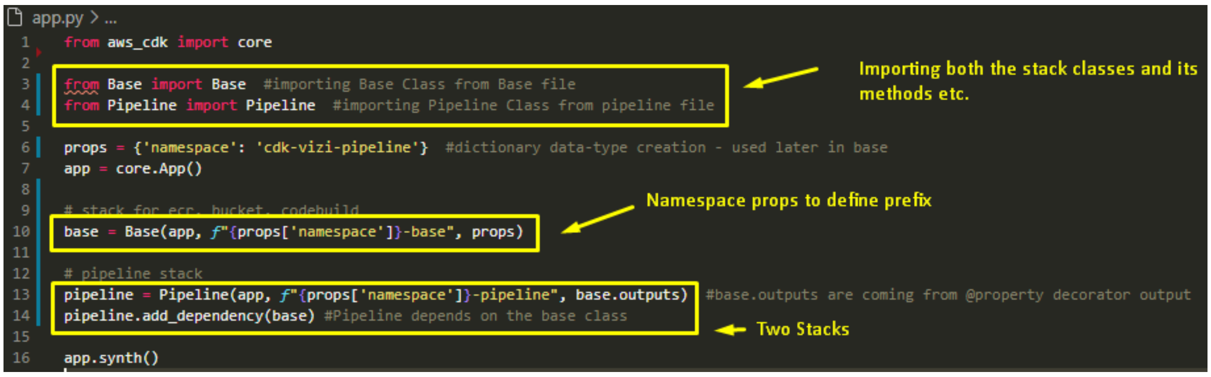
```
ARG PYTHON_VERSION=3.7

FROM python:${PYTHON_VERSION}

RUN mkdir /src
WORKDIR /src

COPY requirements.txt /src/requirements.txt
RUN pip install -r requirements.txt

COPY . /src
RUN pip install .
```

Define Buildpsec.yml file

Weuse here python: 3.7 as runtime and different phases. We also do the unit-testing and login to ECR and building the docker and uploading works.

```
version: 0.2
phases:
  install:
    runtime-versions:
      python: 3.7
    commands:
      - echo Entered the install phase...
      - python --version
      - pip --version
    finally:
      - echo This always runs even if the update or install command fails
  pre_build:
    commands:
      - echo Pre-Build started on `date`
      - echo Installing the python requirements
      - pip install -e .
      - pip list
      - aws --version
      - docker --version
  build:
    commands:
      - echo Build started on `date`
      - echo Testing the python before uploading
      - python setup.py install
      - cd tests
      - python -m unittest -v test_simple.py
      - cd ..
      - echo Building the Docker image...
      - echo $AWS_DEFAULT_REGION
      - echo Logging in to Amazon ECR...
      - $(aws ecr get-login --no-include-email --region $AWS_DEFAULT_REGION)
      - docker build -t ${tag}:latest .
  post_build:
    commands:
      - echo Pushing the Docker images to container registry...
      - echo Build started on `date`
      - docker tag $tag:latest $ecr:$tag
      - docker push $ecr
      - echo Writing image definitions file...
      - printf '{"ImageURI":"%s"}' $REPOSITORY_URI:$IMAGE_TAG > imageDetail.json
      - echo Build completed on `date`
      - cat imageDetail.json
      - echo $ecr:$tag
artifacts:
  files:
    - imageDetail.json
```

CDK Code Explanation.



