**Use Jenkins and AWS Code Deploy as a CI/CD Tool**

**Prepared by**

**Yahser Khan**

**Continuous Integration** is a software development practice where continuous changes and updates in code base are integrated and verified by an automated build scripts using various tools.

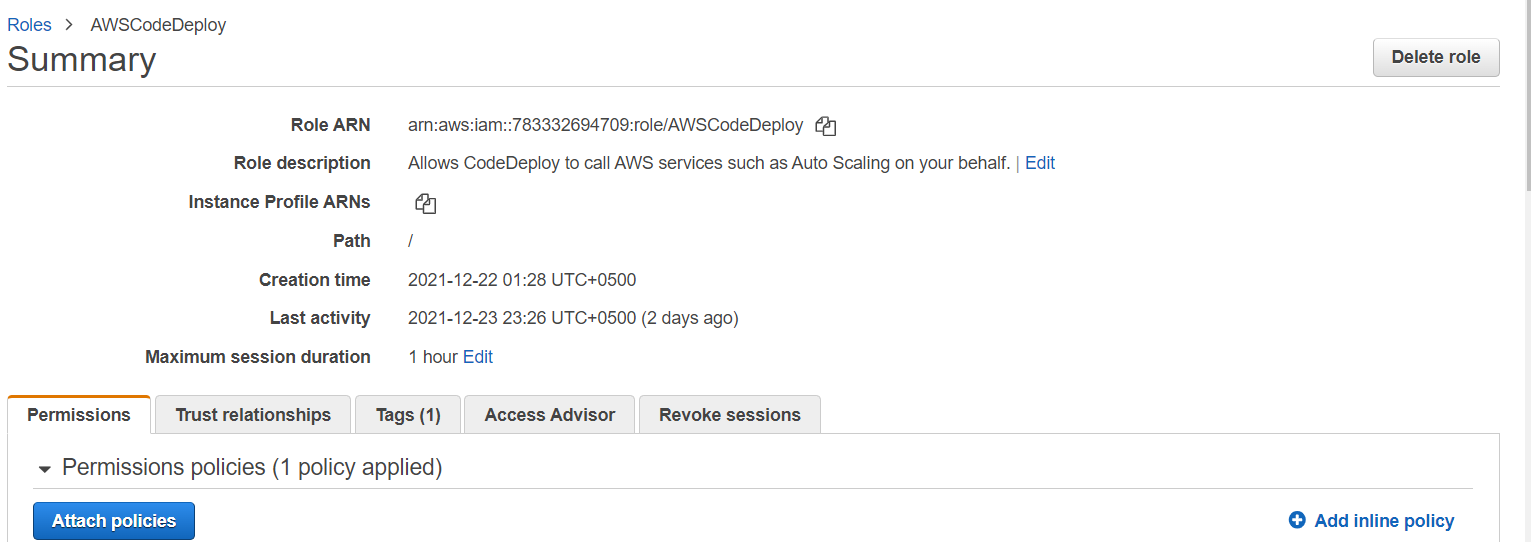
**Continuous Deployment** is also a software development practice whose role is to automatically deploy the code to the specified server and application folder once the code is been integrated successfully.

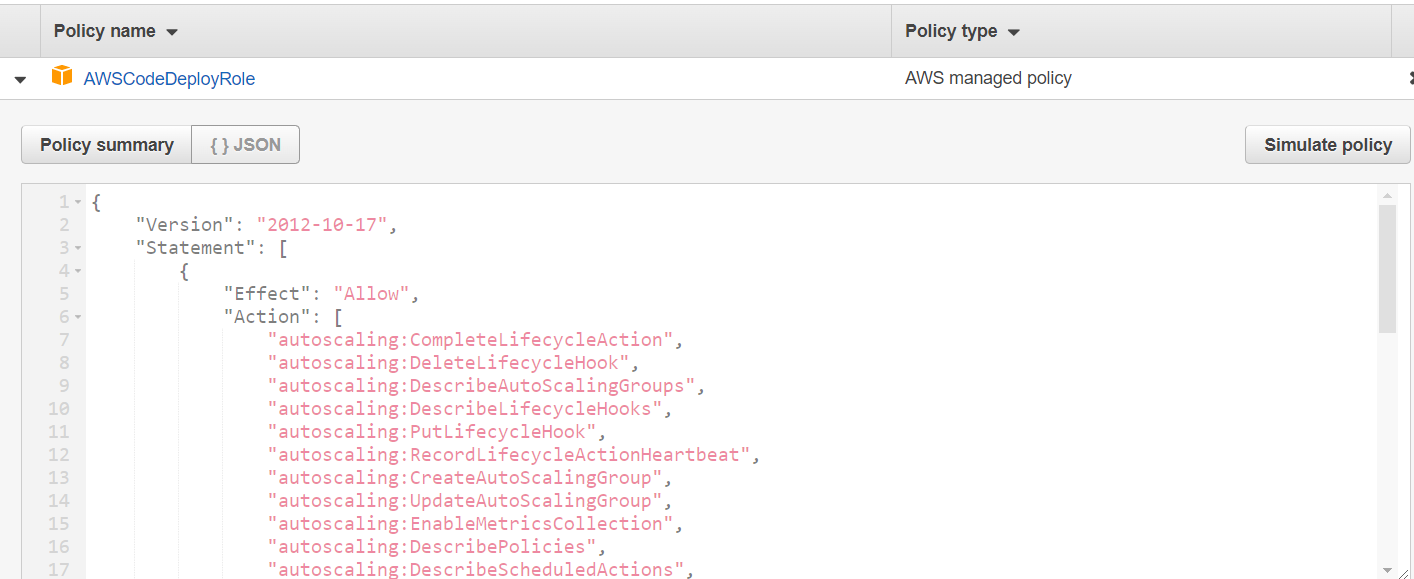
In this Tutorial I have used Jenkins open-source automation server to deploy with AWS Code Deploy and creating a CI/CD Pipeline. The CI/CD Pipeline is triggered by code changes pushed to your GitHub Repo, automatically fed into Jenkins then the application is deployed using Code Deploy. We will use Amazon Web Service as the Cloud platform, GitHub for the code Repository, Jenkins for Continuous Integration (CI), and AWS Code Deploy Service for the continuous Delivery (CD).

