There is a work for 3 challenges given.

1. **Challenge1**

Following components were used

AWS

GitHub

MyLaptop

GitBash

Ubuntu (using Virtualbox)

Jenkins

AWS CLI

Terraform

Docker

SampleWebApp (downloaded from internet)

Created following Jenkins items

**Pipeline - RunTerraformUpdatePipeline**

Infrastructure changes are getting creating.

Initializing -> Plan -> Apply

**Pipeline - DestroyTerraformPipeline**

This job will destroy everything created by above job.

Initializing -> Plan -> Destroy

**Pipeline - CIPipeline**

Pipeline is triggered as soon as commit is pushed to GIT. Pipeline is having 4 stages

Initializing -> BuildImage -> ImageUploading -> UpdateECS

Initializing

Here it checks for filename path for all commits done from last build.

If any filepath starts with app/\* then its application change and need to build image.

If anything else then do nothing. This mean it could be tarraform/jenkins/docker scripts change.

BuildImage

Performs some cleanup of on Ubuntu before building new image.

Here it used Dockerfile to download httpd and copying SampleWebApp and build image.

Also running new image on Ubuntu for me to have quick check on my laptop.

ImageIUploading

Login to AWS ECR

Remove image with tag "Latest" as we want to use same tag for new image.

Tag and Push image to ECR with 2 tags. 1 Build number and 1 latest. Basically it will put both tags to same image

UpdateECS

Retrive service definition from cluster to get task definition ID its using.

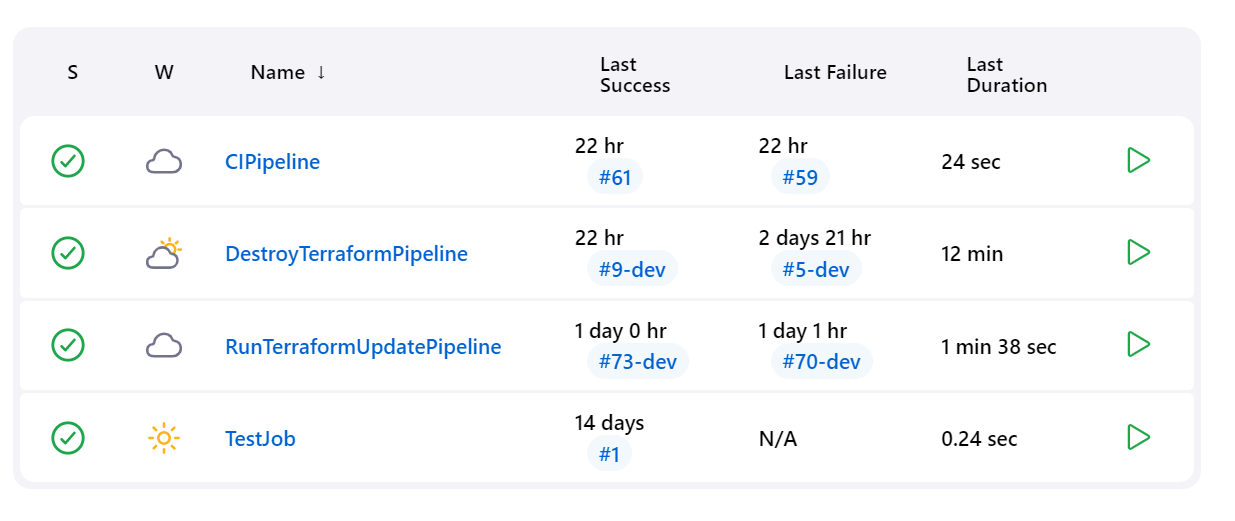
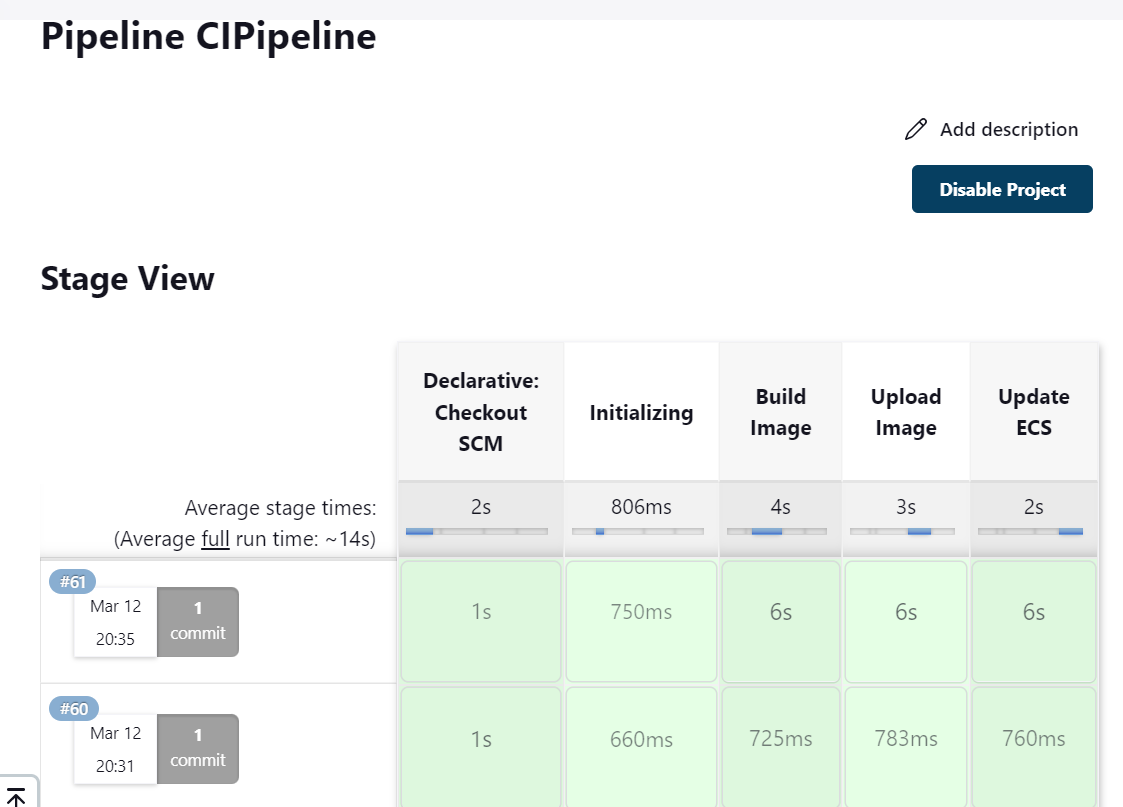
using this task definition ID, retrive task definition in json format.

