

# Continuous Integration and Continuous Deployment of a dockerized application using AWS CodeBuild, CodePipeline and GitHub

## 1. CONTINUOUS INTEGRATION

All the source code is saved in a GitHub repository. We utilized AWS CodeBuild for the purpose of continuous integration, so whenever a change is committed in the GitHub repository, a build automatically triggers in AWS CodeBuild, which builds the docker image and pushes the image to AWS ECR. The build steps are mentioned in buildspec.yml file provided below.

Store the source files in GitHub repository.

Example files:

app.js

```
1 console.log("Hello, World");
```

package.json

```
1 {
2   "name": "my-node-app",
3   "version": "1.0.0",
4   "description": "A simple Node.js 'Hello, World!' application",
5   "main": "app.js",
6   "scripts": {
7     "start": "node app.js"
8   },
9   "author": "Your Name",
10  "license": "MIT",
11  "dependencies": {
12    "express": "^4.17.1"
13  }
14 }
```

Dockerfile

```
# Use an official Node.js runtime as a parent image
FROM node:latest

# Set the working directory inside the container
WORKDIR /usr/src/app

# Copy package.json and package-lock.json to the container
COPY package*.json ./

# Install app dependencies
RUN npm install

# Copy the rest of your application source code to the container
COPY . .

# Expose a port for your Node.js application
EXPOSE 3000

# Define the command to run your Node.js application
CMD ["node", "app.js"]
```

buildspec.yml

Be sure to enter the ECR login command and repository's URI, so the image could be pushed to the intended repository

```
1  version: 0.2
2
3  phases:
4    pre_build:
5      commands:
6        - echo Logging in to Amazon ECR...
7        - aws --version
8        - aws ecr <login-command>
9        - REPOSITORY_URI=
10       COMMIT_HASH=$(echo $CODEBUILD_RESOLVED_SOURCE_VERSION | cut -c 1-7)
11       IMAGE_TAG=build-$(echo $CODEBUILD_BUILD_ID | awk -F":" '{print $2}')
12
13     build:
14       commands:
15         - echo Build started on `date`
16         - echo Building the Docker image...
17         - docker build -t $REPOSITORY_URI:latest .
18         - docker tag $REPOSITORY_URI:latest $REPOSITORY_URI:$IMAGE_TAG
19
20     post_build:
21       commands:
22         - echo Build completed on `date`
23         - echo Pushing the Docker images...
24         - docker push $REPOSITORY_URI:latest
25         - docker push $REPOSITORY_URI:$IMAGE_TAG
26         - echo Writing image definitions file...
27         - printf ' [{"name":"nodeapp","imageUri":"%s"}]' $REPOSITORY_URI:$IMAGE_TAG > imagedefinitions.json
28         - cat imagedefinitions.json
29
30     artifacts:
31       files: imagedefinitions.json
```

Now create a project on AWS CodeBuild

Set your GitHub repository as your primary source

And then select the webhook so the project instantly gets access to any change committed in GitHub code, and rebuilds automatically.

Webhook - *optional* [Info](#)

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

Build type

☒ Single build  
Triggers single build

☐ Batch build  
Triggers multiple builds as single execution

▼ Webhook event filter groups [Add filter group](#)

A build is triggered if any filter group evaluates to true, which occurs when all the filters in the group evaluate to true.

Filter group 1 [Remove filter group](#)

Event type - *optional*

Add one or more webhook event filter groups to specify which events trigger a new build. If you do not add a webhook event filter group, then a new build is triggered every time a code change is pushed to your repository.

Filters

Add one or more filters to specify whether or not a build is triggered based on the selected condition, type and pattern.

► Additional configuration

note the service role name, as you'll have to give this role access to ECR to store the built docker images.

Role name

codebuild-node-app-service-role

Type your service role name

Select the buildspec file option.

▼ Buildspec

Build specifications

☐ Insert build commands  
Store build commands as build project configuration

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

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).

buildspec.yml

After creating the project, give the service role the following permissions (you can give these permissions by navigating to IAM -> Roles -> Role name (noted above) -> Attach permissions

Other permissions policies (2/1038)

Filter by Type

Q contain X All types 14 matches < 1 > ⚙

Policy name	Type	Description
<input checked="" type="checkbox"/> <a href="#">AmazonEC2ContainerRegistryFullAccess</a>	AWS managed	Provides administrative access to Ama...
<input checked="" type="checkbox"/> <a href="#">AmazonEC2ContainerRegistryPowerUser</a>	AWS managed	Provides full access to Amazon EC2 Co...

These permissions allow the built docker images to be automatically pushed to ECR during continuous integration.

Now whenever you commit a change in GitHub repository, a build will automatically be triggered and a new image will be pushed to ECR.

## 2. CONTINUOUS DEPLOYMENT

Create an ECS cluster, create a task definition using the image pushed by CodeBuild in the previous step, and then create a service within the cluster using the task definition.

Create a CodePipeline with custom configurations.

Be sure to select the build provider and enter the required information

**Build - *optional***

**Build provider**  
Choose the tool you want to use to run build commands and specify artifacts for your build action.

☐ Commands

☒ Other build providers

AWS CodeBuild ▼

**Project name**  
Choose a build project that you have already created in the AWS CodeBuild console. Or create a build project in the AWS CodeBuild console and then return to this task.

X or [Create project](#)

Select the deploy provider and enter the required information

**Deploy - *optional***

**Deploy provider**  
Choose how you want to deploy your application or content. Choose the provider, and then provide the configuration details for that provider.

Amazon ECS ▼

Once the pipeline is created, the app will automatically be deployed after any change commits in GitHub.