We need to create IAM Roles, there are two types of IAM roles that are required. **Service Role** and **Instance Role.**

1. **Service Role:** Service Role access will be granted to **AWS Code Deploy** to read through your EC2 Instance tags. below are the snapshot.

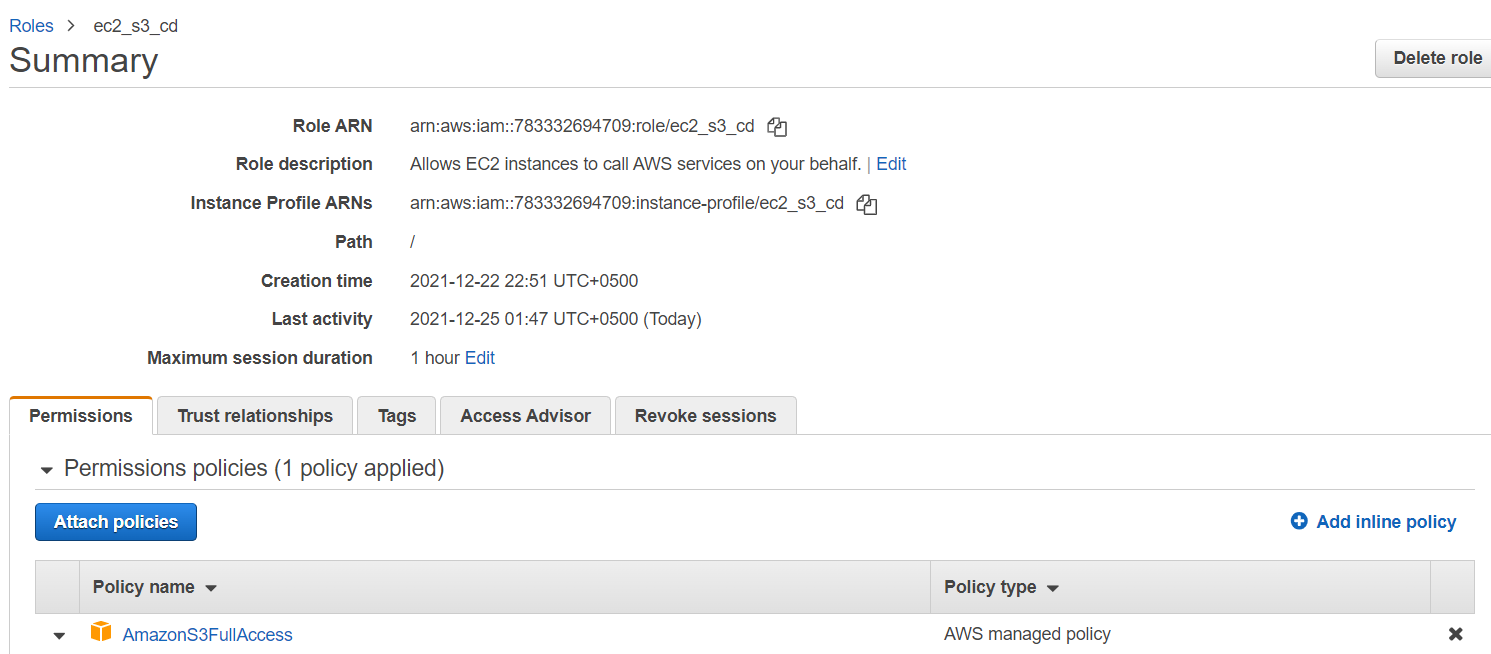
**AWS Code Deploy (Role Snapshot)**

****



1. **Instance Role:** EC2 instanceneed to be launched with proper permissions to accessfiles from S3 Bucket, so the instance Profile Role will give access to EC2 to read code from S3 Buckets, below are the snapshot.