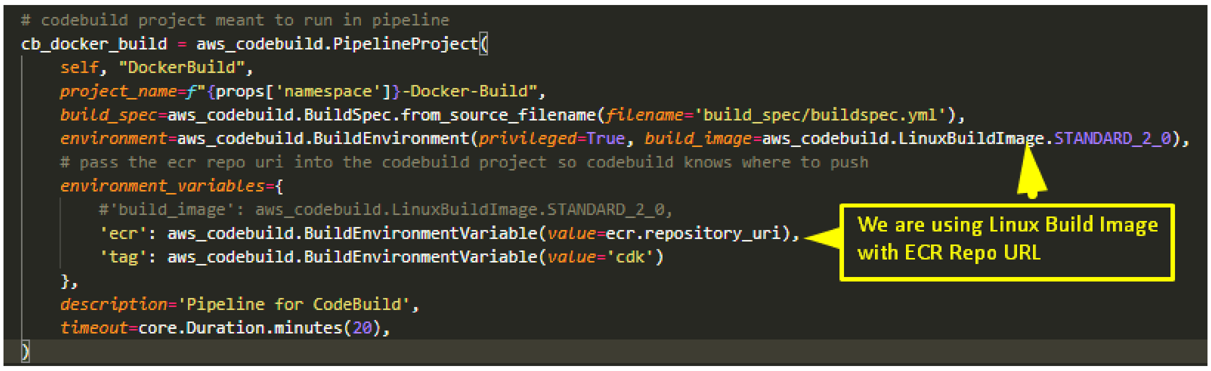
app.py > ...

```python
from aws_cdk import core

from Base import Base  #importing Base Class from Base file
from Pipeline import Pipeline  #importing Pipeline Class from pipeline file

props = {'namespace': 'cdk-vizi-pipeline'}  #dictionary data-type creation - used later in base
app = core.App()

# stack for ecr, bucket, codebuild
base = Base(app, f"{props['namespace']}-base", props)

# pipeline stack
pipeline = Pipeline(app, f"{props['namespace']}-pipeline", base.outputs) #base.outputs are coming from @property decorator output
pipeline.add_dependency(base) #Pipeline depends on the base class

app.synth()
```

Importing both the stack classes and its methods etc.

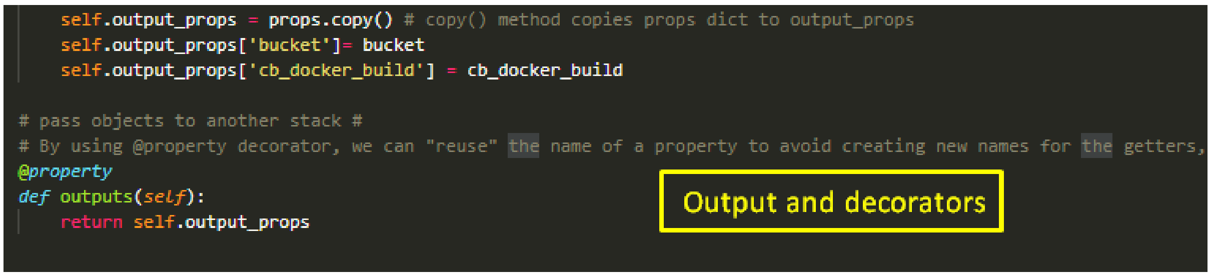
Namespace props to define prefix

Two Stacks

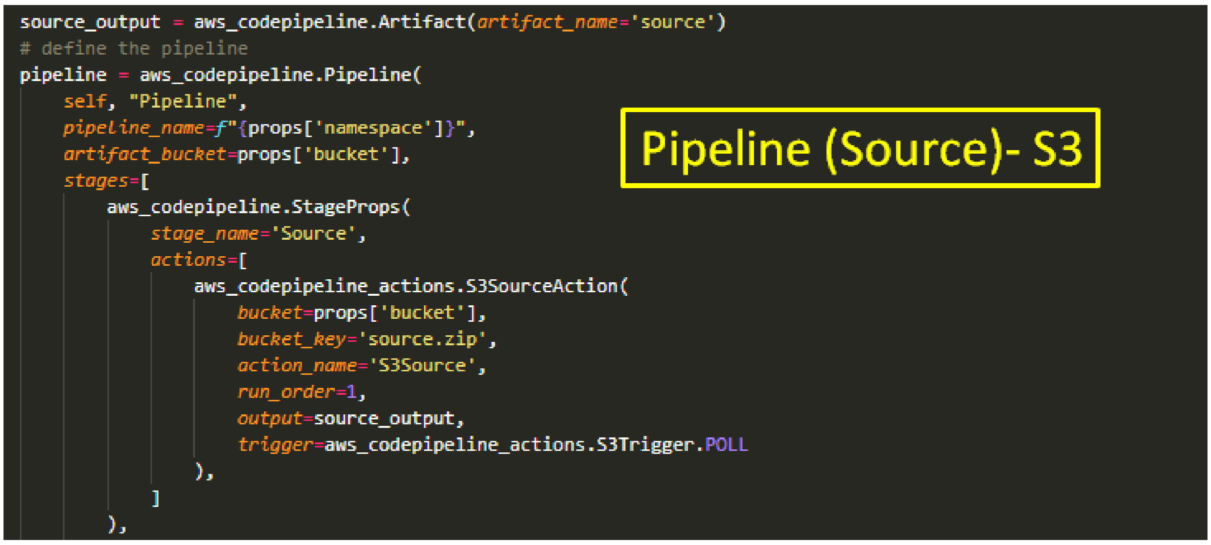


```python
# codebuild project meant to run in pipeline
cb_docker_build = aws_codebuild.PipelineProject(
    self, "DockerBuild",
    project_name=f"{props['namespace']}-Docker-Build",
    build_spec=aws_codebuild.BuildSpec.from_source_filename(filename='build_spec/buildspec.yml'),
    environment=aws_codebuild.BuildEnvironment(privileged=True, build_image=aws_codebuild.LinuxBuildImage.STANDARD_2_0),
    # pass the ecr repo uri into the codebuild project so codebuild knows where to push
    environment_variables={
        #'build_image': aws_codebuild.LinuxBuildImage.STANDARD_2_0,
        'ecr': aws_codebuild.BuildEnvironmentVariable(value=ecr.repository_uri),
        'tag': aws_codebuild.BuildEnvironmentVariable(value='cdk')
    },
    description='Pipeline for CodeBuild',
    timeout=core.Duration.minutes(20),
)
```

