* **Complete AWS\_CI/CD Project Step by Step**
* **Prerequisites =>**
  + An AWS Account
  + Create an IAM user with below permission
    - AWSCodeCommitPowerUser
* Create a S3 bucket to store the Artifact.
* Code for deploy
* Create an ec2 for deployment
* Steps-1
* Go to the Code Commit and Create a repository
* You Can enable CodeGuru for code validation check(optional)
* Goto the IAM user which you have created 🡪 Security Credentials 🡪 HTTPS Git Credentials for Code Commit 🡪 Generate Credentials 🡪 Keep the username & password.
* Copy the HTTP URL of the repository which you have created and clone the repo in your Local (By providing above username and password)
* Create a index.html file and put your code inside the file.
* Then push the code into the remote repo.
* Steps-2 (Build Stage)
* Write a buildspec.yml file, which is required during build process.
* Code Build 🡪Create Build Project 🡪 Project Name 🡪Choose AWSCodeCommit in the Source Provider option 🡪 Put the repo Name in the repository 🡪Master Branch 🡪Environment 🡪Managed Image 🡪Select Ubuntu as an O.S 🡪”Standard” Runtimes 🡪Image🡪 aws/codebuild/standard:7.0🡪Service role (keep as it is) 🡪New Service role (If You don’t have any service role).
* Buildspec (keep as it is).
* Artifacts 🡪 Choose Amazon S3 Type 🡪 Put the Bucket Name 🡪 artifact.zip(Name) 🡪 Artifact Packaging as “Zip” 🡪 Unselect the logs (if Not required) 🡪 Create Build Project. 😊
* Then Start Build and check the status.
* Steps-3 (CodeDeploy)
* Create Application 🡪 Enter the Application Name 🡪 Choose “Ec2” as Compute Platform
* Create Deployment group 🡪Enter a Name of the deployment group 🡪 Service role 🡪 In-place as Deployment type 🡪 Environment Configuration (Amazon Ec2 instance) Choose the Ec2 by tag 🡪 Install AWS Code deploy agent (Never) 🡪Disable load balancing 🡪 Create deployment
* Install the deploy agent in the Ec2 machine.
* Create an appspec.yml file and push into the repo.
* Create two scripts for nginx .(follow the given repo link)
* Create deployment 🡪 paste the bucket path in revision location 🡪 create deployment.
* Goto ec2 🡪 Modify IAM role 🡪 provide ec2-code-deploy permission 🡪 Update IAM role .
* Login to the Ec2 🡪 restart the code deploy agent (sudo service codedeploy-agent restart)
* Check deployment done .
* Steps-4 (Code Pipeline)
* Pipeline 🡪 Create Pipeline 🡪Enter the name of the pipeline 🡪 New Service role 🡪 Next
* Source provider 🡪 AWSCodecommit 🡪 repo name 🡪 master branch 🡪 “AWSCodePipeline” in change detection 🡪 Next
* Build Provider 🡪 AWSCodeBuild 🡪 enter build project name 🡪 Next
* Deploy provider 🡪 AWSCodeDiploy 🡪 application name 🡪 deployment group 🡪 Next
* Review 🡪 Create pipeline

🎉 Exciting News: Unlock AWS CI/CD Excellence! 🌐

Super excited to share a detailed guide on crafting a powerful Continuous Integration and Continuous Deployment (CI/CD) pipeline on Amazon Web Services (AWS). Follow these four key steps:

🔧 Code Commit Repository Setup:

- Create a repository and enable CodeGuru for code validation.

- Get HTTPS Git Credentials from IAM user Security Credentials.

- Local clone, add code to index.html, and push it to the remote repo.

🛠️ Build Stage:

- Craft a buildspec.yml for the build process.

- Configure CodeBuild with the repository and branch.

- Set build settings, artifacts, and kickstart the build.

🚀 CodeDeploy:

- Establish an App and Deployment Group in CodeDeploy.

- Install the CodeDeploy agent on your EC2 machine.

- Push appspec.yml and NGINX scripts from the provided repo link.

- Start deployment, modify EC2's IAM role, restart the CodeDeploy agent.

🔄 CodePipeline:

- Create a CodePipeline: AWSCodeCommit for source, AWSCodeBuild for build, and AWSCodeDeploy for deploy.

- Review and launch the automated pipeline.

Leverage the files - index.html, buildspec.yml, deployagent.sh, and appspec.yml - for seamless application automation. Revolutionize your workflow with AWS CI/CD! 💻✨ #AWS #CICD #DevOps #CloudJourney 🚀