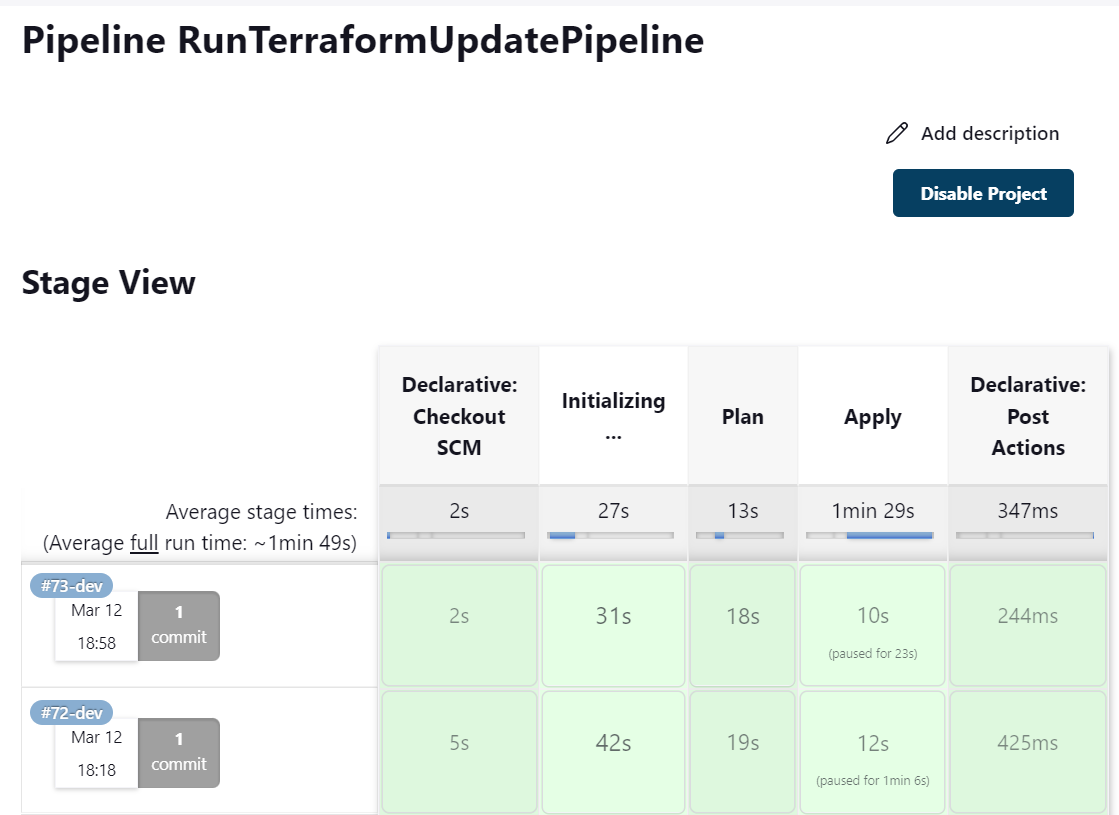
update task definition by replacing current ImageID with new ImageID from ECR.

