



Adding a test action with Code Build

In this exercise, we intentionally introduced a mistake in our code to see how it affects our pipeline and then added automated tests to prevent such issues. Initially, we changed a calculator operator to cause a deliberate error and verified it by pushing the code, which updated the website incorrectly. This highlighted the need for automated testing.

We then created a unit-test-buildspec.yml file to run unit tests using CodeBuild before the build stage. By editing the pipeline to include this new test action, we ensured that tests run first and the build proceeds only if tests pass. After reintroducing the mistake and pushing the code, the pipeline failed at the test stage, preventing the faulty build from deploying. We then corrected the error, re-ran the pipeline, and confirmed everything worked correctly.

End Goal

The aim was to demonstrate the importance of automated testing in a CI/CD pipeline to catch and fix errors before deployment, ensuring reliable and error-free software delivery.

😊 To begin with the Lab:

1. In this lab first we are going to change the version name of our website. For that expand the app folder and you will see a file named app.component.html. from there you need to change the version of your website.

```
FOLDERS: MY-ANGULAR-PROJECT
src > app > app.component.html > app.component.html •
src > app > app.component.html > header.container > div.row.pt-4.pb-4 > div.col > h2.text-end > span.badge.bg-info
1  <header class="container">
2    <div class="row pt-4 pb-4">
3      <div class="col">
4        <h1 id="page-title" class="text-dark text-center">
5          Sample Angular App for <span class="text-danger">AWS CodePipeline Step by Step</span>
6        </h1>
7        <hr>
8        <h2 class="text-end">
9          | <span class="badge bg-info">Version: 3.0</span>
10       </h2>
11     </div>
12   </div>
13 </header>
```

2. Now before doing anything else let's go to our website and check how it is working. So, below you can see that our website is currently working fine, there are no issues with it.

Sample Angular App for AWS CodePipeline Step by Step

Version: 2.0

Congratulations! You successfully built and deployed your code.

This is a simple single-page calculator app developed with Angular and Bootstrap for the build examples on the AWS CodePipeline Step by Step course.



3. Now come back to VS Code and expand the calculator folder now let's say a mistake has been made by us that is we forgot to put the right operator in the highlighted part. You see here instead of the divide symbol, it should be plus but intentionally for this lab I'd changed it to division, you should also do this. Then save everything.
4. Now let's push our code to the repository. Now, if you push this change the plus operator will act like a division, right? Well, is there anything on our pipeline to avoid this?

A screenshot of the Visual Studio Code editor. The left sidebar shows a project structure with 'FOLDERS: MY-ANGULAR-PROJECT' containing 'e2e', 'src' (which has 'app' and 'calculator' subfolders), and various files like 'buildspec.yml', 'app.component.html', etc. The main editor area is focused on 'calculator.component.ts'. A specific line of code, 'this.result = firstInput / secondInput; break;', is highlighted with a red box. The code block is as follows:

```
9  export class CalculatorComponent implements OnInit {
10
11    constructor(private fb: FormBuilder) { }
12
13    ngOnInit() {
14        this.form = this.fb.group({
15            firstInput: ['', [Validators.required]],
16            operator: ['', [Validators.required]],
17            secondInput: ['', [Validators.required]]
18        });
19    }
20
21    onSubmit() {
22        // Calculate according to the operator
23        switch (operator.value) {
24            case '+': {
25                this.result = firstInput + secondInput;
26                break;
27            }
28            case '-': {
29                this.result = firstInput - secondInput;
30                break;
31            }
32            case '*': {
33                this.result = firstInput * secondInput;
34                break;
35            }
36            case '/': {
37                this.result = firstInput / secondInput;
38                break;
39            }
40        }
41    }
42}
```

5. Below you can see that we committed to our changes and pushed it all to the repository. And as expected our pipeline has been started.

