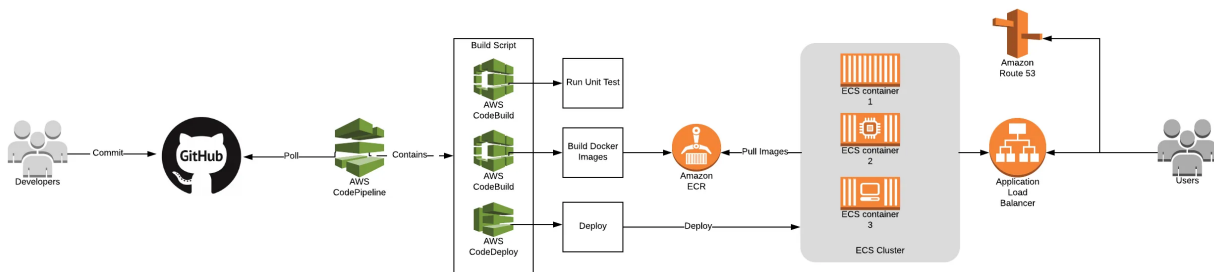


Setup CI/CD for a GitHub repository with Codebuild + Codepipeline



Pre-requisites

- You are having knowledge of the Containers concept (Docker)
- You have a GitHub repository for which you want to set up the CI/CD
- Optionally, have a Dockerfile present (preferably in the root of the repository)

Set up an ECR Repository:

- Open the Amazon ECR console at <https://console.aws.amazon.com/ecr/repositories>.
- If you are a new user, you will see a **Get Started** Button, click that; Otherwise on the **Repositories** page, choose the **Private** tab and then choose **Create repository**.
- For **Visibility settings**, verify that **Private** is selected.
- For the **Repository name**, enter a unique name for your repository.

- The repository name must start with a letter and can only contain lowercase letters, numbers, hyphens, underscores, and forward slashes
- Choose **Create repository**.

Create repository

General settings

Visibility settings [Info](#)

Choose the visibility setting for the repository.

- ☒ **Private**
Access is managed by IAM and repository policy permissions.
- ☐ **Public**
Publicly visible and accessible for image pulls.

Repository name

Provide a concise name. A developer should be able to identify the repository contents by the name.

241973884176.dkr.ecr.us-east-1.amazonaws.com/

20 out of 256 characters maximum (2 minimum). The name must start with a letter and can only contain lowercase letters, numbers, hyphens, underscores, periods and forward slashes.

Tag immutability [Info](#)

Enable tag immutability to prevent image tags from being overwritten by subsequent image pushes using the same tag. Disable tag immutability to allow image tags to be overwritten.

- ☐ Disabled

[i](#) Once a repository has been created, the visibility setting of the repository can't be changed.

Language

© 2023, Amazon Web Services



Deprecation warning

The ScanOnPush configuration at the repository level has been deprecated in favour of registry-level scan filters.

Scan on push

Enable scan on push to have each image automatically scanned after being pushed to a repository. If disabled, each image scan must be manually started to get scan results.

- ☐ Disabled

Encryption settings

KMS encryption

You can use AWS Key Management Service (KMS) to encrypt images stored in this repository instead of using the default encryption settings.

- ☐ Disabled

[i](#) The KMS encryption settings cannot be changed or disabled after the repository has been created.

Cancel

Create repository

Create an IAM Role for CodeBuild:

- Open the IAM console at <https://console.aws.amazon.com/iam/>
- Click on the "Roles" menu from the sidebar
- In the **Trusted entity type**, choose **AWS Service**, For **Use case**, choose AWS Codebuild, Click **Next**

The screenshot shows the 'Select trusted entity' step in the AWS IAM console. On the left, a sidebar lists three steps: 'Step 1: Select trusted entity' (active), 'Step 2: Add permissions', and 'Step 3: Name, review, and create'. The main content area is titled 'Select trusted entity' with an 'Info' link. Under the 'Trusted entity type' section, there are five options: 'AWS service' (selected), 'AWS account', 'Web identity', 'SAML 2.0 federation', and 'Custom trust policy'. Each option has a brief description. Below this, the 'Use case' section allows selecting a common use case ('EC2' or 'Lambda') or a specific AWS service from a dropdown menu. 'CodeBuild' is selected in the dropdown. At the bottom right, there are 'Cancel' and 'Next' buttons.