**ec2\_S3\_cd (Role Snapshot)**

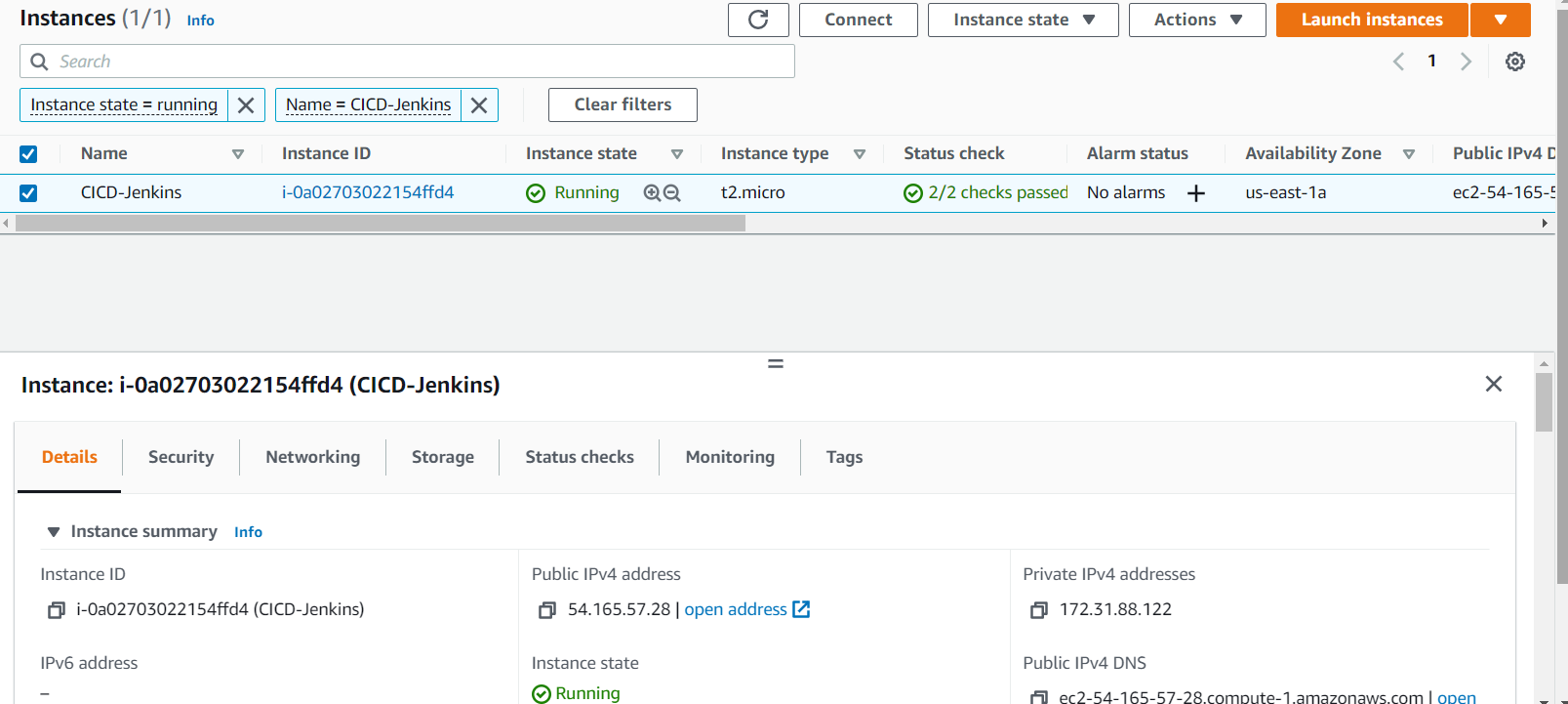
****

****

**Step 1: Setup AWS EC2 instance**

Created EC2 instance, I have selected **Amazon Linux 2 AMI** with instance type **t2.micro**, on configure instance tab on **IAM Role**, I have attached the **ec2\_S3\_cd Role**, below are the snapshot.

