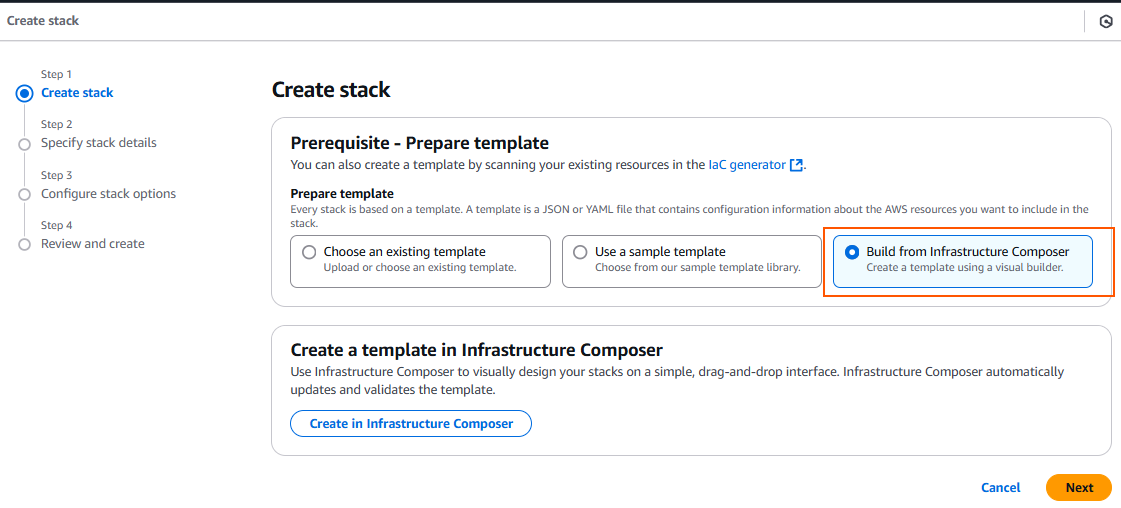
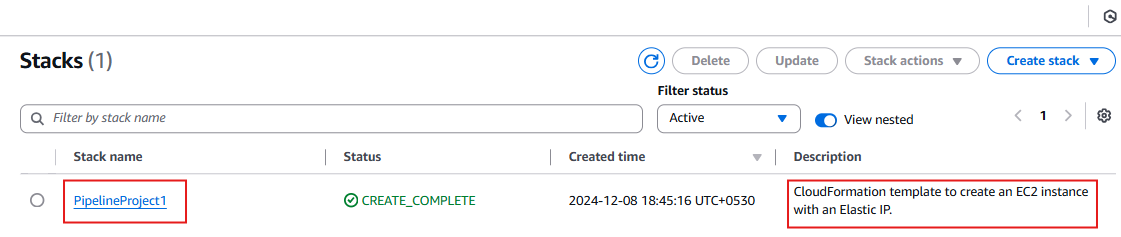
**Jenkin Pipeline project using Webhook trigger and CFT for infrastructure creation**

**Steps:**

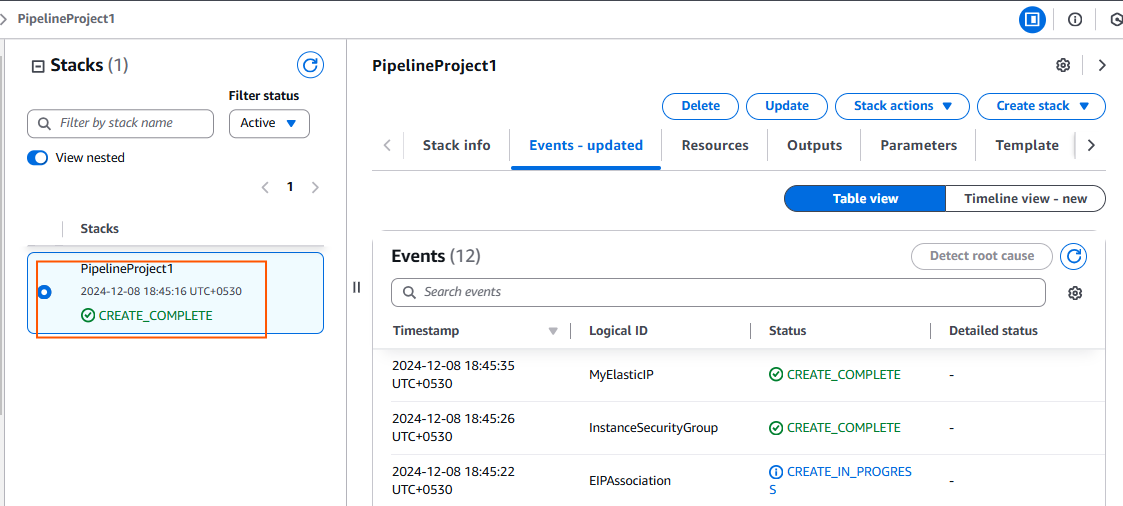
* Cloud Formation was used for Infrastructure Creation (Ec2 with Elastic IP).
* Pushed Jenkinsfile into Github.
* Created a connection between GitHub and Jenkins through Webhook.
* Generated pipeline syntax for Email notification.

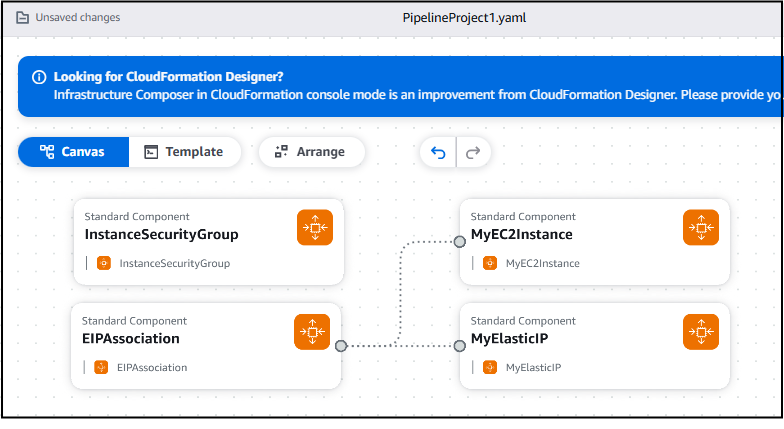
Stack: A stack is created based on a Cloud Formation template.

****

****

A stack is created based on a CloudFormation template.

****

Infrastructure Composer: ****

**YAML CODE:**

AWSTemplateFormatVersion: '2010-09-09'

Description: CloudFormation template to create an EC2 instance with an Elastic IP.

Parameters:

InstanceType:

Description: EC2 instance type

Type: String

Default: t2.micro

AllowedValues:

- t2.micro

ConstraintDescription: Must be a valid EC2 instance type.

KeyName:

Description: Name of an existing EC2 KeyPair to enable SSH access

Type: AWS::EC2::KeyPair::KeyName

ConstraintDescription: Must be the name of an existing EC2 KeyPair.

SSHLocation:

Description: The IP address range allowed to SSH into the instance

Type: String

Default: 0.0.0.0/0

AllowedPattern: (\d{1,3}\.){3}\d{1,3}/\d{1,2}

ConstraintDescription: Must be a valid CIDR range.