A screenshot of a terminal window titled 'TERMINAL'. It shows two command-line sessions. The first session runs 'git commit -a -m "Version 3.0"' followed by a warning about line endings. The second session runs 'git push origin master'. The output shows the commit message, the number of files changed, and the progress of the push operation to an AWS repository.

```
PS C:\Users\PULKIT\Downloads\my-angular-project> git commit -a -m "Version 3.0"
warning: in the working copy of 'src/app/app.component.html', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/app/calculator/calculator.component.ts', LF will be replaced by CRLF the next time Git touches it
[master 8d1df33] Version 3.0
2 files changed, 2 insertions(+), 2 deletions(-)
PS C:\Users\PULKIT\Downloads\my-angular-project> git push origin master
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 12 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 567 bytes | 567.00 KiB/s, done.
Total 7 (delta 6), reused 0 (delta 0), pack-reused 0
remote: Validating objects: 100%
To https://git-codecommit.eu-west-1.amazonaws.com/v1/repos/DemoAngularRepo
  292d760..8d1df33  master -> master
PS C:\Users\PULKIT\Downloads\my-angular-project>
```

- Once your pipeline has been executed successfully then you need to go to the website refresh it check the version and try to do the addition on the calculator.
- Below you can see that our version has been changed which means that the pipeline was executed properly, also when we try to do addition it performed division.

Sample Angular App for AWS CodePipeline Step by Step

Version: 3.0

Congratulations! You successfully built and deployed your code.

This is a simple single-page calculator app developed with Angular and Bootstrap for the build examples on the AWS CodePipeline Step by Step course.

Simple Calculator

The screenshot shows a simple calculator interface with three input fields and two buttons. The first field contains '40', the second contains '+', and the third contains '8'. Below these is a red 'Clear' button and a green 'Calculate' button. At the bottom, a light blue bar displays the result 'Result: 5'.

- We just broke the application. The bad thing is if you have many users, they notice these types of mistakes before you do. Then it will have a bad impact on you.
- But first we will fix our application so go back to VS Code and change the operator to its original state which is addition. Then push the code to the repository, wait for the pipeline to run, and check if our calculator is working properly or not.
- Also, you can change the version of your website. Below you can see that we have changed everything back to normal.

Sample Angular App for AWS CodePipeline Step by Step

Version: 3.1

Congratulations! You successfully built and deployed your code.

This is a simple single-page calculator app developed with Angular and Bootstrap for the build examples on the AWS CodePipeline Step by Step course.

Simple Calculator

The screenshot shows the same calculator interface as before, but now the result is correct. The first field contains '12', the second contains '+', and the third contains '13'. The result bar at the bottom now displays 'Result: 25'.

- Hence, we need the tools to avoid these types of failures before the deployment. So, how can we achieve that? Well, we use automated tests for this purpose. We can have unit tests in our code and run them before the build command. So, we can catch them at the building stage. In addition, we can also have a staging environment and run our integration tests there before deploying to production.



Adding a test action to the code build

- Now open VS Code and create a new YAML file named unit-test-buildspec.yml, we are creating this buildspec file so that it can perform the checking. You can use the code below and paste it into the file, just remember to check the indentation and typo errors.

```

version: 0.2
phases:
  install:
    runtime-versions:
      nodejs: 20
    commands:
      - npm install -g @angular/cli@17

      # Chrome installation for Angular's unit tests
      - wget -q -O - https://dl.google.com/linux/linux_signing_key.pub | gpg --dearmor | tee /etc/apt/keyrings/chrome.gpg
      - echo "deb [signed-by=/etc/apt/keyrings/chrome.gpg] http://dl.google.com/linux/chrome/deb/ stable main" >> /etc/apt/sources.list.d/google-chrome.list
      - apt-get -y update
      - apt-get -y install google-chrome-stable
  pre_build:
    commands:
      - npm install
  build:
    commands:
      - ng test --no-watch --no-progress --browsers=ChromeHeadlessCI

```

```

File Edit Selection View Go Run ... ← → my-angular-project
FOLDERS: MY-ANGULAR-PROJECT
src
  app
    app.component.spec.ts
    app.component.ts
    app.module.ts
    assets
    environments
    favicon.ico
    index.html
    main.ts
    polyfills.ts
    styles.scss
    test.ts
unit-test-buildspec.yml
  .browserslistrc
  .editorconfig
  .gitignore
  angular.json
  buildspec.yml
  karma.conf.js
  package-lock.json
  package.json
  README.md
  tsconfig.app.json
  tsconfig.json
  tsconfig.spec.json
  tslint.json

```

```

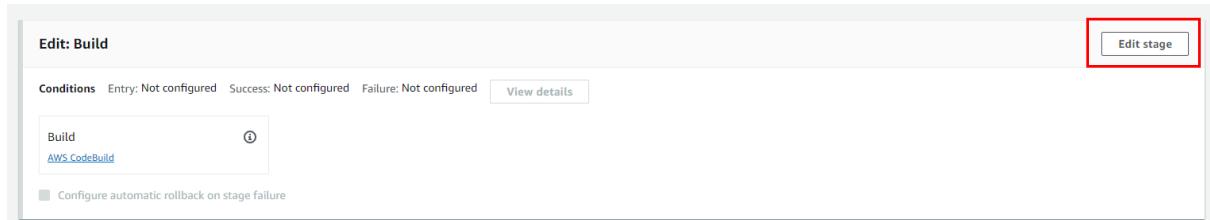
1  version: 0.2
2  phases:
3    install:
4      runtime-versions:
5        nodejs: 20
6      commands:
7        - npm install -g @angular/cli@17
8
9      # Chrome installation for Angular's unit tests
10     - wget -q -O - https://dl.google.com/linux/linux_signing_key.pub | gpg --dearmor
11     - echo "deb [signed-by=/etc/apt/keyrings/chrome.gpg] http://dl.google.com/linux/
12       chrome/deb/ stable main" >> /etc/apt/sources.list.d/google-chrome.list
13     - apt-get -y update
14     - apt-get -y install google-chrome-stable
15
16  pre_build:
17    commands:
18      - npm install
19
20  build:
21    commands:
22      - ng test --no-watch --no-progress --browsers=ChromeHeadlessCI

```

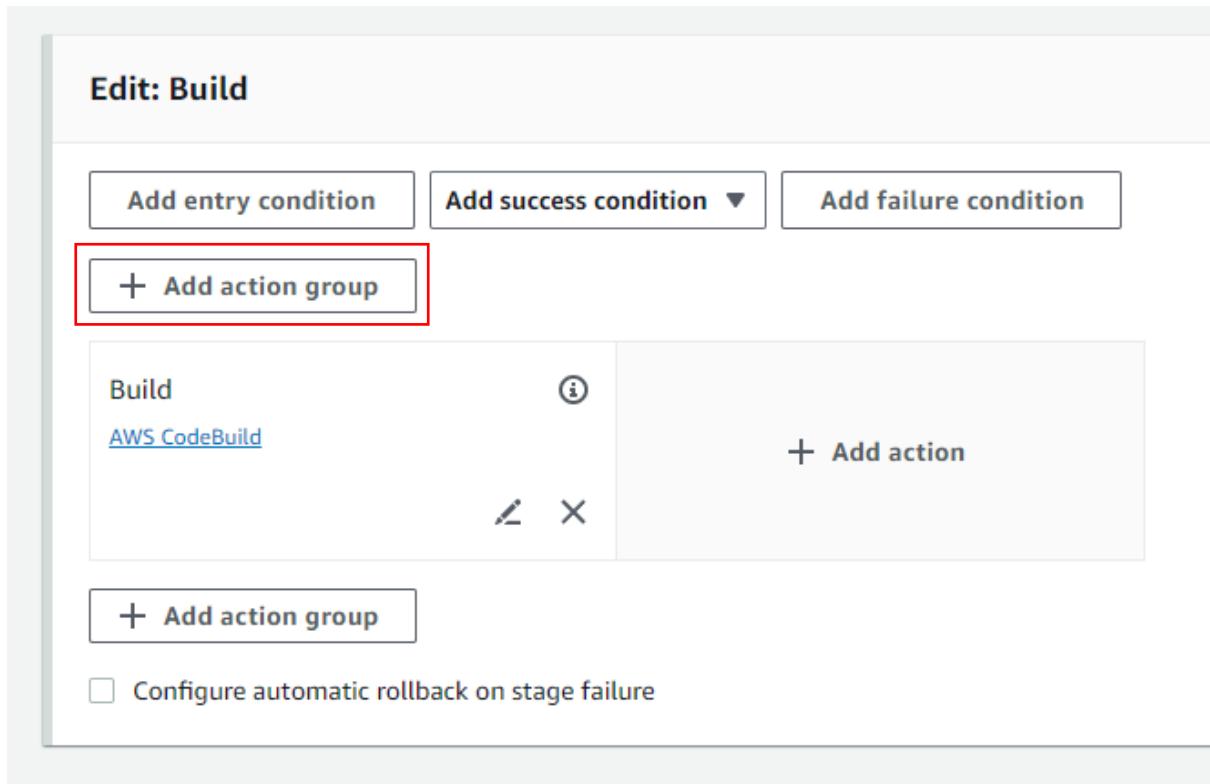
- Now go to your pipeline and we will create a sequential action along with our stage for that we need to click on edit.



3. Now we will add a new action to our build stage. So, click on the edit stage button.



4. Now click on the add action group button.



5. Now you need to give it a name and expand the bar for action provider

Edit action

Action name

Choose a name for your action

No more than 100 characters

Action provider

Variable namespace - *optional*

Choose a namespace for the output variables from this action. You must

6. Here you need to choose AWS Code Build from the Test group.

Action provider

Q

Source

- AWS CodeCommit
- Amazon ECR
- Amazon S3
- Bitbucket
- GitHub (Version 1)
- GitHub (Version 2)
- GitHub Enterprise Server
- GitLab
- GitLab self-managed

Test

- AWS CodeBuild**
- AWS Device Farm
- Add Jenkins
- Ghost Inspector UI Testing
- Micro Focus StormRunner Load
- Runscope API Monitoring

7. Here we need to choose the same region we are working on and then we need to create a new project because our new action will use the new buildspec file we just created. So, click on create project.

Action provider

AWS CodeBuild

Region

Europe (Ireland)

Input artifacts

Choose an input artifact for this action. [Learn more](#)

SourceArtifact

Add

No more than 100 characters

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.

or [Create project](#)

Environment variables - optional

Choose the key, value, and type for your CodeBuild environment variables. In the value field, you can reference variables generated by CodePipeline. [Learn more](#)

8. First, give it a name and keep the rest of the settings to default and scroll down.

Create build project

Project configuration

Project name

AngularUnitTest

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 _.

► Additional configuration

Description, Build badge, Concurrent build limit, tags

Environment

Provisioning model [Info](#)

- On-demand
Automatically provision build infrastructure in response to new builds.
- Reserved capacity
Use a dedicated fleet of instances for builds. A fleet's compute and environment type will be used for the project.

Environment image

- Managed image
Use an image managed by AWS CodeBuild
- Custom image
Specify a Docker image

Compute

- EC2
Optimized for flexibility during action runs
- Lambda
Optimized for speed and minimizes the start up time of workflow actions

9. Then you need to choose the OS as Ubuntu and then in the service role choose Create New Role, scroll down.

Operating system

Ubuntu ▾

Runtime(s)

Standard ▾

Image

aws/codebuild/standard:7.0 ▾

Image version

Always use the latest image for this runtime version ▾

Use GPU-enhanced compute

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

TestServiceRole

Type your service role name

► **Additional configuration**
[Timeout](#), [privileged](#), [certificate](#), [VPC](#), [compute type](#), [environment variables](#), [file systems](#)

10. Now choose to use a buildspec file and then specify the name of your file. If you have created your buildspec file inside of a subfolder then you need to provide its relative path here, but we have created that file in the root of the project folder.

11. After that click on continue to code pipeline.

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).

unit-test-buildspec.yml

12. We can see our project name here and this time we don't need to provide any output artifact because we don't need to produce any output. So, click on done.

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.

Environment variables - optional
Choose the key, value, and type for your CodeBuild environment variables. In the value field, you can reference variables generated by CodePipeline. [Learn more](#)

Build type
 Single build
Triggers a single build.
 Batch build
Triggers multiple builds as a single execution.

Variable namespace - optional
Choose a namespace for the output variables from this action. You must choose a namespace if you want to use the variables this action produces in your configuration. [Learn more](#)

Output artifacts
Choose a name for the output of this action.

No more than 100 characters

13. Below you can see that we have the unit test stage above the build stage. Now here also click on done and close the editing mode of the build stage.
14. Then you need to scroll up to the top of the page and save your changes.

Edit: Build

Add entry condition Add success condition ▾ Add failure condition

+ Add action group

UnitTest i

AWS CodeBuild

✎ X

+ Add action

↓ + Add action group

Build i

AWS CodeBuild

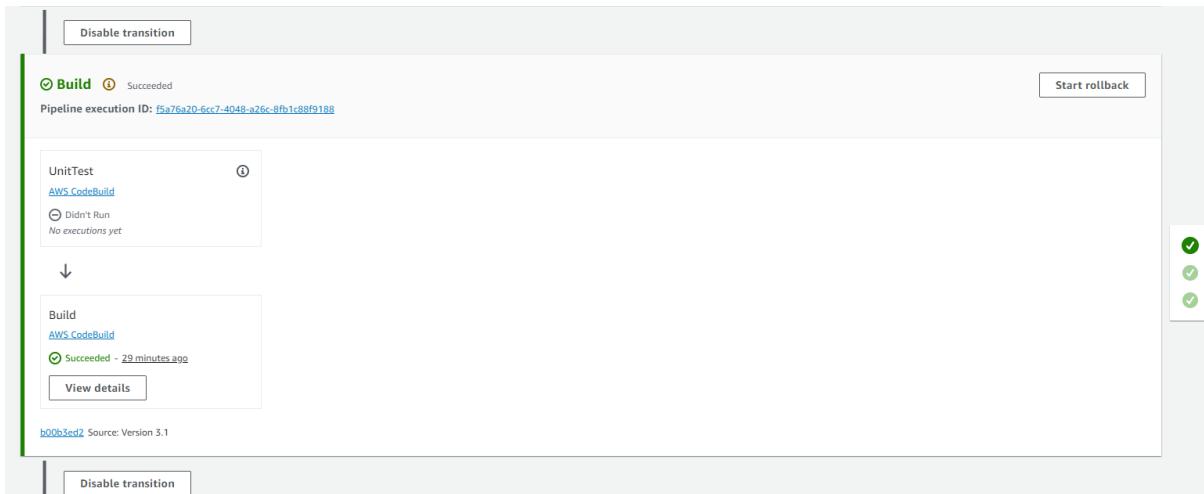
✎ X

+ Add action

+ Add action group

Configure automatic rollback on stage failure

15. As you see, our new test action was added to the Build stage before the build action. Its current state is 'Didn't Run', but it will run in the next pipeline execution. So, let's go back to our code and try to break it again.



16. Like we did at the start, change the version of your website to 4.0 and then change the operator from addition to division. So, we have simulated the same mistake.
17. Now we will push our code to the repository and soon after that our pipeline will start executing, so this time we will see the preventive method which we have set up to check whether the mistake is getting caught or not.
18. Yes, the pipeline execution is in progress. As you see, the source stage was finished, and its source version is 4.0. The build stage is currently in progress. Its source version is 4.0, too. But this time, our new Unit Test action is running first in the build stage. The build action has not been triggered yet because it will wait for the test action to succeed beforehand. This will take time because CodeBuild will provision a new container and make the necessary installations we configured as before.

