How to create a pipeline for continuous deployment apps using a custom Docker image

1. Create Docker image

-needs to have:

- Specific python version for the execution of the project

- AWS CLI to execute aws commands latter on.

- Authorization to access the git repository:

- Add the main code commit URL needs to be added as a known host in the client service/vm/container where the repository will be cloned. This can be done by updating/creating (if it does not exist) the .ssh folder

- Include the aws credentials as well ( .aws folder) to access S3

1. Create the CodeCommit repo for the source code.
2. Create S3 bucket for the storage of the /dist/ content.
3. Provider the role/user with permissions to List and Read Objects in the s3 bucket

EX(needs update with our own)

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Principal": {

"AWS": "arn:aws:iam::$ACCOUT\_ID:role/service-role/$NAME\_OF\_ROLL"

},

"Action": "s3:ListBucket",

"Resource": "arn:aws:s3:::$NAME\_OF\_BUCKET"

}

]

}

1. Create role for codeBuild (company permissions)

-it needs permissions to ListObjects GetObjects DeleObjects in the s3 bucket

- these permissions are:

“ecr:GetDownloadUrlForLayer",

"ecr:BatchGetImage",

"ecr:BatchCheckLayerAvailability"

"ecr:BatchCheckLayerAvailability",

"ecr:PutImage",

"ecr:InitiateLayerUpload",

"ecr:UploadLayerPart",

"ecr:CompleteLayerUpload"

1. Create yaml file to instruct codeBuild the steps of execution.

-clone your repo in the image

-install needed dependencies for project to compile.

-sync or copy the dist files into s3 bucket.

1. Create lambda function to invalidate CloudFront (needs update with company own permissions).

* Needs to have permissions to:

codeCommit cloudFront logs codePipeline S3

1. Create pipeline

* Deployment stage is not necessary can put it in the yaml file.

Side note: after created the pipeline we will add another stage to choose the lambda we created, current version of aws interface doesn’t allow to choose the lambda at the beginning .