Step 1
Select trusted entity

Step 2
Add permissions

Step 3
Name, review, and create

Select trusted entity [Info](#)

Trusted entity type

- ☒ **AWS service**
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- ☐ **AWS account**
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- ☐ **Web identity**
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
- ☐ **SAML 2.0 federation**
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- ☐ **Custom trust policy**
Create a custom trust policy to enable others to perform actions in this account.

Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Common use cases

- ☐ **EC2**
Allows EC2 instances to call AWS services on your behalf.
- ☐ **Lambda**
Allows Lambda functions to call AWS services on your behalf.

Use cases for other AWS services:

CodeBuild

- ☒ **CodeBuild**
Allows CodeBuild to call AWS services on your behalf.

[Cancel](#) [Next](#)


- In the **Add permissions** step, Search for and select "AmazonEC2ContainerRegistryPowerUser", Click **Next**
- In the Role Details, set Role Name (eg: TestCodeBuildRole) and click **Create Role**

Add permissions [Info](#)

Permissions policies (Selected 1/812) [Info](#)
Choose one or more policies to attach to your new role.

1 match < 1 > [Settings](#)

X [Clear filters](#)

<input checked="" type="checkbox"/>	Policy name ↗	Type ↕	Description
<input checked="" type="checkbox"/>	 AmazonEC2ContainerRegistryPowerUser	AWS m...	Provides full access to Amazon EC2 Contai...

▶ **Set permissions boundary - optional** [Info](#)
Set a permissions boundary to control the maximum permissions this role can have. This is not a common setting, but you can use it to delegate permission management to others.

[Cancel](#)[Previous](#)[Next](#)

Create an AWS CodeBuild project:

- Open the [AWS CodeBuild console](#)
- If a CodeBuild information page is displayed, choose **Create build project**. Otherwise, on the navigation pane, expand **Build**, choose **Build projects**, and then choose **Create build project**.
- On the **Create build project** page, in **Project configuration**, for the **Project name**, enter a name for this build project
- In **Source**, for **Source provider**, choose "GitHub".

Create build project

Project configuration

Project name

test-project

A project name must be 2 to 255 characters. It can include the letters A-Z and a-z, the numbers 0-9, and the special characters - and _.

Description - *optional*

Build badge - *optional*

☐ Enable build badge

Enable concurrent build limit - *optional*

Limit the number of allowed concurrent builds for this project.

☐ Restrict number of concurrent builds this project can start

► Additional configuration

tags

Source

Add source

Source 1 - Primary

Source provider

GitHub

Repository

☐ Public repository

☒ Repository in my GitHub account

GitHub repository

Q https://github.com/kylegalbraith/aws-ecr-codepipeline-demo

×

↺

https://github.com/<user-name>/<repository-name>

Connection status

You are connected to GitHub using OAuth.

Disconnect from GitHub

Source version - *optional* [Info](#)

Enter a pull request, branch, commit ID, tag, or reference and a commit ID.

► Additional configuration

Git clone depth, Git submodules, Build status config

Primary source webhook events

Webhook - *optional*

☐ Rebuild every time a code change is pushed to this repository

⚠ [Important security considerations](#) [🔗](#)

- Connect to your GitHub account and select the repository that you want to use for CI/CD.
- In **Environment**, for **Environment image**, leave **Managed image** selected.
- For the **Operating system**, choose **Amazon Linux 2**.
- For **Runtime(s)**, choose **Standard**.
- For **Image**, choose **aws/codebuild/amazonlinux2-x86_64-standard:3.0**.
- IMPORTANT: Check **Privileged section** box (Enable this flag if you want to build Docker images or want your builds to get elevated privileges) if you are building a docker image
- In **Service role**, choose **Existing service role**, and select the role we created in previous step.
- For **Buildspec**, leave **Use a buildspec file** selected.
- In **Artifacts**, for **Type**, choose **No Artifacts**.
- Choose **Create build project**.

The programming language runtimes are now included in the standard image of Ubuntu 18.04, which is recommended for new CodeBuild projects created in the console. See [Docker Images Provided by CodeBuild for details](#).

Runtime(s)

Standard

Image

aws/codebuild/amazonlinux2-x86_64-standard:3.0

Image version

Always use the latest image for this runtime version

Environment type

Linux

Privileged

☐ Enable this flag if you want to build Docker images or want your builds to get elevated privileges

Service role

☐ New service role
Create a service role in your account

☒ Existing service role
Choose an existing service role from your account

Role ARN

☒ Allow AWS CodeBuild to modify this service role so it can be used with this build project

Additional configuration

Timeout, certificate, VPC, compute type, environment variables, file systems

Buildspec

Build specifications

☒ Use a buildspec file
Store build commands in a YAML-formatted buildspec file

☐ Insert build commands
Store build commands as build project configuration

Buildspec name - optional

By default, CodeBuild looks for a file named buildspec.yml in the source code root directory. If your buildspec file uses a different name or location, enter its path from the source root here (for example, buildspec-two.yml or configuration/buildspec.yml).

Batch configuration

You can run a group of builds as a single execution. Batch configuration is also available in advanced option when starting build.

☐ Define batch configuration - optional
You can also define or override batch configuration when starting a build batch.

Artifacts

Add artifact

Artifact 1 - Primary

Type

No artifacts

You might choose no artifacts if you are running tests or pushing a Docker image to Amazon ECR.

Additional configuration

Cache, encryption key

Logs

CloudWatch

☐ CloudWatch logs - optional
Checking this option will upload build output logs to CloudWatch.

S3

☐ S3 logs - optional
Checking this option will upload build output logs to S3.

Cancel

Create build project

Create a buildspec file:

- Create a file named "buildspec.yml" in the root of your GitHub repository.
- Paste the following code into the buildspec file, which builds the Docker image and pushes it to the ECR repository:

```
version: 0.2
phases:
  install:
    commands:
      - echo install step...
  pre_build:
    commands:
      - echo logging in to AWS ECR...
      - $(aws ecr get-login --no-include-email --region us-east-1)
  build:
    commands:
      - echo build Docker image on `date`
      - cd src
      - docker build -t <image name>:latest .
      - docker tag <image name>:latest <ECR repository URI>:latest
  post_build:
    commands:
      - echo build Docker image complete `date`
      - echo push latest Docker images to ECR...
      - docker push <ECR repository URI>:latest
```

- Replace <image name> and <ECR repository URI> with the actual values.
- Run `git add buildspec.yml` and `git push` to commit the file

Create an AWS CodePipeline pipeline:

- [Click here to open the AWS CodePipeline console.](#)
- On the Welcome page, click **Create pipeline**. If this is your first time using AWS CodePipeline, an introductory page appears instead of Welcome. Click **Get Started**.
- Enter the name for your pipeline, Choose **New service role**, and in **Role Name**, enter the name for your new service role. Click **Next**

Pipeline settings

Pipeline name
Enter the pipeline name. You cannot edit the pipeline name after it is created.

No more than 100 characters

Service role

☒ **New service role**
Create a service role in your account

☐ **Existing service role**
Choose an existing service role from your account

Role name

Type your service role name

☒ Allow AWS CodePipeline to create a service role so it can be used with this new pipeline

► **Advanced settings**

Cancel

Next

- On the **Add source stage** page, for the **Source provider**, choose **GitHub (v2)** , Click on **Connect To Github** and follow the instructions
- Set the **repository** and **branch name** and set **Output artifact format** to **CodePipeline default**

Add source stage [Info](#)

Source

Source provider

This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

GitHub (Version 2) ▼



New GitHub version 2 (app-based) action

To add a GitHub version 2 action in CodePipeline, you create a connection, which uses GitHub Apps to access your repository. Use the options below to choose an existing connection or create a new one. [Learn more](#)

Connection

Choose an existing connection that you have already configured, or create a new one and then return to this task.