We are using Linux Build Image with ECR Repo URL



```python
    self.output_props = props.copy() # copy() method copies props dict to output_props
    self.output_props['bucket']= bucket
    self.output_props['cb_docker_build'] = cb_docker_build

# pass objects to another stack #
# By using @property decorator, we can "reuse" the name of a property to avoid creating new names for the getters,
@property
def outputs(self):
    return self.output_props
```



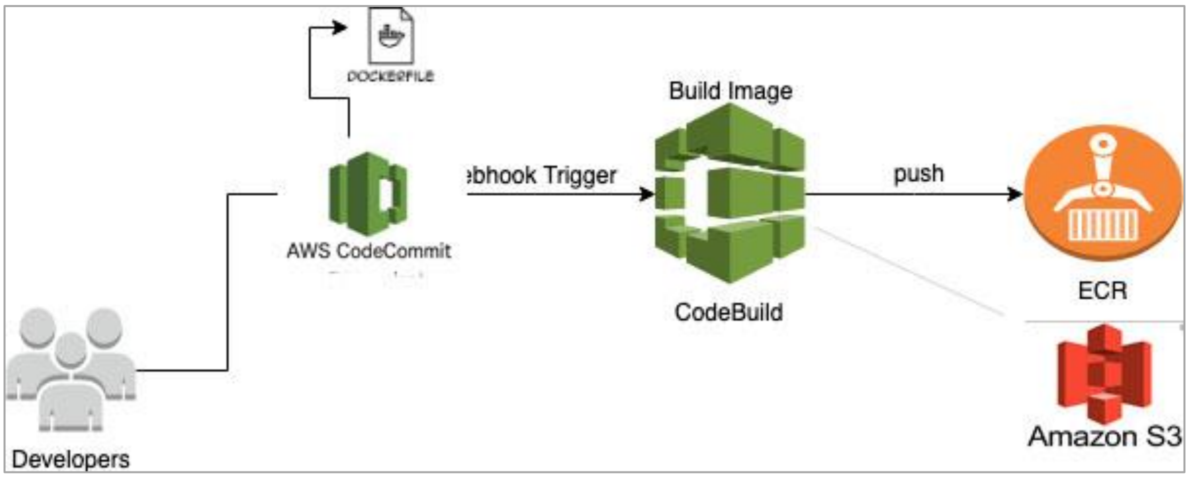
```python
source_output = aws_codepipeline.Artifact(artifact_name='source')
# define the pipeline
pipeline = aws_codepipeline.Pipeline(
    self, "Pipeline",
    pipeline_name=f"{props['namespace']}",
    artifact_bucket=props['bucket'],
    stages=[
        aws_codepipeline.StageProps(
            stage_name='Source',
            actions=[
                aws_codepipeline_actions.S3SourceAction(
                    bucket=props['bucket'],
                    bucket_key='source.zip',
                    action_name='S3Source',
                    run_order=1,
                    output=source_output,
                    trigger=aws_codepipeline_actions.S3Trigger.POLL
                ),
            ]
        ),
```

Method-2

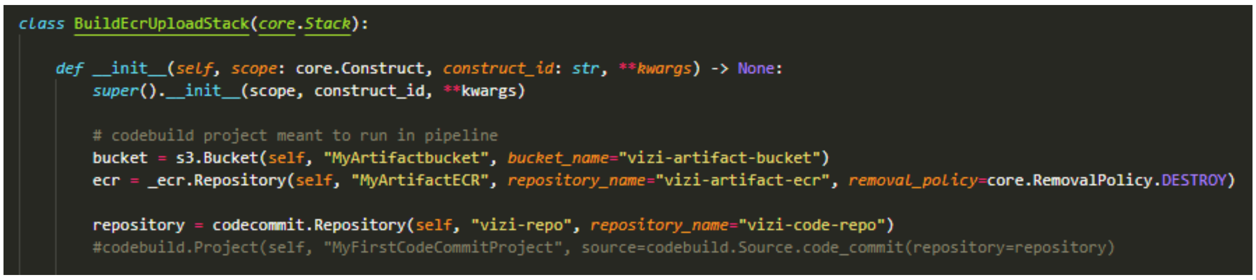
In this example we are just creating the codebuild Infra with ECR and S3 bucket. We are also creating Code Commit repository from CDK.

Target: ECR and S3
Source: Code Commit (manual) trigger.

Diagram:

