

LAB: Create first CodeBuild Project

You need:

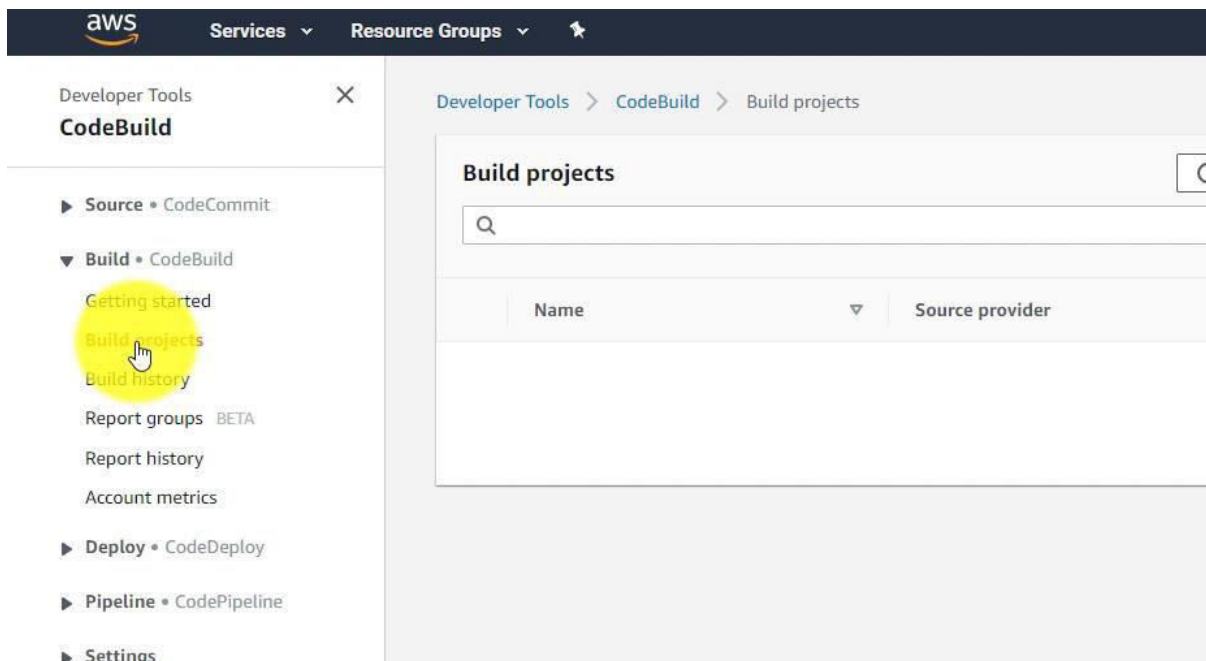
- An AWS Account
- Code in the CodeCommit Repository

Duration of the Lab: 30 Minutes.

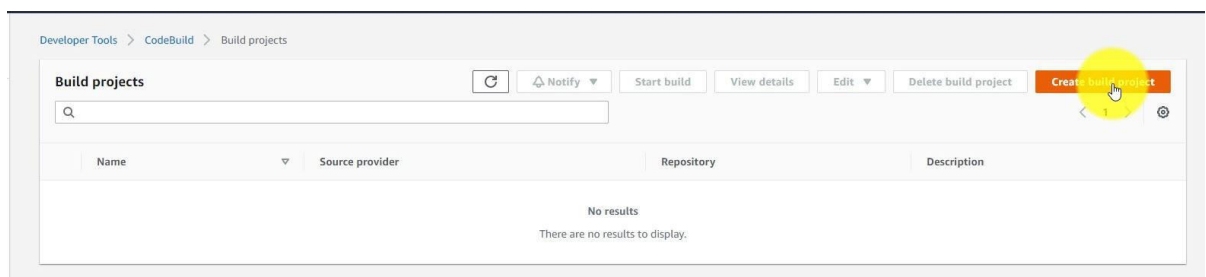
Difficulty: medium

Create a first CodeBuild Project

Open the CodeBuild section of the AWS Developer Tools Dashboard



Create a new Project:



Give the Project a name of your choice, e.g. “buildMyAwesomeProject”.

