**CI/CD Pipeline with AWS**

**Step 1: Set Up IAM Roles**

**1.1 Create IAM Role for EC2 (CodeDeploy EC2 Role)**

1. **Navigate to the IAM Console:**
   * Go to the [IAM console](https://console.aws.amazon.com/iam/).
2. **Create a Role for EC2:**
   * **Service:** EC2
   * **Permissions:** Attach AWSCodeDeployRole, AmazonS3ReadOnlyAccess, and CloudWatchAgentServerPolicy policies.
   * **Role Name:** CodeDeploy-EC2-Role.
3. **Attach the Role to EC2:**
   * Go to your EC2 instance, choose **Actions > Security > Modify IAM Role**, and attach the CodeDeploy-EC2-Role.

**1.2 Create IAM Role for AWS CodeDeploy**

1. **Navigate to the IAM Console:**
   * Go to the [IAM console](https://console.aws.amazon.com/iam/).
2. **Create a Role for CodeDeploy:**
   * **Service:** CodeDeploy
   * **Permissions:** Attach AWSCodeDeployRole.
   * **Role Name:** CodeDeploy-Service-Role.
3. **Assign the Role to CodeDeploy Application:**
   * While setting up CodeDeploy, select CodeDeploy-Service-Role as the service role.

**1.3 Create IAM Role for AWS CodePipeline**

1. **Navigate to the IAM Console:**
   * Go to the [IAM console](https://console.aws.amazon.com/iam/).
2. **Create a Role for CodePipeline:**
   * **Service:** CodePipeline
   * **Permissions:** Attach AWSCodePipelineFullAccess, AWSCodeDeployRole, AWSCodeBuildAdminAccess, and AmazonS3FullAccess.
   * **Role Name:** CodePipeline-Service-Role.
3. **Assign the Role to CodePipeline:**
   * While setting up CodePipeline, select CodePipeline-Service-Role as the service role.

**1.4 Create IAM Role for AWS CodeBuild**

1. **Navigate to the IAM Console:**
   * Go to the [IAM console](https://console.aws.amazon.com/iam/).
2. **Create a Role for CodeBuild:**
   * **Service:** CodeBuild
   * **Permissions:** Attach AmazonS3ReadOnlyAccess, AWSCodeDeployDeployerAccess, and CloudWatchLogsFullAccess.
   * **Role Name:** CodeBuild-Service-Role.
3. **Assign the Role to CodeBuild Project:**
   * While setting up CodeBuild, select CodeBuild-Service-Role as the service role.

**Step 2: Set Up Your GitHub Repository**

1. **Create a Repository on GitHub:**
   * Push your project files to GitHub, including the appspec.yml, buildspec.yml, index.html, install\_nginx.sh, and start\_nginx.sh.
2. **Example Files:**
   * **appspec.yml:**

yaml

Copy code

version: 0.0

os: linux

files:

- source: /

destination: /var/www/html

hooks:

AfterInstall:

- location: install\_nginx.sh

timeout: 300

runas: root

ApplicationStart:

- location: start\_nginx.sh

timeout: 300

runas: root

* + **buildspec.yml:**

yaml

Copy code

version: 0.2

phases:

install:

commands:

- echo Installing NGINX

- sudo apt-get update

- sudo apt-get install nginx -y

build:

commands:

- echo Build started on `date`

- cp index.html /var/www/html

post\_build:

commands:

- echo Configuring NGINX

artifacts:

files:

- "\*\*/\*"

* + **index.html:**

html

Copy code

<html>

<head>

<title>AWS Website</title>

</head>

<body>

<h1>Welcome to AWESOME AWS website</h1>

<p>This is a simple website hosted on AWS</p>

</body>

</html>

* + **install\_nginx.sh:**

bash

Copy code

#!/bin/bash

sudo apt-get update

sudo apt-get install nginx -y

* + **start\_nginx.sh:**

bash

Copy code

#!/bin/bash

sudo service nginx start

**Step 3: Set Up AWS CodeDeploy**

1. **Create an Application in CodeDeploy:**
   * **Application Name:** Choose a name like MyWebApp.
   * **Compute Platform:** EC2/On-Premises.
2. **Create a Deployment Group:**
   * **Deployment Group Name:** MyWebApp-DeploymentGroup.
   * **Service Role:** Select CodeDeploy-Service-Role.
   * **Environment Configuration:** Choose your EC2 instances using tags or manually.
   * **Deployment Type:** In-Place.

**Step 4: Set Up AWS CodeBuild**

1. **Create a Build Project in CodeBuild:**
   * **Project Name:** MyWebAppBuild.
   * **Source:** Select GitHub and link to your repository.
   * **Environment:** Managed image (Ubuntu, standard image).
   * **Buildspec:** Choose "Use a buildspec file" to use the buildspec.yml.
   * **Artifacts:** Choose S3 as the storage location if needed.
   * **Service Role:** Select CodeBuild-Service-Role.

**Step 5: Set Up AWS CodePipeline**

1. **Create a Pipeline in CodePipeline:**
   * **Pipeline Name:** MyWebAppPipeline.
   * **Service Role:** Select CodePipeline-Service-Role.
   * **Artifact Store:** Choose an S3 bucket for your artifacts.
2. **Add Source Stage:**
   * **Source Provider:** GitHub.
   * **Repository:** Select your GitHub repository.
   * **Branch:** Select your branch (e.g., main).
3. **Add Build Stage:**
   * **Build Provider:** AWS CodeBuild.
   * **Project Name:** Select MyWebAppBuild.
4. **Add Deploy Stage:**
   * **Deploy Provider:** AWS CodeDeploy.
   * **Application Name:** Select MyWebApp.
   * **Deployment Group:** Select MyWebApp-DeploymentGroup.

**Step 6: Test Your CI/CD Pipeline**

1. **Push Changes to GitHub:**
   * Every time you push changes to the specified branch, the pipeline will automatically trigger.
2. **Monitor the Pipeline:**
   * Check the progress in the CodePipeline console.
   * The pipeline will pull the latest code, build it, and deploy it to your EC2 instance.
3. **Verify Deployment:**
   * Visit the public IP of your EC2 instance in a web browser. You should see the "Welcome to AWESOME AWS website" message from your index.html.