**DevOps CI/CD Pipeline**

1. Create Lambda function using Python.



1. Create CodeCommit repository via console or through CLI (cloud9/CloudShell) commands. Log in to cloud9 and configure the CodeCommit credential helper as below.

**git config --global credential.helper '!aws codecommit credential-helper $@'**

**git config --global credential.UseHttpPath true**

1. Using below command we can create CodeCommit repository.

**aws codecommit create-repository --repository-name LambdaRepoforCICD--repository-description " LambdaRepoforCICDtest "**

A screenshot of a computer

Description automatically generated

1. Login to cloud9 and download the Lambda function in the cloud 9 terminal so that you can update or modify the Lambda function.

Log in cloud9 🡪explorer🡪select the region 🡪select the Lambda function 🡪right click on the Lambda function and download🡪select the workspace folder for your project.

A screenshot of a computer program

Description automatically generated

1. Change the directory to the lambda directory.

**cd Lambda-for-cicd**

1. Initialize git in the same directory by **git init**, then get the clone URL from CodeCommit repo as shown below.

A green and white box with black text

Description automatically generated

1. Now create the Buildspec file named `buildspec.yml` in the same directory. Buildspec file specify how the Build will happen step by step using Bash commands. Here is the screenshot of the Buildspec file which I used for the build.

A screenshot of a computer code

Description automatically generated

1. Try to push the lambda\_function.py and buildspec.yml using below git commands from cloud 9.
2. Git status🡪git add . 🡪 git commit -m "initial commit" 🡪 git remote add origin https://git-codecommit.us-east-1.amazonaws.com/v1/repos/LambdaRepoforCICD (the cloned URL for CodeCommit repo)🡪 git push -u origin master.
3. Now if you go back and check the CodeCommit repo you will find the updated lambda\_function.py and buildspec.yml files in the same repo as shown below.

A screenshot of a computer

Description automatically generated

1. Create a Service Role for CodeBuild with a trust policy for AWS CodeBuild in IAM and attach necessary permissions to this role. Here I have added codecommit full access and Lambda full access as below.

A screenshot of a computer

Description automatically generated

1. Create the CodeBuild Project.

- Name: build-project-for-cicd

- Source Provider: CodeCommit with master branch

- Environment: Managed image, Amazon Linux 2 standard 5.0

- Service role: the one I just created (service-role-for-codebuild)

- Buildspec: No need to name it but leave enabled

- Optionally enable logging to CloudWatch (recommended)

A screenshot of a computer

Description automatically generated

1. Now we can start Build and we can see build is successful. We can also get the full log view for the build as below.

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A close-up of a computer screen

Description automatically generated

1. We can update the lambda function and push it through cloud9 and if we start the build, we will get the response back from Lambda function.
2. To automate this process we will have to create code pipeline so that whenever, we will modify our code in Lambda and push it through cloud9 into the codecommit repo, pipeline will get triggered automatically and build the project. We do not need to start the build manually. Here I have created codepipeline (code-pipeline-for-lambda-cicd) as below to automate build process. Where you will have to specify source code as your codecommit, repo name ([LambdaRepoforCICD](https://us-east-1.console.aws.amazon.com/codesuite/codecommit/repositories/LambdaRepoforCICD/browse?region=us-east-1)), branch as master and code build project as your build project name (build-project-for-cicd). You can skip the deploy stage because deployment has already been specified in the buildspec file.

Here is the pipeline that is created.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Now we can publish the function through an API.

- Create the API in API Gateway

- Create a REST API (my-rest-API)

-Create resource name (/app)

- Create a GET method

- Integrate the Lambda function as a Lambda proxy integration

- Deploy to the "DEV" stage.

Here is the below endpoint URL, if we browse the URL we will get response back from the Lambda as below.

<https://c6kt6bz431.execute-api.us-east-1.amazonaws.com/DEV/app>

A blue and black text

Description automatically generated

1. Now to test the whole process I will modify the Lambda function code and will push it through the latest changes to codecommit, our code pipeline will get trigger automatically and build the project. Once Build is successful, we can browse the URL and we will get the updated response from Lambda function which will prove the automated DevOps CI/CD pipeline.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated