\*\*Prerequisites:\*\*

1. Have an AWS account.

2. Create an IAM user with the `AWSCodeCommitPowerUser` permission and configure AWS-CLI on your local/virtual machine.

\*\*Step 1: Code Commit Repository Setup\*\*

1. Go to Code Commit and create a repository.

2. Optionally enable CodeGuru for code validation.

3. Retrieve HTTPS Git Credentials for Code Commit from IAM user Security Credentials.

4. Clone the repository locally using the provided username and password.

5. Create an `index.html` file, put your code inside, and push it to the remote repo.

\*\*Step 2: Build Stage\*\*

1. Write a `buildspec.yml` file for the build process.

2. Create a CodeBuild project:

- Choose AWSCodeCommit as the Source Provider.

- Specify the repository and branch.

- Select Ubuntu as the OS.

- Use the `aws/codebuild/standard:7.0` image.

- Configure the buildspec file.

- Choose S3 as the artifact type, specify the bucket, and set artifact packaging to "Zip."

- Create the Build Project and start the build.

\*\*Step 3: CodeDeploy\*\*

1. Create an Application in CodeDeploy and enter the name.

2. Create a Deployment Group:

- Enter a name for the deployment group.

- Choose "Ec2" as the Compute Platform.

- Select In-place as Deployment type.

- Configure Environment (Amazon EC2 instance) using tags.

- Disable load balancing and create the deployment.

3. Install the CodeDeploy agent on the EC2 machine.

4. Create an `appspec.yml` file and push it to the repo.

5. Create two scripts for NGINX in the provided repository link.

6. Create a deployment, paste the S3 bucket path in revision location, and create the deployment.

7. Modify the IAM role of the EC2 instance to provide `ec2-code-deploy` permission and update the IAM role.

8. Login to the EC2 instance and restart the CodeDeploy agent (`sudo service codedeploy-agent restart`).

9. Check if the deployment is successful.

\*\*Step 4: CodePipeline\*\*

1. Create a CodePipeline:

- Enter the pipeline name.

- Choose a new service role.

- Configure source provider as AWSCodeCommit with the repository and branch.

- Configure build provider as AWSCodeBuild with the build project.

- Configure deploy provider as AWSCodeDeploy with the application name and deployment group.

- Review and create the pipeline.

\*\*Files and their Usage:\*\*

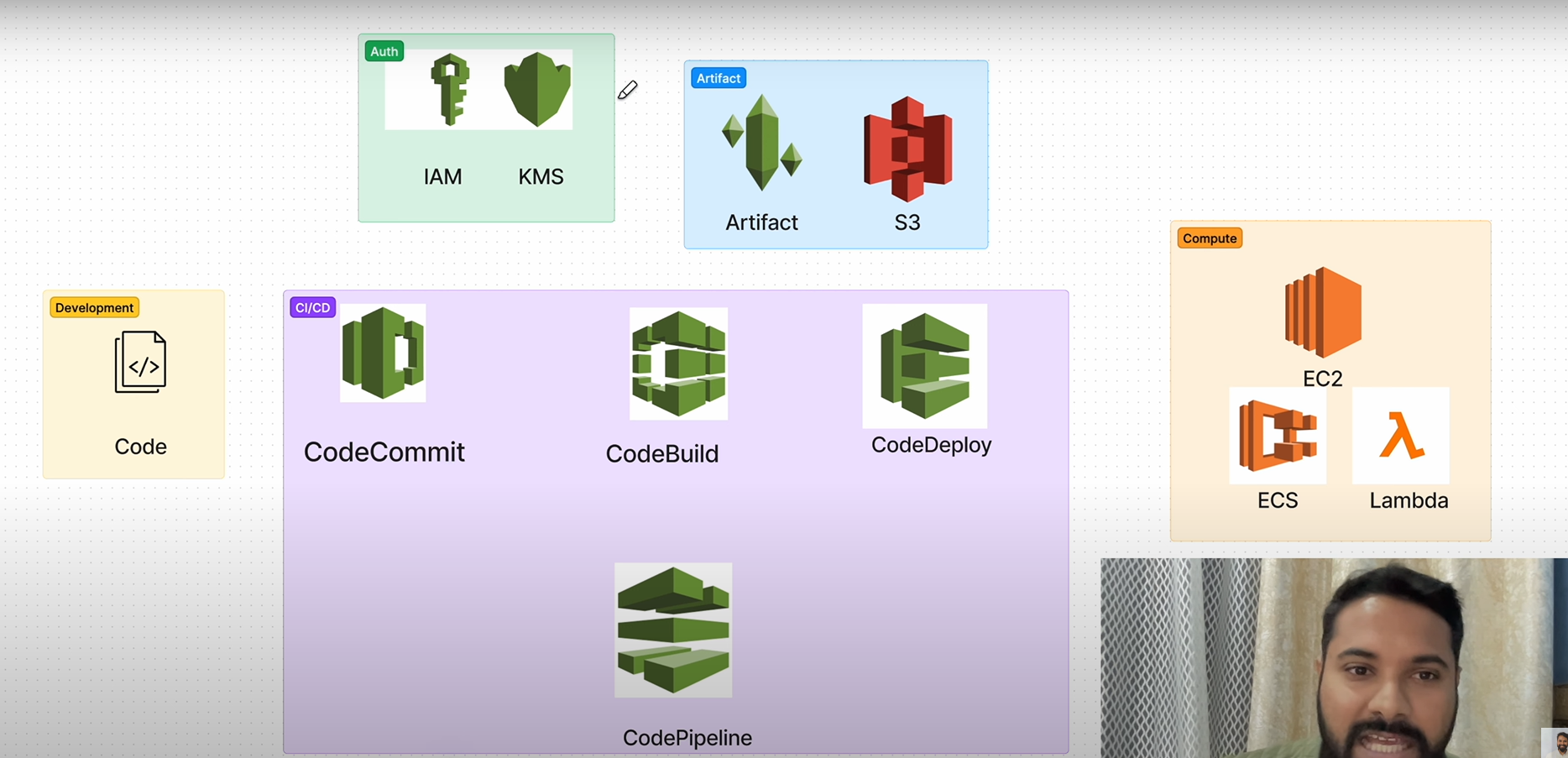
- `index.html`: This is your application code. It's a simple HTML file in this example.

- `buildspec.yml`: This file defines the build process, specifying commands to install NGINX and configure it.

- `deployagent.sh`: A script to install the CodeDeploy agent on an Ubuntu machine.

- `appspec.yml`: This file is used by AWS CodeDeploy to manage the deployment lifecycle on the EC2 instances.

By following these steps and using these files, you set up a comprehensive CI/CD pipeline on AWS, allowing you to automate the building, testing, and deployment of your applications.



**AWS CI/CD**