🔍 arm:aws:codestar-connections:us-east-1:241973884176:connection/001b4b0 ✕ or

Connect to GitHub



Ready to connect

Your GitHub connection is ready for use.

Repository name

Choose a repository in your GitHub account.



<account>/<repository-name>

Branch name

Choose a branch of the repository.



Change detection options

☒ Start the pipeline on source code change

Automatically starts your pipeline when a change occurs in the source code. If turned off, your pipeline only runs if you start it manually or on a schedule.

Output artifact format

Choose the output artifact format.



CodePipeline default

AWS CodePipeline uses the default zip format for artifacts in the pipeline. Does not include git metadata about the repository.



Full clone

AWS CodePipeline passes metadata about the repository that allows subsequent actions to do a full git clone. Only supported for AWS CodeBuild actions.

Cancel

Previous

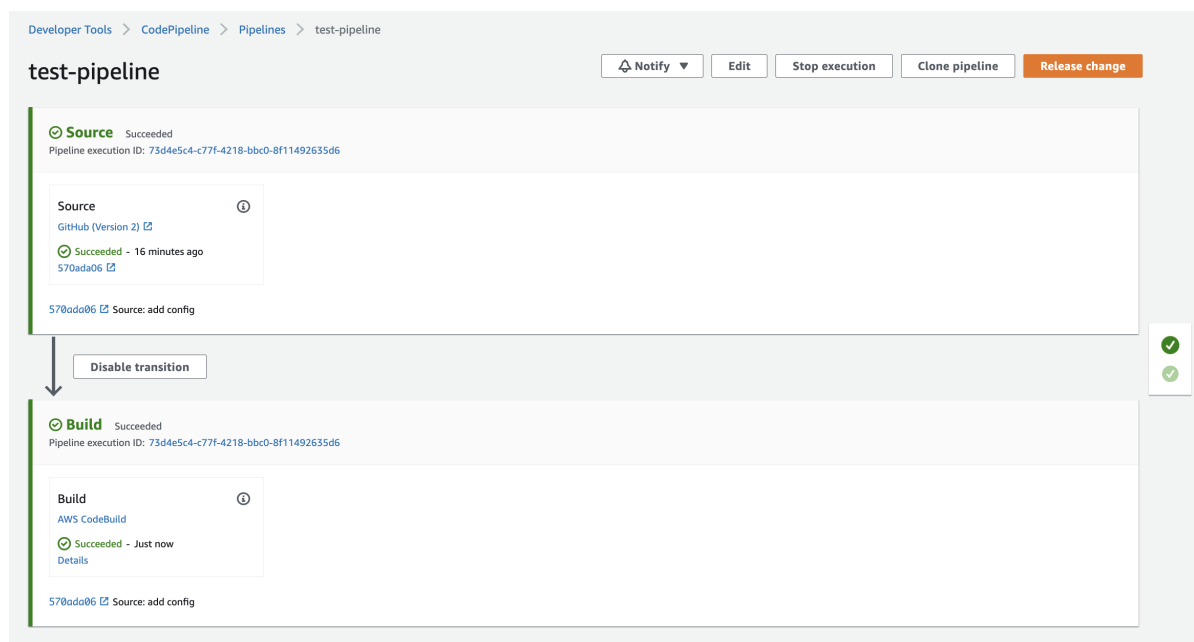
Next

- For the **build stage**, select **AWS CodeBuild** and choose the CodeBuild project created in the previous step
- For the **Add deploy stage**, choose to **skip deploy stage**. Click OK on the warning

- Go to the bottom of the review page and click **Create Pipeline**

Test the Pipeline:

- Go to your GitHub repository and make some changes to the code.
- Commit and push the changes to the repository.
- Go to the AWS CodePipeline service and check the status of your pipeline.
- You should see that the pipeline has detected the changes, started a new build, and deployed the new image to the ECR repository you added.



And that's it! You now have a fully functional CI/CD pipeline for your GitHub repository using AWS CodeBuild and CodePipeline. Every time you update your GitHub repository, the Docker image will be rebuilt and redeployed to the ECR repository