**Ec2 instance Snapshot**

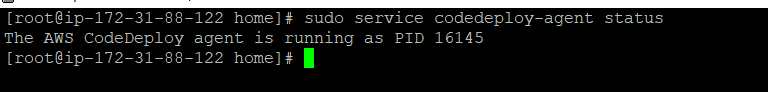




**Step 2: Install Code Deploy Agent on EC2**

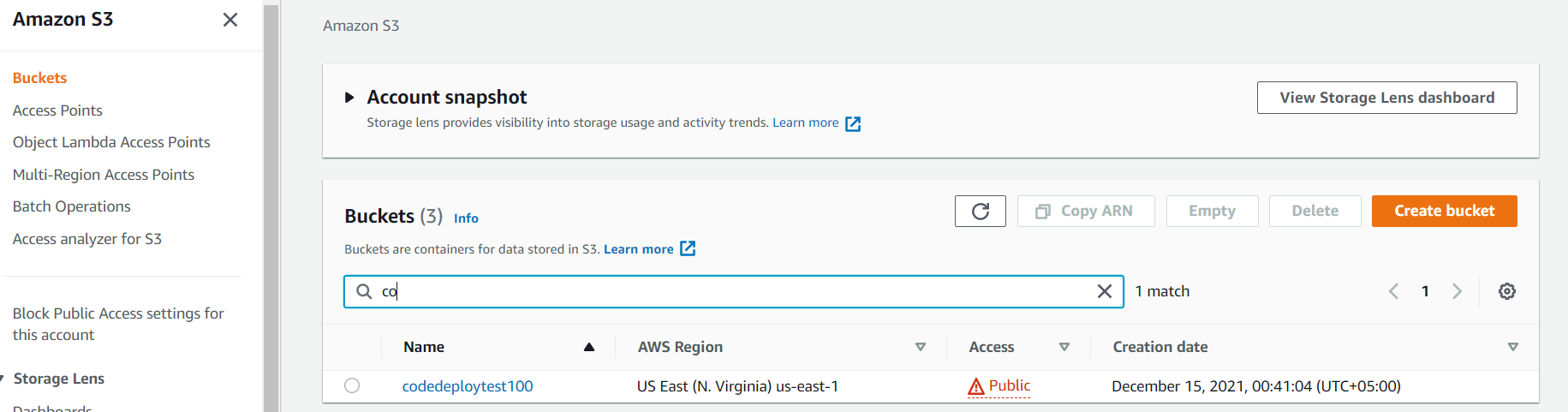
The instance we have created must have a Code Deploy agent installed on it in order to proceed to the code deployment, I have installed CodeDeploy on EC2 instance, below are the snapshot of codedeploy-agent status

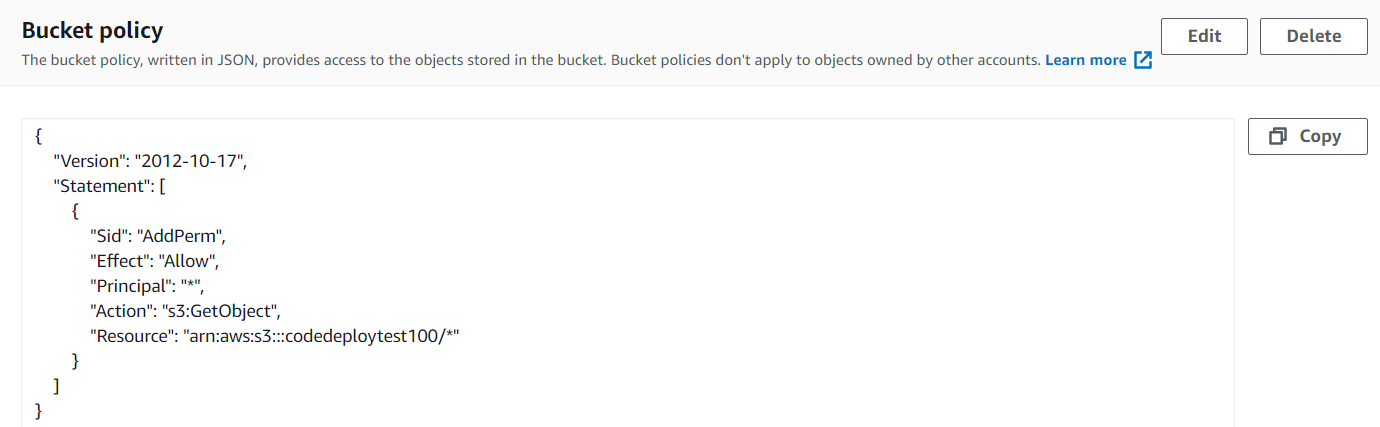
**CodeDeploy-agent status**

****

**Step 3: Create an S3 Bucket for the Application Code**

The application code should be uploaded to a location where CodeDeploy can deploy it. In this case, the AWS S3 bucket is used, below are the snapshot of S3 Bucket.



****

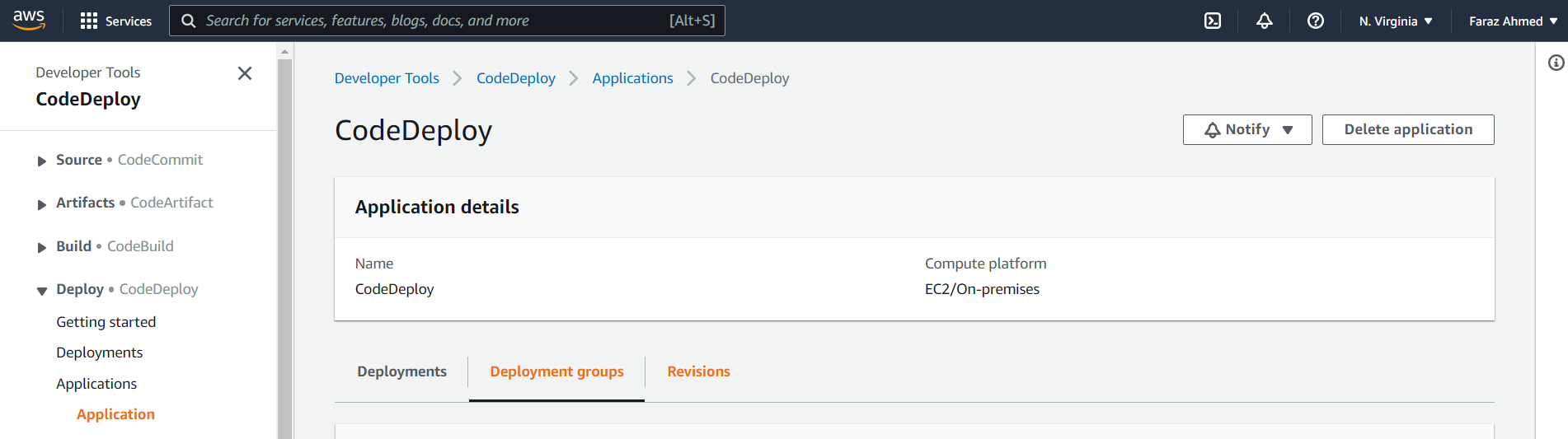
**Step 4: Configure AWS Code Deploy Service**

