**AWS CodeCommit:**

**Service Purpose:** CodeCommit is a fully managed source control service that makes it easy for teams to host secure and scalable Git repositories.

Key Features:

Git-Compatible: It's compatible with Git, so you can use Git commands and tools with it.

**Version Control:** It provides version control for your source code, allowing multiple developers to collaborate on code changes.

**Integration:** **It integrates with other AWS services like AWS CodePipeline and AWS CodeBuild for automated CI/CD (Continuous Integration/Continuous Deployment) pipelines**.

**Security:** **CodeCommit** provides access control and encryption features to secure your source code.

**Scalability:** It can scale to handle large code repositories.

**AWS CodeDeploy:**

**Service Purpose**: CodeDeploy is a deployment service that automates software deployments to various compute resources, including Amazon EC2 instances, Lambda functions, and on-premises servers.

Key Features:

**Deployment Automation**: It automates the deployment process, making it easy to roll out new code updates and rollbacks.

**Deployment Hooks:** CodeDeploy allows you to specify hooks or scripts that run during different stages of deployment, such as "before install" or "after install."

**Platform Support:** It works with a variety of application types and platforms, not just EC2 instances.

**Blue/Green Deployments:** CodeDeploy supports blue/green deployments, which enable you to release new versions of your application with minimal downtime.

Integration: It can be integrated with other AWS services like CodePipeline for complete CI/CD pipelines**.**

**AWS CodePipeline:**

**Service Purpose**: CodePipeline is a continuous integration and continuous delivery (CI/CD) service that automates the building, testing, and deployment of applications.

Key Features:

**Workflow Automation**: It enables you to define and automate the entire software release process, including source code integration, testing, and deployment.

**Integration:** CodePipeline integrates with various AWS services, including CodeCommit, CodeBuild, CodeDeploy, and Lambda, as well as third-party tools, to create a customizable pipeline.

**Multiple Stages:** Pipelines consist of multiple stages, and each stage can represent a different part of your software delivery process.

**Artifact Storage**: It can store and manage build artifacts and deployment files.

Visualization: CodePipeline provides a visual representation of your CI/CD pipeline and supports manual approval steps.

**Step: 1 Prerequisites**

1. Download and install Git on you local system(Linux and Windows)
2. Configure GIT on your system
   1. git config --global user.email <your email id>
   2. git config --global user.name "your name"

1. Create New AWS IAM user (Or use existing admin user)

a. Attach IAM policy "AWScodecommitFullAccess"

b. Attach IAM policy "AWScodePipelineFullAccess"

c. Security Credentials -->> HTTPS Git Credentials -->>Generate and Save the credentials

1. Create Service Role for **CodeDeploy**

a. Create IAM role for CodeDeploy service

b. Attach existing IAM policy "AWSCodeDeployRole"

1. Create IAM Role for Ec2 to download logs from S3

a. Create IAM role for EC2 service to attach "AttachS3readonlyAccess"policy.

**Step: 2 Configure Code Commit Respository**

1. Go to CodeCommit Service ( ex: Use North Virginia Region )
   1. Create new code repository with name "**project1**"
   2. Copy the Clone URL >>Clone HTTPS
2. Clone Git Repository locally
   1. Git clone <URL>
   2. Provide username/Password (**This should create empty repository**)
3. Download Sample Application
   1. Download application locally
   2. Unzip and move all files and folders inside your local repository "**Alpha"** directory
   3. Change your CMD to Demo-Project ( master )
   4. git status
   5. git add -A
   6. Git commit -m "Added sample application files"
   7. Git push

**STEP: 3 LAUCH EC2 INSTANCE TO HOST APPLICATION**

1. Launch EC2 instance with below Requirements

a. AMI: **Amazon Linux2**

b. Instance Type: **T2 Micro**

c. VPC: Default

d. IAM Role : **Role that we created in step #1**

e. in Advance option ,paste the below user data install codeDeploy Agent

#!/bin/bash

yum -y update

yum install -y ruby

yum install -y aws-ch

cd/home/ec2-user

aws s3 cp://aws-codedeploy-ap-south-t/latest/install . --region ap-south-1

chmod +X ./install

./install auto

f.Tags: key name, value: Demo

G.Secuity Group: Open port No#22 & 80 for my ip or internet

1. Launch

**STEP: 4 CREATE APPLICATION IN CODE DEPLOY**

* 1. Go to CodeDeploy service
  2. Create New Application
     1. Name:**Demo-Alpha-Application**
     2. Compute Platform: **Ec2/On-premises**
     3. Create Application
  3. Deployment Groups
     1. Name**: Demo-Alpha-Group**
     2. Deployment Type:**In-place DeploymSent**
     3. Environment configuration :
        + 1. Amazon EC2 Instances
          2. Enter the Key=Name and Value=Demo (That you created while launching EC2 instance)
  4. Deployment Configuration
     1. Select CodeDeployDefault.**OneAtaTime**