Resources:

MyEC2Instance:

Type: AWS::EC2::Instance

Properties:

InstanceType: !Ref InstanceType

KeyName: !Ref KeyName

ImageId: ami-053b12d3152c0cc71

SubnetId: subnet-0570e93c963933352

SecurityGroupIds: [sg-0343fdbe2b0c3d4a6]

Tags:

- Key: Name

Value: MyEC2Instance

MyElasticIP:

Type: AWS::EC2::EIP

Properties:

Domain: vpc

EIPAssociation:

Type: AWS::EC2::EIPAssociation

Properties:

InstanceId: !Ref MyEC2Instance

EIP: !Ref MyElasticIP

Outputs:

InstanceId:

Description: The ID of the created EC2 instance

Value: !Ref MyEC2Instance

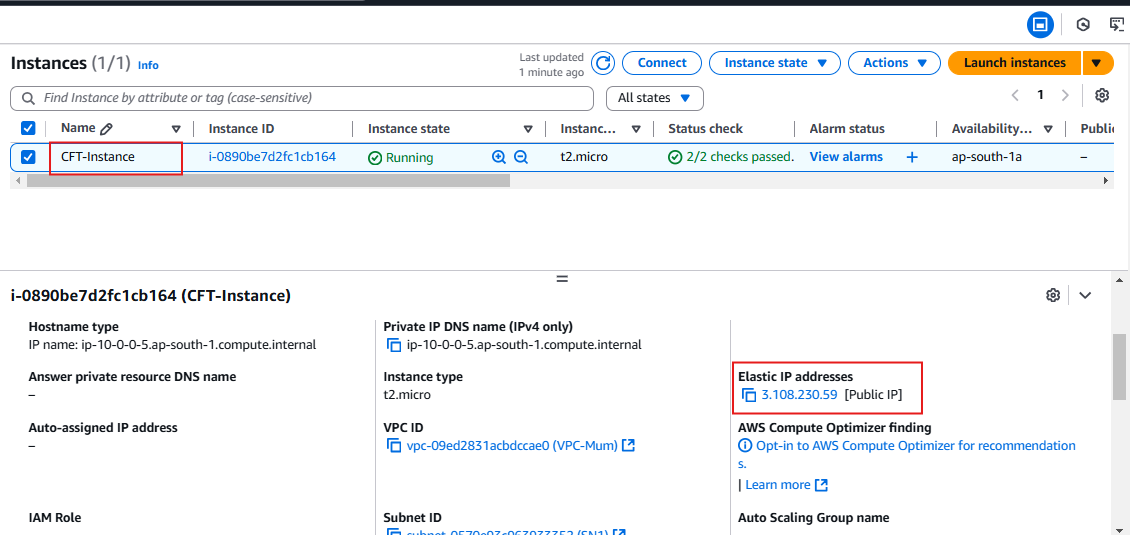
ElasticIPAddress:

Description: The Elastic IP address of the instance

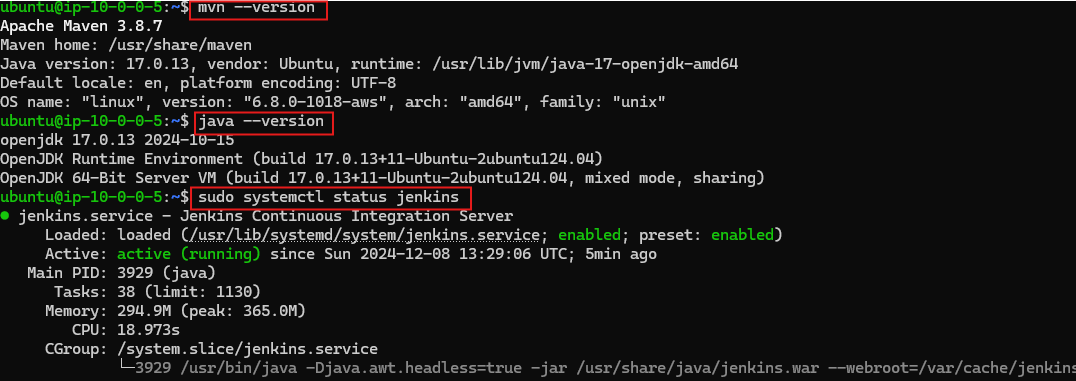
Value: !Ref MyElasticIP

According to the YAML code required infrastructure has been created.