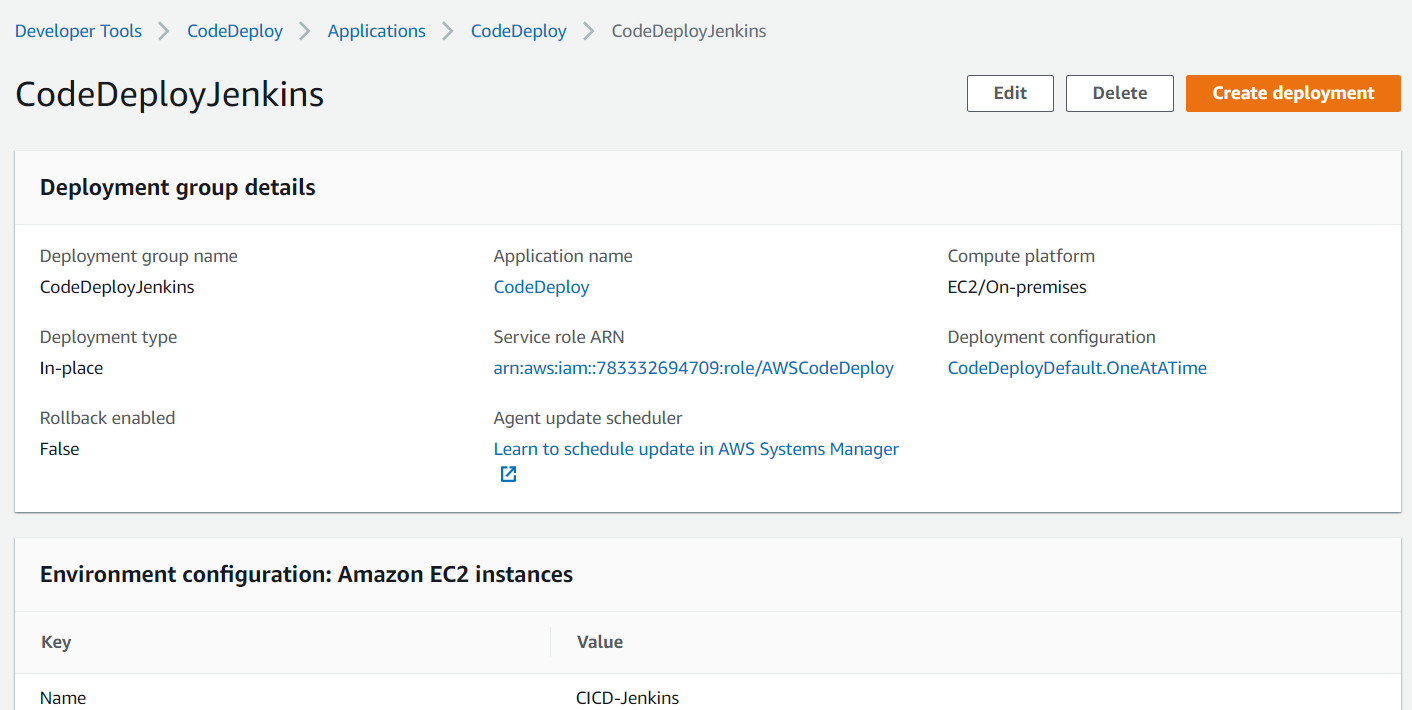
We need the following requirements to use the AWS Code Deploy Service.