As a source select “CodeCommit” as our code is still in the CodeCommit Repository.

Then select the repository name we created earlier in the course including the master-branch:

Complete AWS ECS DevOps Masterclass for Beginners

SourceAdd source

Source 1 - Primary

Source provider:

AWS CodeCommit

Repository

Q myAwesomeProject X

Reference type
Choose the source version reference type that contains your source code.

☒ Branch

☐ Git tag

☐ Commit ID

Branch
Choose a branch that contains the code to build.

master

Commit ID - optional
Choose a commit ID. This can shorten the duration of your build.

Q

Source version [Info](#)

refs/heads/master

ff08332e we made some changes

► Additional configuration

Git clone depth, Git submodules

As Environment choose a managed Image from AWS:

Amazon Linux 2, Standard Runtime, version 3.0:

Environment

Environment image

☒ **Managed image**
Use an image managed by AWS CodeBuild

☐ **Custom image**
Specify a Docker image

Operating system

Amazon Linux 2

i 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

Because we want to build Docker-Containers on CodeBuild, we have to give privileged permissions:

Privileged

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

Scroll down to the Builds spec file. Instead of uploading our own builds spec file we will create one directly in CodeBuild:

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

Build commands

Enter commands you want to run during the build phase. Separate each build command with "&&." For example, "mvn test && mvn package." Use a buildspec file to run commands in other phases or if you have a long list of commands.

Switch to editor

Switch to the Editor and write down some simple commands:

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

Build commands

```
20      #If you use the Ubuntu standard image 2.0 or later, you must specify runtime-versions.
21      #If you specify runtime-versions and use an image other than Ubuntu standard image 2.0, the t
22      #runtime-versions:
23      #   name: version
24      #   name: version
25      #commands:
26      #   - command
27      #   - command
28      pre_build:
29      commands:
30      - echo "Prebuild phase"
31      - aws --version
32      - docker --version
33      #   - command
34      #   - command
35      build:
36      commands:
37      - echo "building now"
38      #   - command
39      #   - command
40      #post_build:
41      #commands:
42
```

Switch to single line

Be careful with indentation – yaml works off spaces.

```
pre_build:
  commands:
    - echo "Prebuild phase"
    - aws --version
    - docker --version
build:
  commands:
    - echo "building now"
```

Then scroll down and create the build project.

Starting the Build

Now it's time to start the build:

Complete AWS ECS DevOps Masterclass for Beginners

Developer Tools > CodeBuild > Build projects > buildMyAwesomeProject

buildMyAwesomeProject

[Notify](#) [Share](#) [Edit](#) [Delete build project](#)

Configuration

Source provider AWS CodeCommit	Primary repository myAwesomeProject	Artifacts upload location -	Build badge Disabled
-----------------------------------	--	--------------------------------	-------------------------

Build history | Build details | Build triggers | Metrics

Build history

[Refresh](#) [Stop build](#) [View artifacts](#) [View logs](#) [Delete builds](#) [R...](#)

	Build run	Status	Build Number	Source version	Submitter	Duration	Completed
<p>No results</p> <p>There are no results to display.</p> <p>Start build</p>							

Leave everything on default and start the build:

Complete AWS ECS DevOps Masterclass for Beginners

Timeout must be between 5 minutes and 8 hours

☐ **Disable artifacts**
Turn off artifacts configured for this project

Source

Source 1 - Primary

Source provider

AWS CodeCommit

Repository

myAwesomeProject

Reference type
Choose the source version reference type that contains your source code.

☒ Branch
☐ Git tag
☐ Commit ID

Branch
Choose a branch that contains the code to build.

master

Commit ID - *optional*
Choose a commit ID. This can shorten the duration of your build.

Source version [Info](#)

refs/heads/master

ff08332e we made some changes

► **Environment variables override**

Cancel

Start build

After a while it should simply print the container output in the log window:

Complete AWS ECS DevOps Masterclass for Beginners

Build started
You have successfully started the following build: buildMyAwesomeProject:c35003e3-b86d-48d9-9c65-fa81f5f1fc3a

Developer Tools > CodeBuild > Build projects > buildMyAwesomeProject > buildMyAwesomeProject:c35003e3-b86d-48d9-9c65-fa81f5f1fc3a

buildMyAwesomeProject:c35003e3-b86d-48d9-9c65-fa81f5f1fc3a

Build status

Status	Initiator	Build ARN	Resolved source version
Succeeded	thomasw	arn:aws:codebuild:eu-central-1:161952721022:build/buildMyAwesomeProject:c35003e3-b86d-48d9-9c65-fa81f5f1fc3a	ff08332e9aad6103b4882654
Start time	End time	Build Number	
Apr 9, 2020 1:02 PM (UTC+2:00)	Apr 9, 2020 1:03 PM (UTC+2:00)	1	

Build logs | Phase details | Reports | Environment variables | Build details

Showing the last 38 lines of the build log. [View entire log](#)

^ Show previous logs

```
1 [Container] 2020/04/09 11:03:07 Waiting for agent ping
2 [Container] 2020/04/09 11:03:09 Waiting for DOWNLOAD_SOURCE
3 [Container] 2020/04/09 11:03:11 Phase is DOWNLOAD_SOURCE
4 [Container] 2020/04/09 11:03:11 CODEBUILD_SRC_DIR=/codebuild/output/src333128148/src/git-codecommit.eu-central-1.amazonaws.com/v1/repos/myAwesomeProject
5 [Container] 2020/04/09 11:03:11 YAML location is /codebuild/readonly/buildspec.yml
6 [Container] 2020/04/09 11:03:11 Processing environment variables
7 [Container] 2020/04/09 11:03:12 No runtime version selected in buildspec.
8 [Container] 2020/04/09 11:03:12 Moving to directory /codebuild/output/src333128148/src/git-codecommit.eu-central-1.amazonaws.com/v1/repos/myAwesomeProject
9 [Container] 2020/04/09 11:03:12 Registering with agent
10 [Container] 2020/04/09 11:03:12 Phases found in YAML: 2
11 [Container] 2020/04/09 11:03:12 PRE_BUILD: 3 commands
12 [Container] 2020/04/09 11:03:12 BUILD: 1 commands
13 [Container] 2020/04/09 11:03:12 Phase complete: DOWNLOAD_SOURCE State: SUCCEEDED
14 [Container] 2020/04/09 11:03:12 Phase context status code: Message:
15 [Container] 2020/04/09 11:03:12 Entering phase INSTALL
16 [Container] 2020/04/09 11:03:12 Phase complete: INSTALL State: SUCCEEDED
```

So, the build project is running, but it doesn't do anything yet. Let's use CodeBuild to upload a new docker image when we push something to the repository.

Create and Upload containers to ECR with CodeBuild

Delete your old ECR Repository if you still have it, because we will not need this anymore. We will create a new one.

ECR > Repositories

Repositories (1 of 1)

[View push commands](#) [Delete](#)

Repository name	URI	Created at	Tag immutability
my-php-sample-repository	161952721022.dkr.ecr.eu-central-1.amazonaws.com/my-php-sample-repository	04/01/20, 02:22:35 PM	Disabled

Then create a new ECR Repository:

ECR > Repositories > Create repository

Create repository

Repository configuration

Repository name

161952721022.dkr.ecr.eu-central-1.amazonaws.com/

A namespace can be included with your repository name (e.g. namespace/repo-name).

Tag immutability

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

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

Cancel

Create repository

Integrate a buildspec.yml file

Now we need to integrate a buildspec.yml file into our codecommit repository, so that codebuild knows what to do. I took this example file from the following amazon blog post:

<https://aws.amazon.com/blogs/devops/build-a-continuous-delivery-pipeline-for-your-container-images-with-amazon-ecr-as-source/>

add the following buildspec.yml file to your codecommit repository containing the index.php and the Dockerfile:


```
version: 0.2

phases:
  pre_build:
    commands:
      - echo Logging in to Amazon ECR...
      - aws --version
      - $(aws ecr get-login --region $AWS_DEFAULT_REGION --no-include-email)
      - REPOSITORY_URI=012345678910.dkr.ecr.us-east-1.amazonaws.com/base-image
      - COMMIT_HASH=$(echo $CODEBUILD_RESOLVED_SOURCE_VERSION | cut -c 1-7)
      - IMAGE_TAG=${COMMIT_HASH:=latest}
  build:
    commands:
      - echo Build started on `date`
      - echo Building the Docker image...
      - docker build -t $REPOSITORY_URI:latest .
      - docker tag $REPOSITORY_URI:latest $REPOSITORY_URI:$IMAGE_TAG
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker images...
      - docker push $REPOSITORY_URI:latest
      - docker push $REPOSITORY_URI:$IMAGE_TAG
```

Edit the REPOSITORY_URI to the ECR URI and then stage, commit and push it:

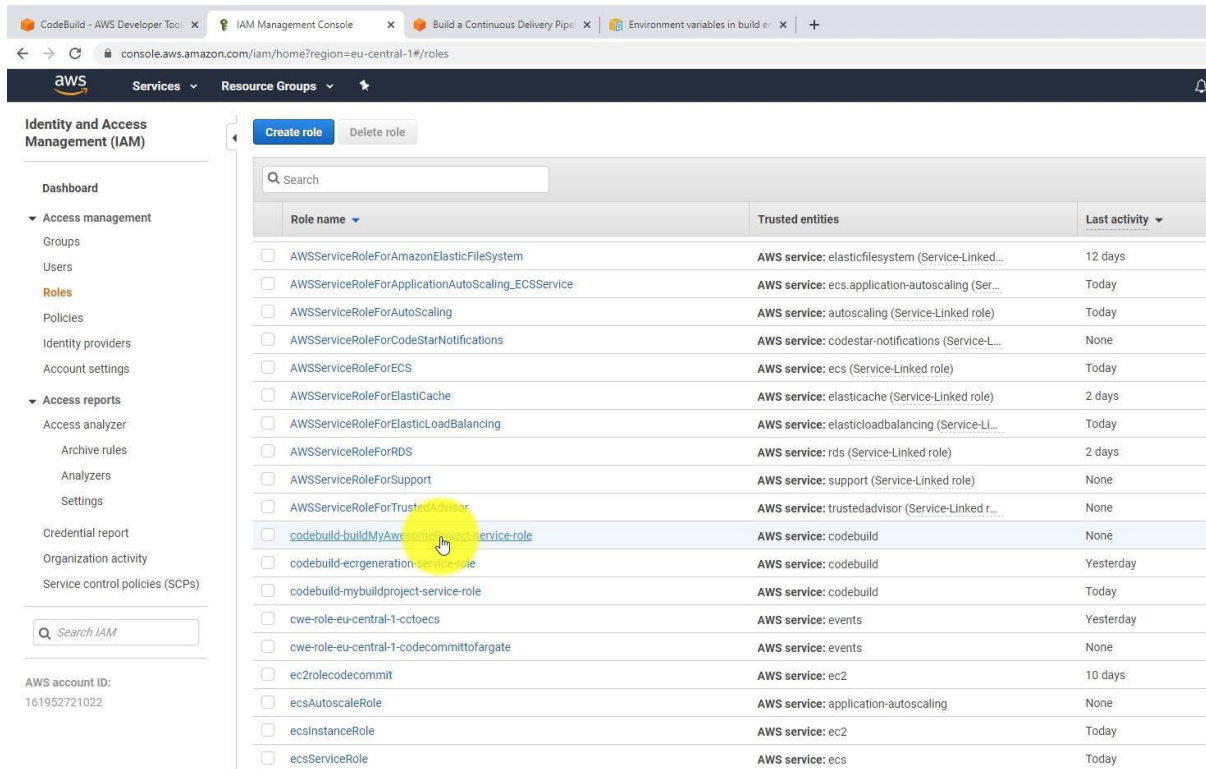
```
git add .
git commit -a -m "added buildspec.yml file"
git push origin master
```

After that edit the IAM Role for the Build project, so that it can actually push to ECR.

[Edit IAM Roles](#)

Find the Role for the codebuild project in the IAM console:

Complete AWS ECS DevOps Masterclass for Beginners




The screenshot shows the AWS IAM console with the 'Roles' tab selected. A list of roles is displayed, and the role 'codebuild-buildMyAwesomeProject-service-role' is highlighted with a yellow circle. The role is associated with the 'AWS service: codebuild' trusted entity and has no last activity.

Role name	Trusted entities	Last activity
AWSServiceRoleForAmazonElasticFileSystem	AWS service: elasticfilesystem (Service-Linked...	12 days
AWSServiceRoleForApplicationAutoScaling_EC2Service	AWS service: ecs.application-autoscaling (Ser...	Today
AWSServiceRoleForAutoScaling	AWS service: autoscaling (Service-Linked role)	Today
AWSServiceRoleForCodeStarNotifications	AWS service: codestarc-notifications (Service-L...	None
AWSServiceRoleForECS	AWS service: ecs (Service-Linked role)	Today
AWSServiceRoleForElasticCache	AWS service: elasticcache (Service-Linked role)	2 days
AWSServiceRoleForElasticLoadBalancing	AWS service: elasticloadbalancing (Service-LI...	Today
AWSServiceRoleForRDS	AWS service: rds (Service-Linked role)	2 days
AWSServiceRoleForSupport	AWS service: support (Service-Linked role)	None
AWSServiceRoleForTrustedAdvisor	AWS service: trustedadvisor (Service-Linked r...	None
codebuild-buildMyAwesomeProject-service-role	AWS service: codebuild	None
codebuild-ecrgeneration-service-role	AWS service: codebuild	Yesterday
codebuild-mybuildproject-service-role	AWS service: codebuild	Today
cwe-role-eu-central-1-cctoecs	AWS service: events	Yesterday
cwe-role-eu-central-1-codecommitofargate	AWS service: events	None
ec2rolecodecommit	AWS service: ec2	10 days
ecsAutoscaleRole	AWS service: application-autoscaling	None
ecsInstanceRole	AWS service: ec2	Today
ecsServiceRole	AWS service: ecs	Today

And attach a new policy, called “AmazonEC2ContainerRegistryPowerUser”:

Add permissions to codebuild-buildMyAwesomeProject-service-role

Attach Permissions



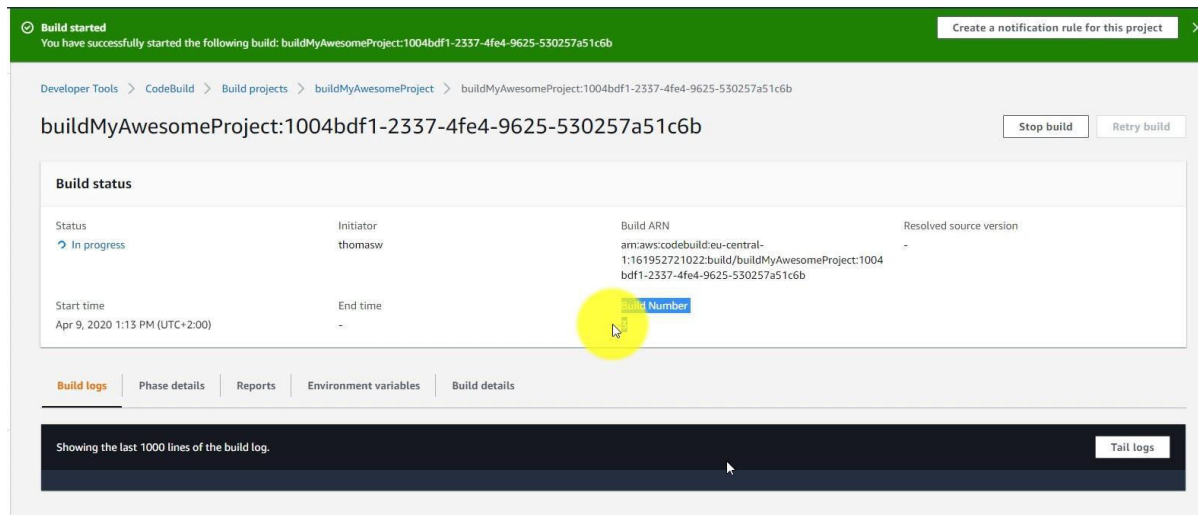
The screenshot shows the 'Attach Permissions' step in the AWS IAM console. A list of policies is displayed, and the policy 'AmazonEC2ContainerRegistryPowerUser' is selected with a yellow circle. The policy is associated with the 'AWS managed' type and is used as 'Permissions policy (1)'.

