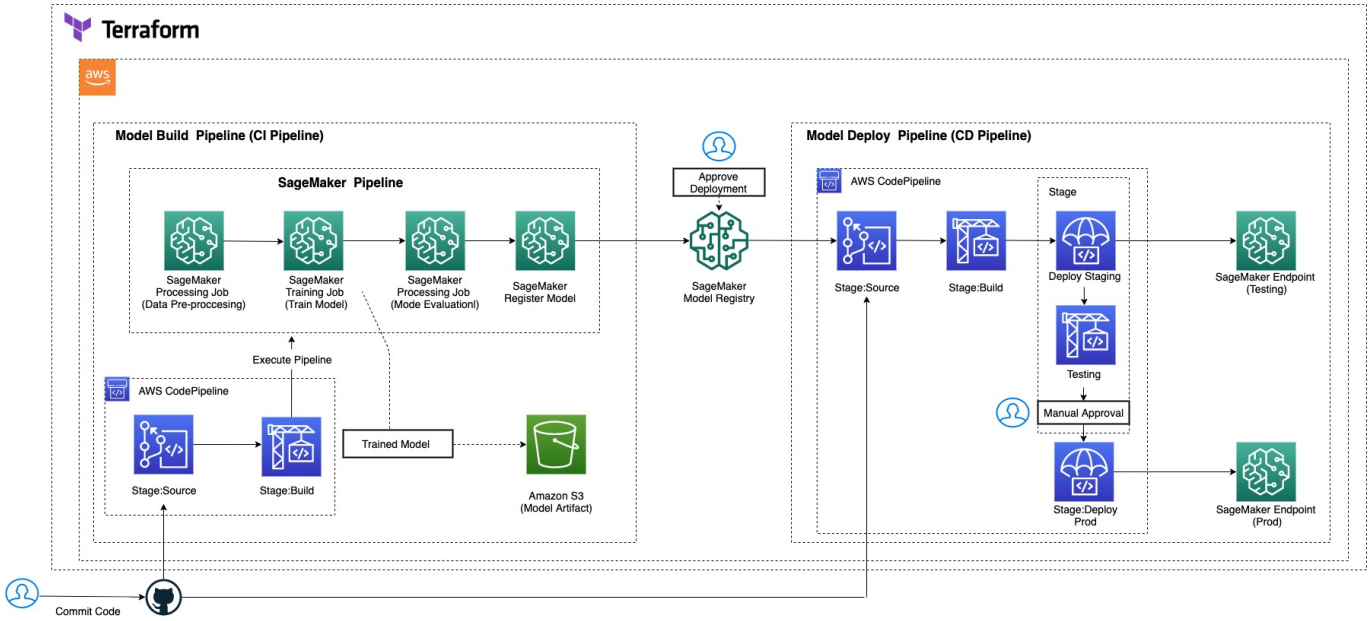
Deploy SageMaker MLOps Pipeline with Terraform and AWS CodePipeline

AWS CodePipeline is a CI/CD orchestration tool for infrastructure on Amazon AWS. Terraform can be used to deploy AWS resources. This example desmonstrates how to use Terraform to deploy a CI/CD pipeline for a SageMaker application.

The MLOps Pipeline is illustrated as below:



This folder consists of following files:

```
├── README.md
├── modelbuild_buildspec.yml
├── modelbuild_ci_pipline.tf
├── modelbuild_codebuild.tf
├── modelbuild_hooks.tf
├── modeldeploy_buildspec.yml
├── modeldeploy_cd_pipline.tf
├── modeldeploy_codebuild.tf
├── modeldeploy_hooks.tf
├── modeldeploy_testbuild.tf
├── events.tf
├── iam_roles.tf
├── main.tf
├── s3.tf
└── variables.tf
```

Two pipelines are built. One is model build CI pipeline, which trains and registers model. The other one is model deployment CD pipeline, which tests and deploys endpoints.

The model build CI pipeline (modelbuild_ci_pipeline.tf)includes the following stages:

- Source-GitHub source control
- modelbuild_codebuild - use buildspec to deploy SageMaker pipeline

The model deployment CD pipeline (modeldeploy_cd_pipeline.tf)includes the following stages:

- Source-GitHub source control
- modeldeploy_codebuild - use buildspec to create endpoint configurations
- Deploy Staging - deploy a testing endpoint
- Deploy Prod - deploy a prod endpoint

Terraform providers are defined in main.tf

Variables are defined in: variables.tf

Please update variables to your own case (github id, github token, s3 bucket must be filled. One bucket will be created for storing artifiacts. You can also change other varaible default values)

Web modelbuild_hooks for AWS and GitHub build repository, using random secret key si required for triggering automated builds.

Web modeldeploy_hooks for AWS and GitHub deploy repository, using random secret key si required for triggering automated builds.

Events captures new 'approved' registered model event and trigger the CD pipeline.

Finally we are defining roles with attached required policies, s3 buckets for build and deploy.

Pre-requirements:

- The Terraform CLI (https://learn.hashicorp.com/tutorials/terraform/install-cli)
- The AWS CLI installed. (https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2.html)
- An AWS account ("default" prolfe is used in this case.)
- Your AWS credentials. You can create a new Access Key on this page.
- Github respositories. Create two Github repositories. One resposity is for build repo (tf-sm-modelbuild), and the other one is for deploy repo(tf-sm-modeldeploy).

Instructions:

Step 1: Update "varaibles.tf" per your project

Step 2: Push the files from "tf-sm-modelbuild" and "tf-sm-modeldeploy" folders into the name plate repos.

Step 3: Run terraform code

To run Terraform, use the following simple commands:

```
terraform init
terraform apply
```

Step 4: Push updated code to github repositories to update pipeline.