1) An application

2) A Deployment Group

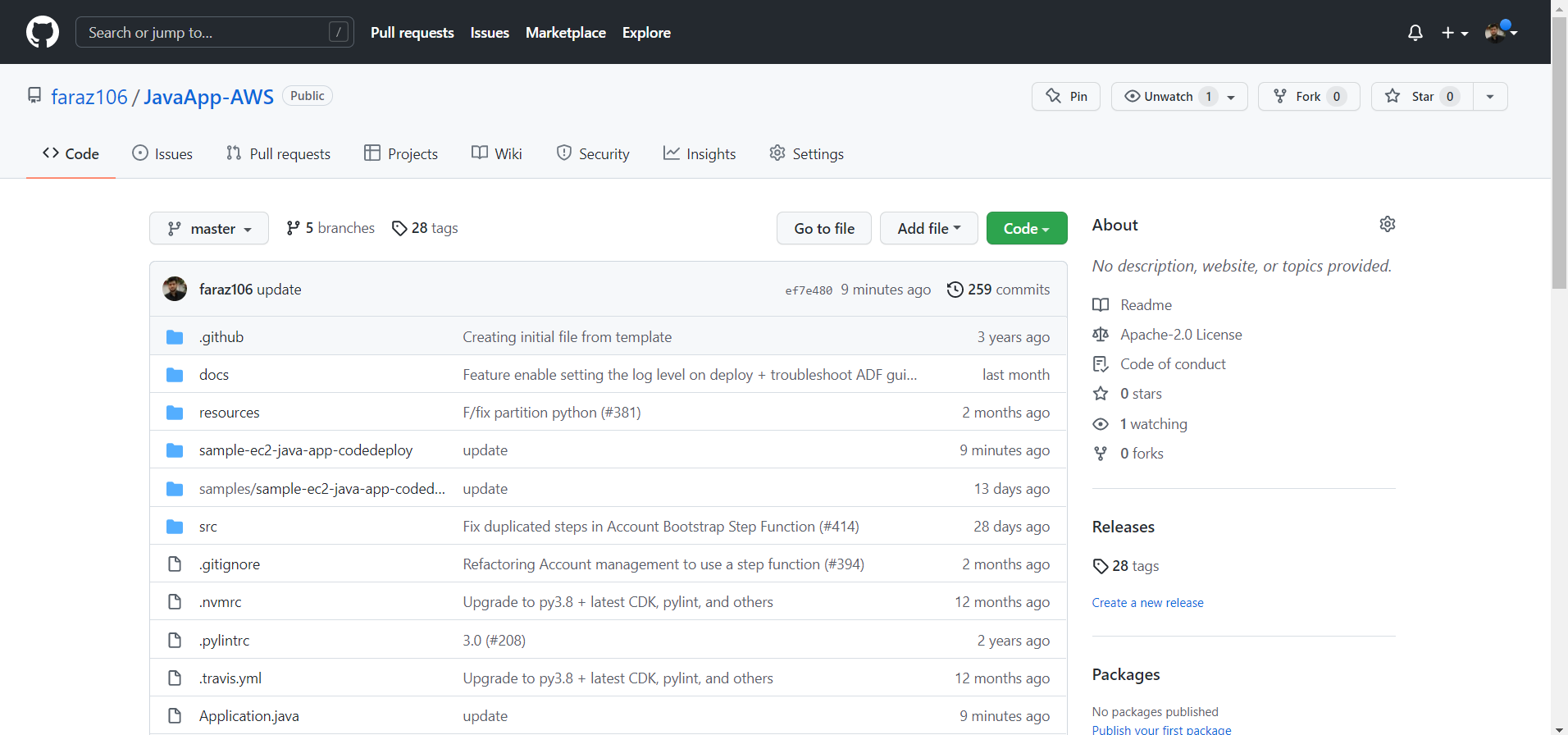
I have created Application and create Deployment Group, below are the Snapshot of **Application** and **Deployment Configuration**

****

****

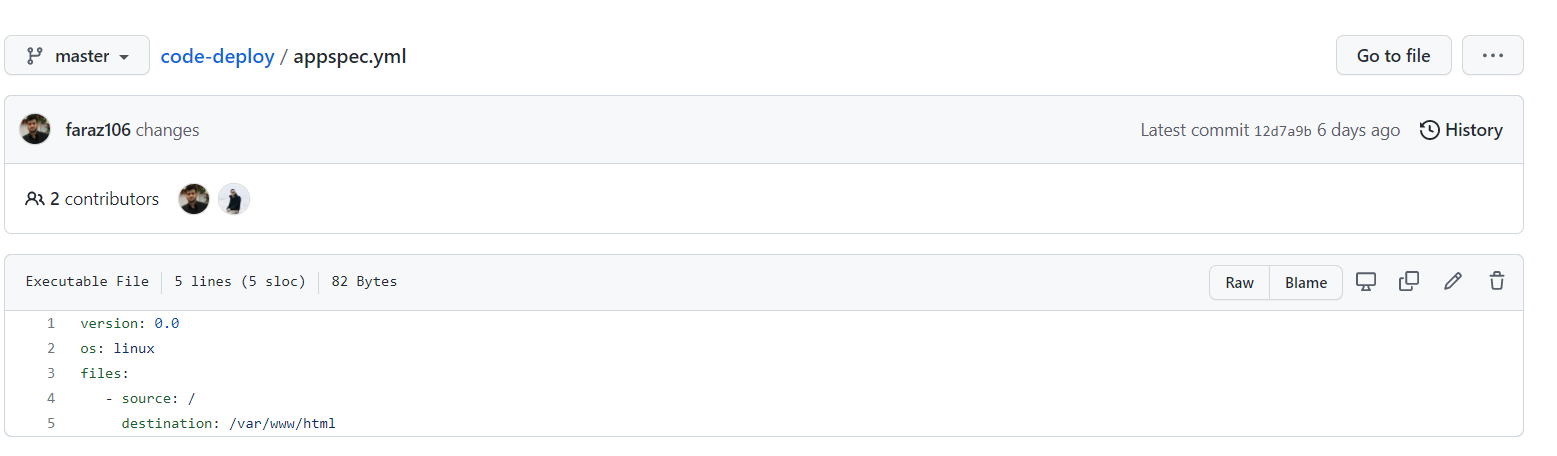
**Step 5: Git Hub**

In this Tutorial I am using GitHub for the code repository, below are my GitHub Java App and Repo URL Snapshot

****

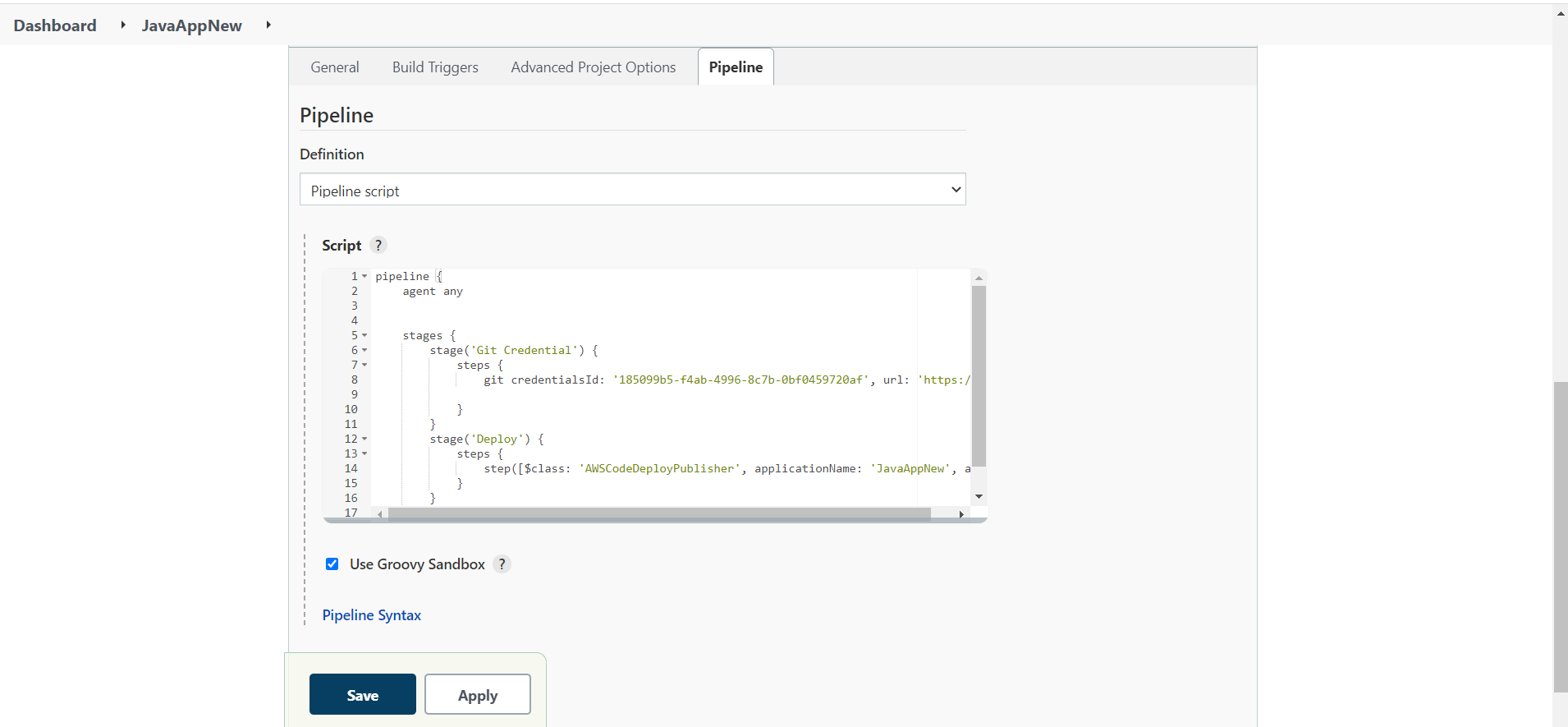
**Appspec.yml**

The Application specification file (AppSpec file) is a YAML formatted file used by AWS CodeDeploy to manage Deployment, below are the snapshot of AppSpec file.