Policy name	Type	Used as
AmazonEC2ContainerRegistryFullAccess	AWS managed	None
AmazonEC2ContainerRegistryPowerUser	AWS managed	Permissions policy (1)
AmazonEC2ContainerRegistryReadOnly	AWS managed	None
AmazonRoute53AutoNamingRegistrantAccess	AWS managed	None
AWSCloudMapRegisterInstanceAccess	AWS managed	None
AWSIoTThingsRegistration	AWS managed	None
AWSMarketplaceMeteringRegisterUsage	AWS managed	None
AWSOpsWorksInstanceRegistration	AWS managed	None
AWSOpsWorksRegisterCLI_EC2	AWS managed	None
AWSOpsWorksRegisterCLI_OnPremises	AWS managed	None

Start the Build

Then simply start the build in CodeBuild:

Complete AWS ECS DevOps Masterclass for Beginners



The screenshot shows the AWS CodeBuild console for a build named `buildMyAwesomeProject:1004bdf1-2337-4fe4-9625-530257a51c6b`. The build status is **In progress**. A yellow circle highlights the **Build Number** link. The console also shows the build ARN and the resolved source version. At the bottom, there is a section for build logs, indicating the last 1000 lines are shown.

Build started
You have successfully started the following build: `buildMyAwesomeProject:1004bdf1-2337-4fe4-9625-530257a51c6b`

Developer Tools > CodeBuild > Build projects > `buildMyAwesomeProject` > `buildMyAwesomeProject:1004bdf1-2337-4fe4-9625-530257a51c6b`

buildMyAwesomeProject:1004bdf1-2337-4fe4-9625-530257a51c6b [Stop build] [Retry build]

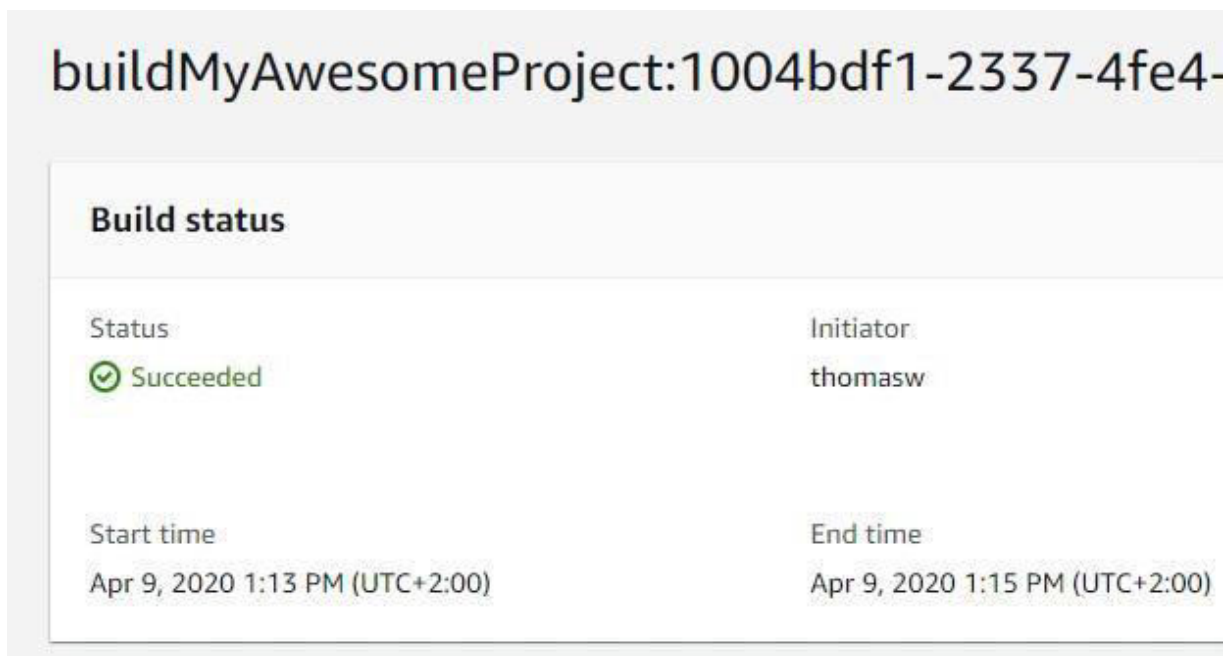
Build status

Status In progress	Initiator thomasw	Build ARN am:aws:codebuild:eu-central-1:161952721022:build/buildMyAwesomeProject:1004bdf1-2337-4fe4-9625-530257a51c6b	Resolved source version
Start time Apr 9, 2020 1:13 PM (UTC+2:00)	End time		

Build logs | Phase details | Reports | Environment variables | Build details

Showing the last 1000 lines of the build log. [Tail logs]

Wait until it says “Build Status: succeeded”:



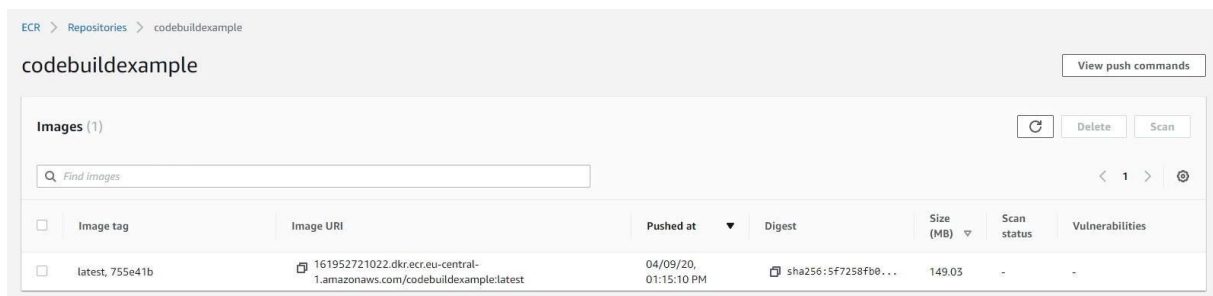
The screenshot shows the same AWS CodeBuild console, but the build status is now **Succeeded**. The build number is `buildMyAwesomeProject:1004bdf1-2337-4fe4-9625-530257a51c6b`. The console shows the build ARN and the resolved source version. The start time is `Apr 9, 2020 1:13 PM (UTC+2:00)` and the end time is `Apr 9, 2020 1:15 PM (UTC+2:00)`.

buildMyAwesomeProject:1004bdf1-2337-4fe4-9625-530257a51c6b

Build status

Status Succeeded	Initiator thomasw
Start time Apr 9, 2020 1:13 PM (UTC+2:00)	End time Apr 9, 2020 1:15 PM (UTC+2:00)

And head over to ECR to check if the image was uploaded correctly:



The screenshot shows the AWS ECR console for a repository named `codebuildexample`. It displays one image with the tag `latest, 755e41b`. The image URI is `161952721022.dkr.ecr.eu-central-1.amazonaws.com/codebuildexample:latest`. The pushed at time is `04/09/20, 01:15:10 PM` and the digest is `sha256:5f7258fb0...`. The size is `149.03` MB. The scan status is `-` and there are no vulnerabilities.

ECR > Repositories > `codebuildexample`

codebuildexample [View push commands]

Images (1)

Find images

Image tag	Image URI	Pushed at	Digest	Size (MB)	Scan status	Vulnerabilities
latest, 755e41b	161952721022.dkr.ecr.eu-central-1.amazonaws.com/codebuildexample:latest	04/09/20, 01:15:10 PM	sha256:5f7258fb0...	149.03	-	-

Lab End