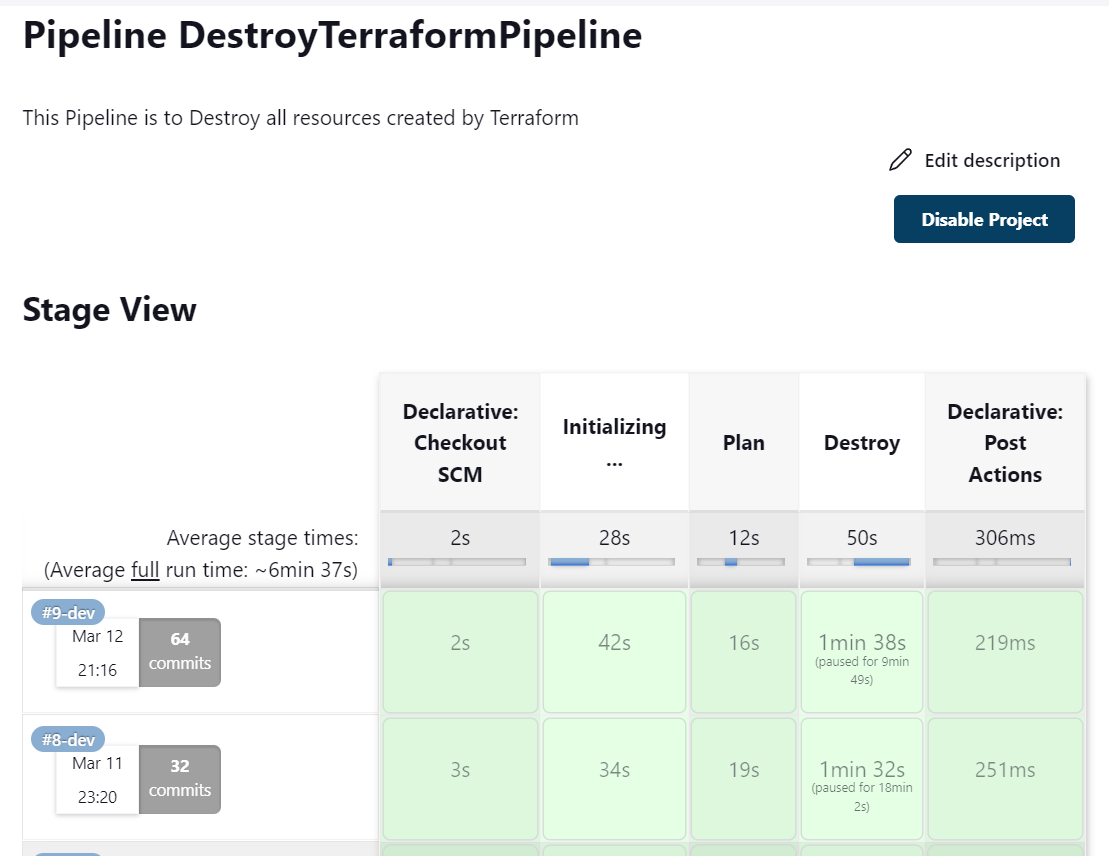
Register new task definition with new Image.

Update ECS service to start using new task definition.

ECS takes care of starting new task before stopping existing one. And hence you can have deployment without downtime.





1. Challenge2

challenge2.py is present under folder Challenge2

1. Challenge3