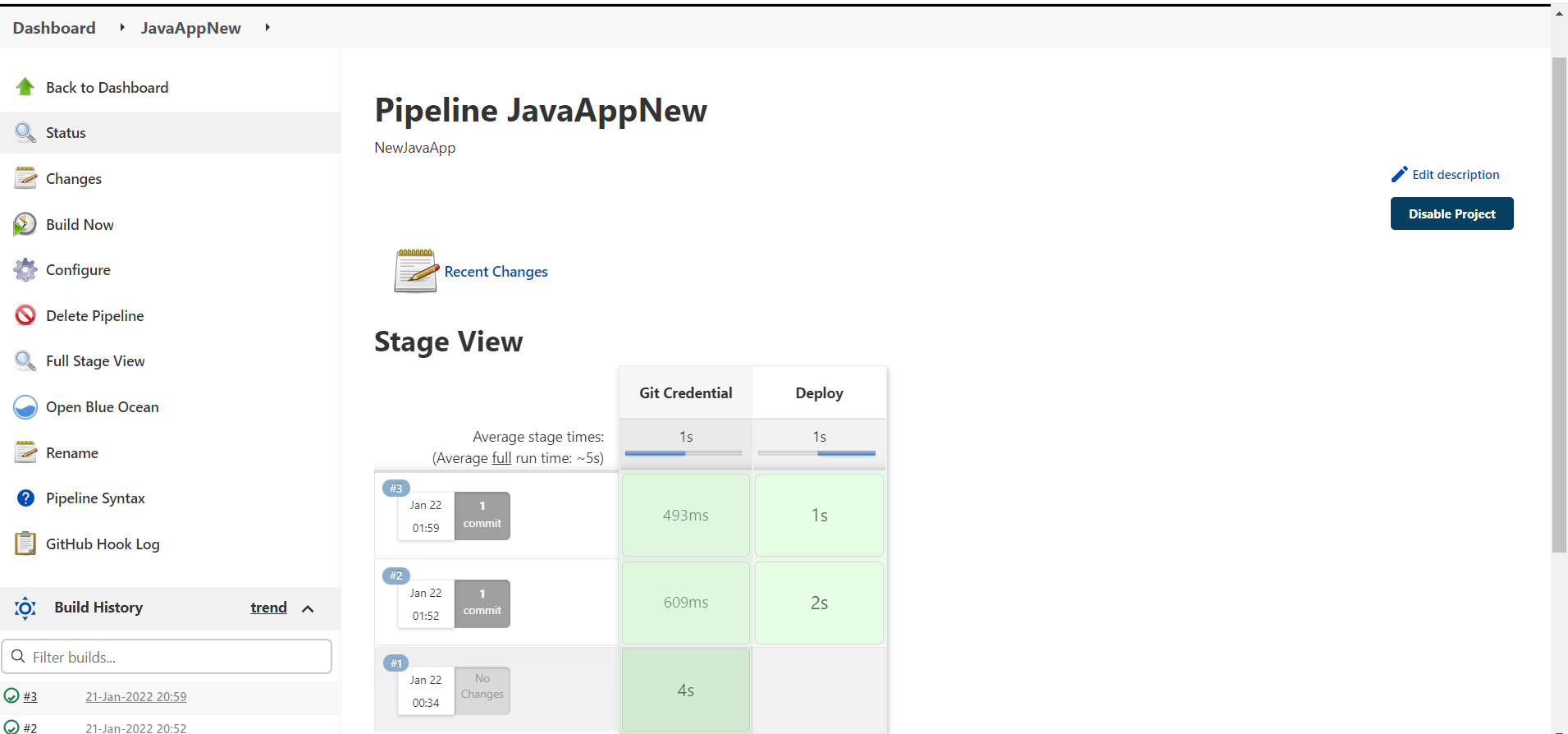


**Step 6: Configure Jenkins with AWS Code Deploy Plugin**

I have installed AWS Code Deploy plugin in Jenkins, I have create a project with name **JavaAppNewDeploy** and select Pipeline job, I have created groovy script for multiple stages, below are the snapshot of pipeline script.



**Stage View Screenshot**



**Testing the Whole CI/CD Pipeline**

To test the whole solution, put an Java application on GitHub Repository, below are the sample code

**https://github.com/faraz106/JavaApp-AWS.git**

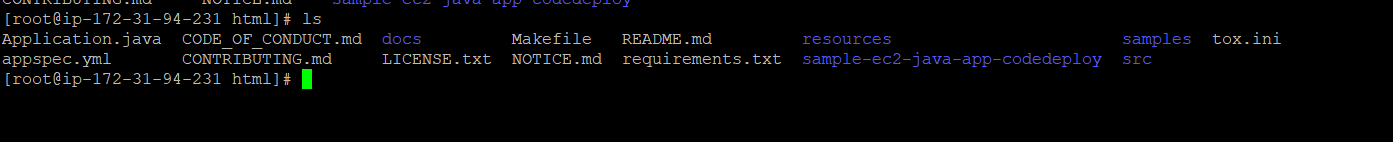
Once code push to the Repository, Jnkins Jobs runs (CI) automotically and after that deployment job start and successfully deployed Java App to the instance below are the snapshot of Jenkins and deployment snapshot.

**Jenkins CICD Snapshot**



**Java App Deployment Snapshot**

JavaApp successfully deployed to the instance, and we can see the github Java App on to the html directory below are the snapshot



After hit the publicIP of ec2 instance along with application name **http://3.93.151.168/Application.java** below are the snapshot