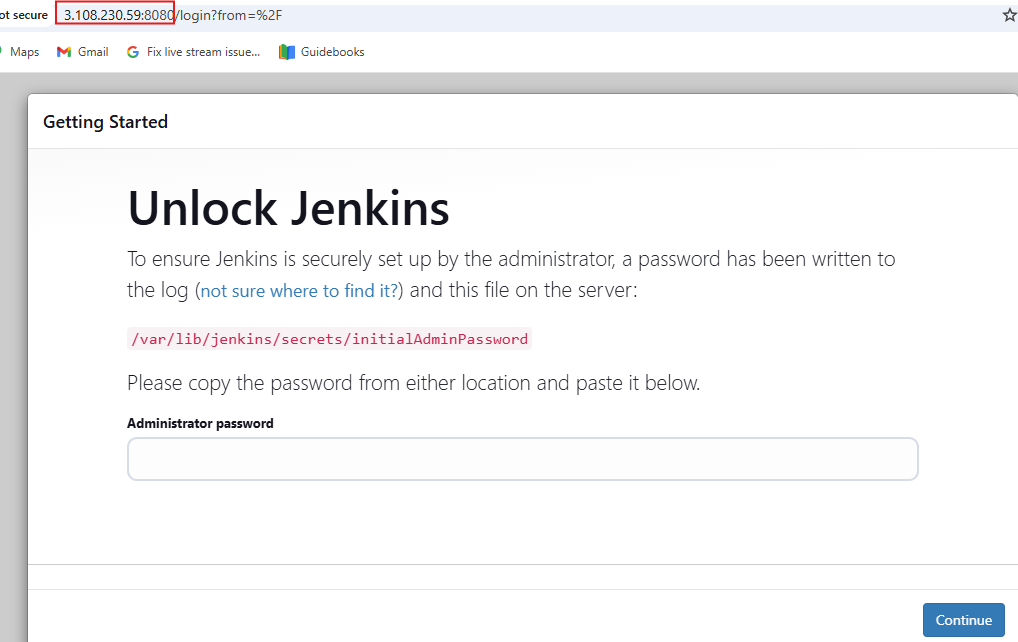
Instance created along with EIP.

****

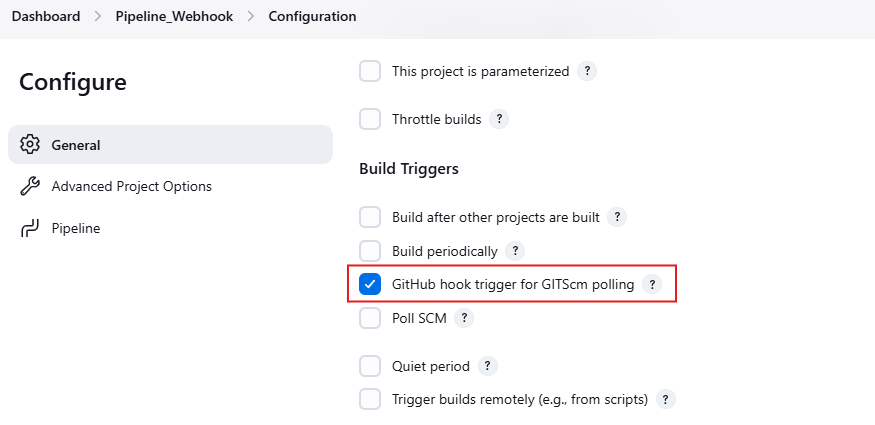
Installed Maven, Java and Jenkins in Ubuntu server.

****

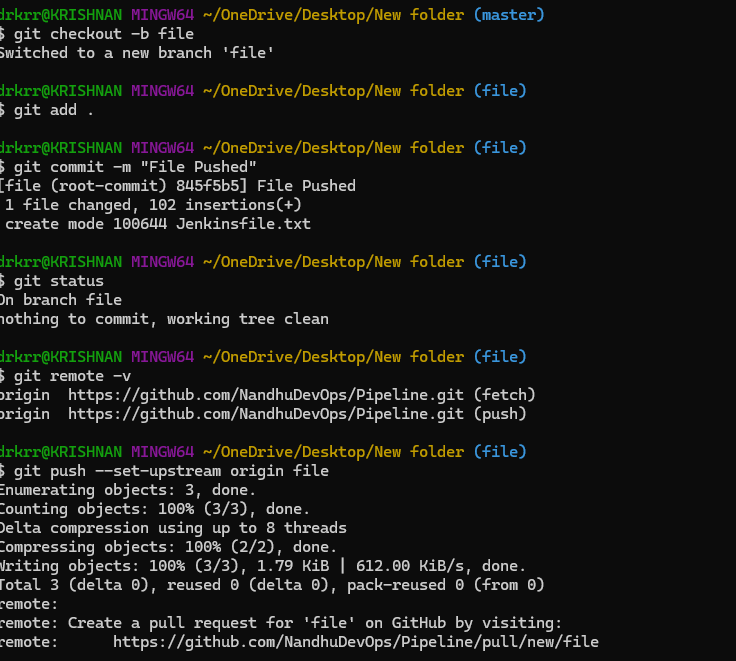
**Accessed Jenkin using EIP address.**

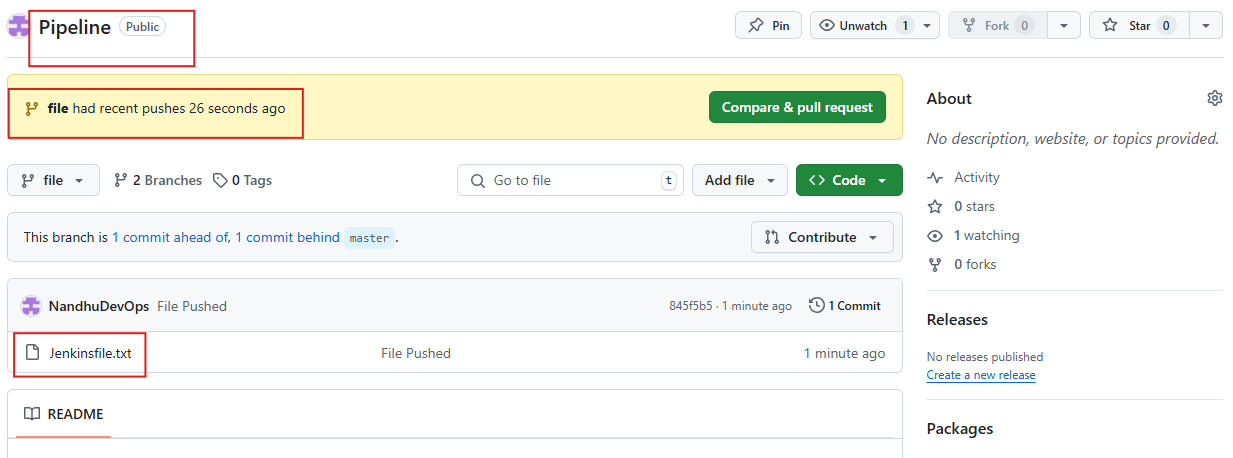
****

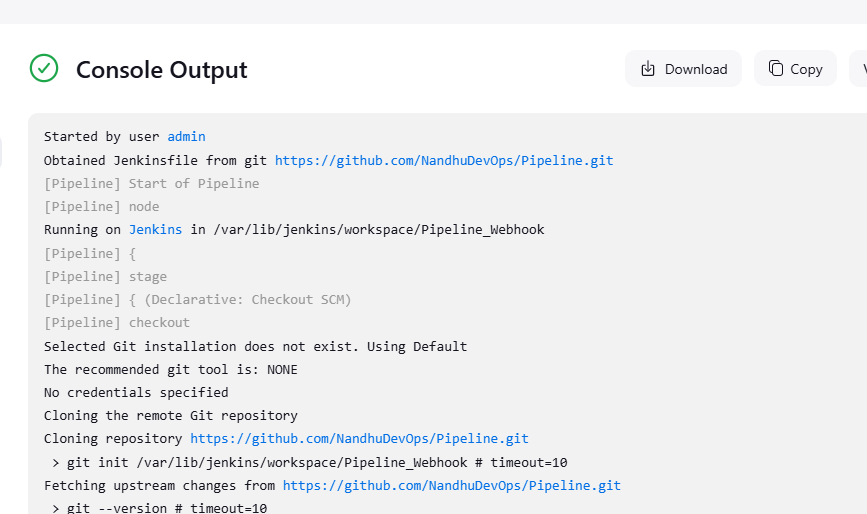
**Created New Pipeline project with Github hook trigger.**

****

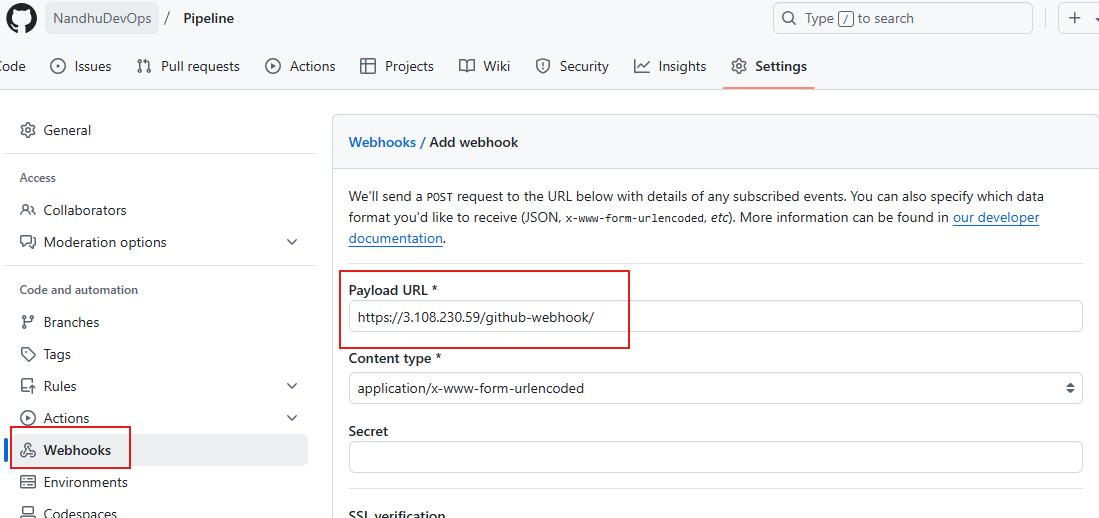
**Pushed Jenkinsfile from local to Github repository.**

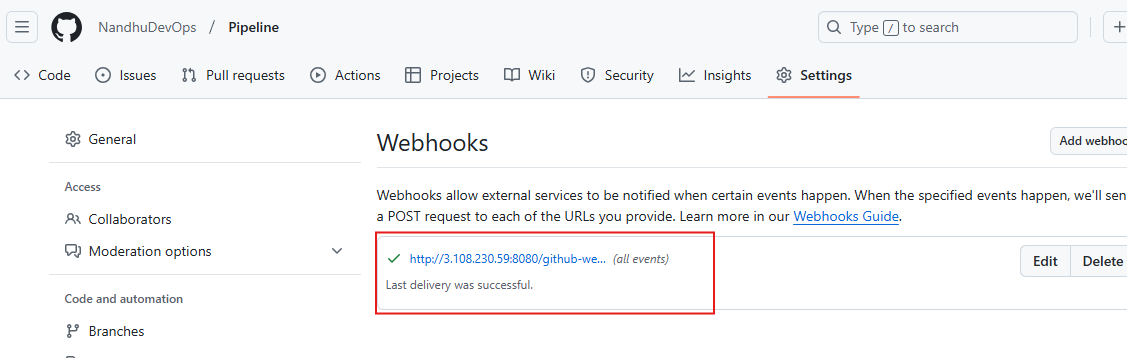
****

****

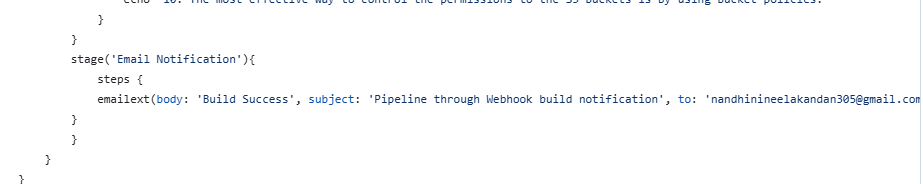
****

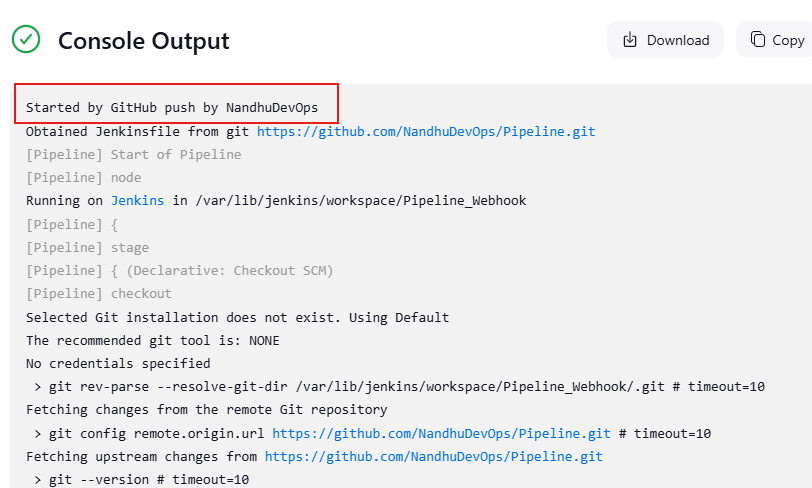
**Created Webhook for trigger in GitHub repo.**

****

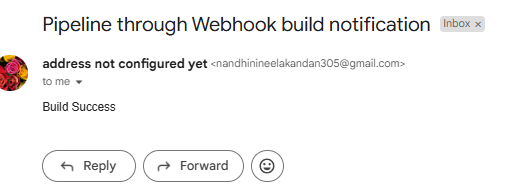
****

**Change made in Jenkinsfile to trigger mail, Webhook build the job automatically.**

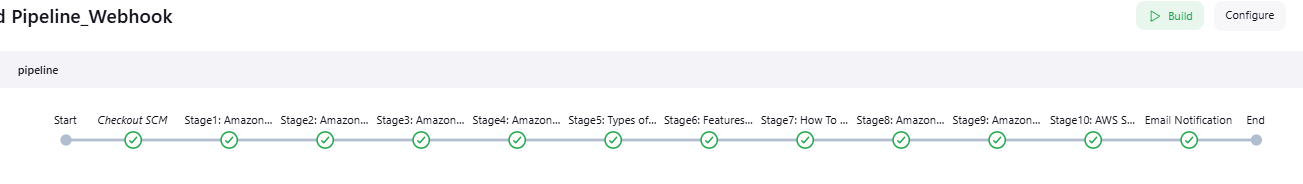
****

****

**Email Notification:**

****

**Stage View of Build:**

****

**Jenkinsfile:**

pipeline {

agent any

stages {

stage('Stage1: Amazon s3:') {

steps {

echo 'AWS S3 is a scalable storage service.'

}

}

stage('Stage2: Amazon S3 Used for:') {

steps {

echo '1.Data Storage'

echo '2.Backup and Recovery'

}

}

stage('Stage3: Amazon S3 bucket:') {

steps {

echo '1. Amazon S3 bucket is a fundamental Storage Container feature in AWS S3 Service.'

echo '2. It provides a secure and scalable repository for storing Objects.'

echo '3. Each S3 bucket name should be named globally unique.'

}

}

stage('Stage4: Amazon S3 works:') {

steps {

echo '1. Amazon S3 Buckets and Objects.'

echo '2. Amazon S3 Versioning and Access Control.'

echo '3. Bucket Policies and Life Cycles'

echo '4. Keys and Null Objects'

}

}

stage('Stage5: Types of S3 Storage Classes:') {

steps {

echo '1. Standard'

echo '2. Standard Infrequent Access (Standard IA)'

echo '3. Intelligent Tiering'

echo '4. One Zone Infrequent Access'

echo '5. Reduced Redundancy Storage'

}

}

stage('Stage6: Features of Amazon S3:') {

steps {

echo '1. Durability'

echo '2. Availability'

echo '3. Server-Side-Encryption (SSE)'

echo '4. File Size support'

echo '5. Infinite storage space'

echo '6. Pay as you use'

}

}

stage('Stage7: How To Use an Amazon S3 Bucket:') {

steps {

echo '1. Login into the Amazon account, search form S3, and click on the S3.'

echo '2. click on the option "Create bucket” and configure all the options.'

echo '3. After configuring the AWS bucket now upload the objects into the buckets.'

echo '4. AWS CLI command: aws s3 cp <local-file-path> s3://<bucket-name>/(to upload the object).'

echo '5. You can control the permissions of the objects uploaded into the S3 buckets.'

echo '6. You can make the bucket public or private by default the S3 buckets will be in private mode.'

echo '7. You need to enable the S3 access logging to record who requested the objects.'

}

}

stage('Stage8: Amazon S3 Buckets and Objects:') {

steps {

echo '1. Data, in S3, is stored in buckets.'

echo '2. Each bucket will have its own policies and configurations.'

echo '3. Bucket Names must be unique.'

echo '4. There is a limit of 100 buckets per AWS account.'

echo '5. Fundamental entity type stored in AWS S3.'

echo '6. You can store as many objects.'

echo '7. The maximum size of an AWS S3 bucket is 5TB.'

echo '8. It consists of Key, Version ID, Value, Tag.'

}

}

stage('Stage9: Amazon S3 Versioning and Access Control:') {

steps {

echo '1. Versioning means always keeping a record of previously uploaded files in S3.'

echo '2. Points to Versioning are not enabled by default.'

echo '3. Once enabled, it is enabled for all objects in a bucket.'

echo '4. Versioning keeps all the copies of your file, it adds cost for storing multiple copies of your data.'

echo '5. Versioning is helpful to prevent unintended overwrites and deletions.'

echo '6. Objects with the same key can be stored in a bucket if versioning is enabled.'

echo '7. Access control lists: A document verifying access to S3 buckets from outside your AWS account.'

echo '8. An ACL is specific to each bucket.'

echo '9. You can utilize S3 Object Ownership.'

}

}

stage('Stage10: AWS S3 Bucket Permissions:') {

steps {

echo '1. Bucket policies can be attached directly to the S3 bucket and they are in JSON format.'

echo '2. Permissions can be granted to the users who can access the objects in the bucket.'

echo '3. You can create the bucket policy by using Python.'

echo '4. ACLs are legacy access control mechanisms for S3 buckets.'

echo '5. By using ACL you can grant the read, and access to the S3 bucket.'

echo '6. IAM policies manage the permissions to the users, groups, and resources.'

echo '7. IAM policy can be attached to an IAM entity.'

echo '8. Amazon S3 follows a pay as you go model on charging for storage of Objects files.'

echo '9. Based on the amount of data and total volumes of data transfers and requests.'

echo '10. The most effective way to control the permissions to the S3 buckets is by using bucket policies.'

}

}

stage('Email Notification'){

steps {

emailext(body: 'Build Success', subject: 'Pipeline through Webhook build notification', to: 'nandhinineelakandan305@gmail.com')

}

}

}

}