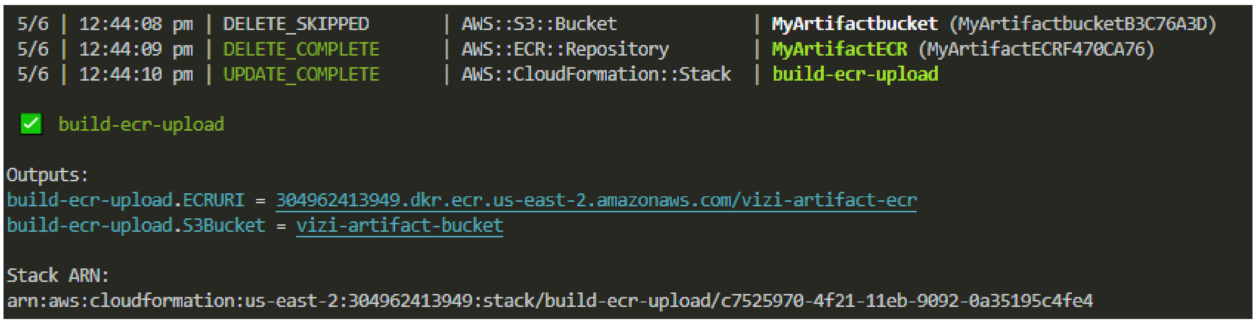


Codes sample for creating Bucket, ECR, and Code Commit Repo



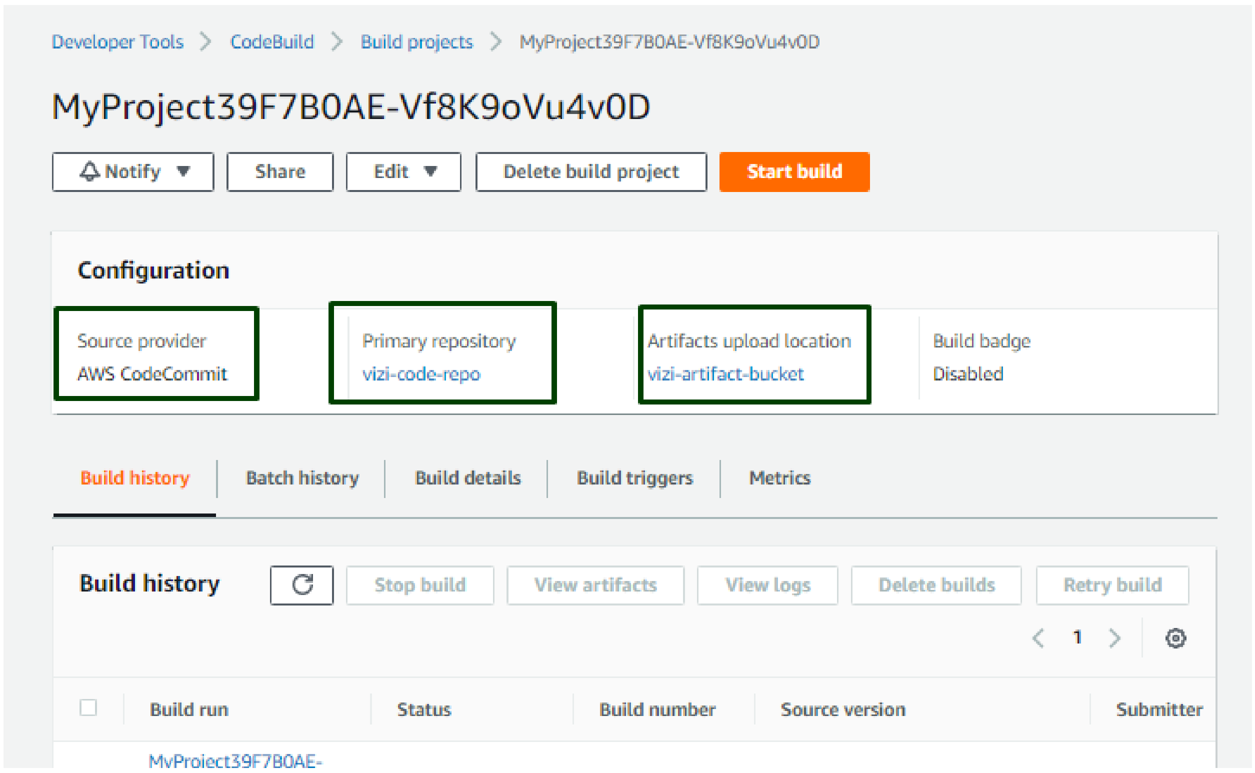
```python
class BuildEcrUploadStack(core.Stack):

    def __init__(self, scope: core.Construct, construct_id: str, **kwargs) -> None:
        super().__init__(scope, construct_id, **kwargs)

        # codebuild project meant to run in pipeline
        bucket = s3.Bucket(self, "MyArtifactbucket", bucket_name="vizi-artifact-bucket")
        ecr = _ecr.Repository(self, "MyArtifactECR", repository_name="vizi-artifact-ecr", removal_policy=core.RemovalPolicy.DESTROY)

        repository = codecommit.Repository(self, "vizi-repo", repository_name="vizi-code-repo")
        #codebuild.Project(self, "MyFirstCodeCommitProject", source=codebuild.Source.code_commit(repository=repository)
```

By using Code build we push the repo in ecr and S3. CDK Output (Terminal)



```
 5/6 | 12:44:08 pm | DELETE_SKIPPED       | AWS::S3::Bucket             | MyArtifactbucket (MyArtifactbucketB3C76A3D)
 5/6 | 12:44:09 pm | DELETE_COMPLETE      | AWS::ECR::Repository        | MyArtifactECR (MyArtifactECRF470CA76)
 5/6 | 12:44:10 pm | UPDATE_COMPLETE      | AWS::CloudFormation::Stack  | build-ecr-upload

 ✅  build-ecr-upload

Outputs:
build-ecr-upload.ECRURI = 304962413949.dkr.ecr.us-east-2.amazonaws.com/vizi-artifact-ecr
build-ecr-upload.S3Bucket = vizi-artifact-bucket

Stack ARN:
arn:aws:cloudformation:us-east-2:304962413949:stack/build-ecr-upload/c7525970-4f21-11eb-9092-0a35195c4fe4
```

Build Projects status on Console



Developer Tools > CodeBuild > Build projects > MyProject39F7B0AE-Vf8K9oVu4v0D

MyProject39F7B0AE-Vf8K9oVu4v0D

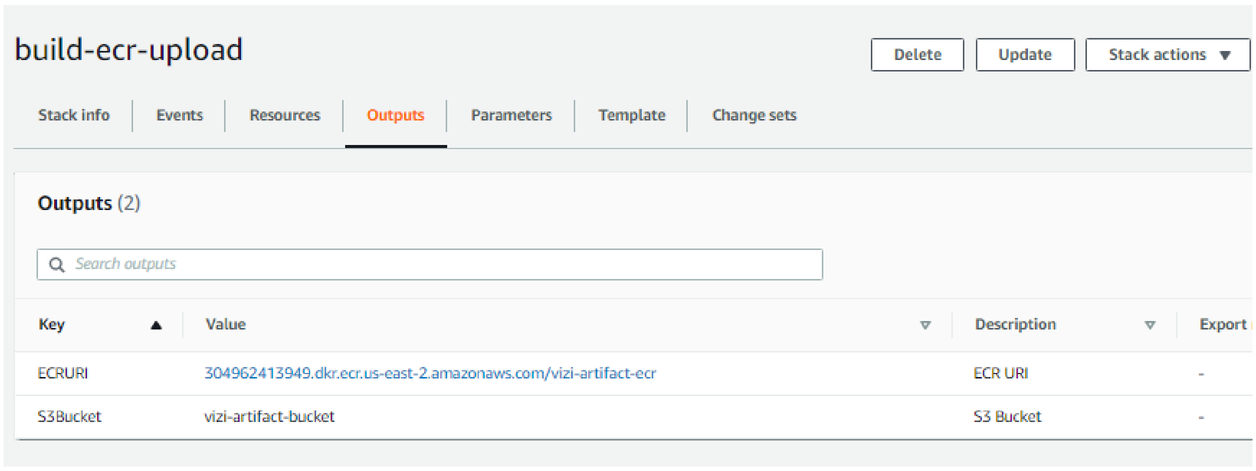
Configuration

Source provider	Primary repository	Artifacts upload location	Build badge
AWS CodeCommit	vizi-code-repo	vizi-artifact-bucket	Disabled

Build history

Build run	Status	Build number	Source version	Submitter
MyProject39F7B0AE-				

Output Value:



build-ecr-upload

Outputs (2)

Key	Value	Description	Export
ECRURI	304962413949.dkr.ecr.us-east-2.amazonaws.com/vizi-artifact-ecr	ECR URI	-
S3Bucket	vizi-artifact-bucket	S3 Bucket	-

Complete CDK Code.

```python
from aws_cdk import core
import aws_cdk.aws_codebuild as codebuild
import aws_cdk.aws_s3 as s3
import aws_cdk.aws_ecr as _ecr
import aws_cdk.aws_codecommit as codecommit

class BuildEcrUploadStack(core.Stack):

    def __init__(self, scope: core.Construct, construct_id: str, **kwargs) -> None:
        super().__init__(scope, construct_id, **kwargs)
        # codebuild project meant to run in pipeline
        bucket = s3.Bucket(self, "MyArtifactbucket", bucket_name="vizi-artifact-bucket")
        ecr = _ecr.Repository(self, "MyArtifactECR", repository_name="vizi-artifact-
ecr", removal_policy=core.RemovalPolicy.DESTROY)

        repository = codecommit.Repository(self, "vizi-repo", repository_name="vizi-code-repo")
        #codebuild.Project(self, "MyFirstCodeCommitProject", source=codebuild.Source.code_commit(r
epository=repository)

        codebuild.Project(self, "MyProject",
                source=codebuild.Source.code_commit(repository=repository),
                build_spec=codebuild.BuildSpec.from_object({
                "version": "0.2",
                "phases": {
                    "build": {
                        "commands": [
                            "echo \"Hello, CodeBuild!\"",
                            "python setup.py install",
                            "cd tests",
                            "python -m unittest -v test_simple.py",
                            "cd ..",
                            "echo Logging in to Amazon ECR...",
                            "$(aws ecr get-login --no-include-email --
region $AWS_DEFAULT_REGION)",
                            "docker build -t ${tag}:latest .",
                            "docker tag $tag:latest $ecr:$tag",
                            "docker push $ecr",
                            "echo Build completed"
                            ]
                        }
                    }
            }),
                artifacts=codebuild.Artifacts.s3(
                bucket=bucket,
                include_build_id=False,
                package_zip=True,
                path="source/files",
                identifier="AddArtifact1"
            )
        )

        #Output Section
        core.CfnOutput(
            self, "ECRURI",
            description="ECR URI",
            value=ecr.repository_uri,
        )
        core.CfnOutput(
            self, "S3Bucket",
            description="S3 Bucket",
            value=bucket.bucket_name
        )
```

(Infra is tested but deployment is not tested)
Thank you,