Disable transition

Build i In progress

Pipeline execution ID: [dd142ce4-597c-45e6-8ddf-2b66e5b71b4c](#)

UnitTest

[AWS CodeBuild](#)

i In progress - Just now

[View details](#)

↓

Build

i

[AWS CodeBuild](#)

- Didn't Run

No executions yet

[ec903251](#) Source: Version 4.0

19. Below you can see that the pipeline execution has failed on the build stage. As you see, the UnitTest action failed as we expected, and the build action did not execute as a result.

The screenshot shows the AWS Lambda function configuration interface. At the top, there is a button labeled "Disable transition". Below it, a section titled "Build" is shown with a red "X" icon and the status "Failed". A pipeline execution ID is listed as [dd142ce4-597c-45e6-8ddf-2b66e5b71b4c](#). The "Build" step is expanded, revealing a sub-step named "UnitTest" which was run by "AWS CodeBuild". This sub-step is also marked as "Failed" with a red "X" icon and a timestamp of "1 minute ago". A "View details" button is present. Below this, another "Build" step is shown, which is associated with "AWS CodeBuild" and is currently "Didn't Run" with the message "No executions yet". A blue link "ec903251" with the note "Source: Version 4.0" is located at the bottom of the expanded "Build" section.

Disable transition

✖ Build ⓘ Failed

Pipeline execution ID: [dd142ce4-597c-45e6-8ddf-2b66e5b71b4c](#)

UnitTest

[AWS CodeBuild](#)

✖ Failed - 1 minute ago

[View details](#)

↓

Build ⓘ

[AWS CodeBuild](#)

⊖ Didn't Run
No executions yet

[ec903251](#) Source: Version 4.0

20. Now click on view details then click on view in code build.

Action execution details

Action name: UnitTest Status: Failed

Summary | **Logs** | **Input**

Status	Last updated
✖ Failed	3 minutes ago
Action execution ID	
d4e07496-d1b3-49dd-93de-2d592b489b83	
Error code	
Action execution failed	
Error message	
Build terminated with state: FAILED	
Failed phase	
DOWNLOAD_SOURCE	
Failed phase code	
YAML_FILE_ERROR	
Failed phase message	
stat /codebuild/output/src2437898196/src/unit-test-buildspec.yml: no such file or directory	
Execution details	
View in CodeBuild	

Done

21. Now click on the tail logs button.

Developer Tools > CodeBuild > Build projects > AngularUnitTest > AngularUnitTest:c048136b-a5ab-4906-80a9-d6c76e61f76e

AngularUnitTest:c048136b-a5ab-4906-80a9-d6c76e61f76e

Stop build Retry build

Build status		
Status ✖ Failed	Initiator codepipeline/AngularPipeline	Build ARN arn:aws:codebuild:eu-west-1:463646775279:build/AngularUnitTest:c048136b-a5ab-4906-80a9-d6c76e61f76e
Resolved source version ec9032510b9ad01e029de52dcfb631c435754cf4		
Start time Aug 7, 2024 8:36 PM (UTC+5:45)	End time Aug 7, 2024 8:36 PM (UTC+5:45)	Build number 1

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

Showing the last 7 lines of the build log. [View entire log](#)

No previous logs

Tail logs

```

1 [Container] 2024/08/07 14:51:46.267183 Running on CodeBuild On-demand
2 [Container] 2024/08/07 14:51:46.267183 Waiting for download...
3 [Container] 2024/08/07 14:51:46.368205 Waiting for DOWNLOAD_SOURCE
4 [Container] 2024/08/07 14:51:47.634645 Phase is DOWNLOAD_SOURCE
5 [Container] 2024/08/07 14:51:47.635670 CODEBUILD_SRC_DIR=/codebuild/output/src2437898196/src
6 [Container] 2024/08/07 14:51:47.639368 Phase complete: DOWNLOAD_SOURCE State: FAILED
7 [Container] 2024/08/07 14:51:47.639384 Phase context status code: YAML_FILE_ERROR Message: stat /codebuild/output/src2437898196/src/unit-test-buildspec.yml: no such file or directory
8

```

22. Here you can see that proper error that caused the failure.

Build logs

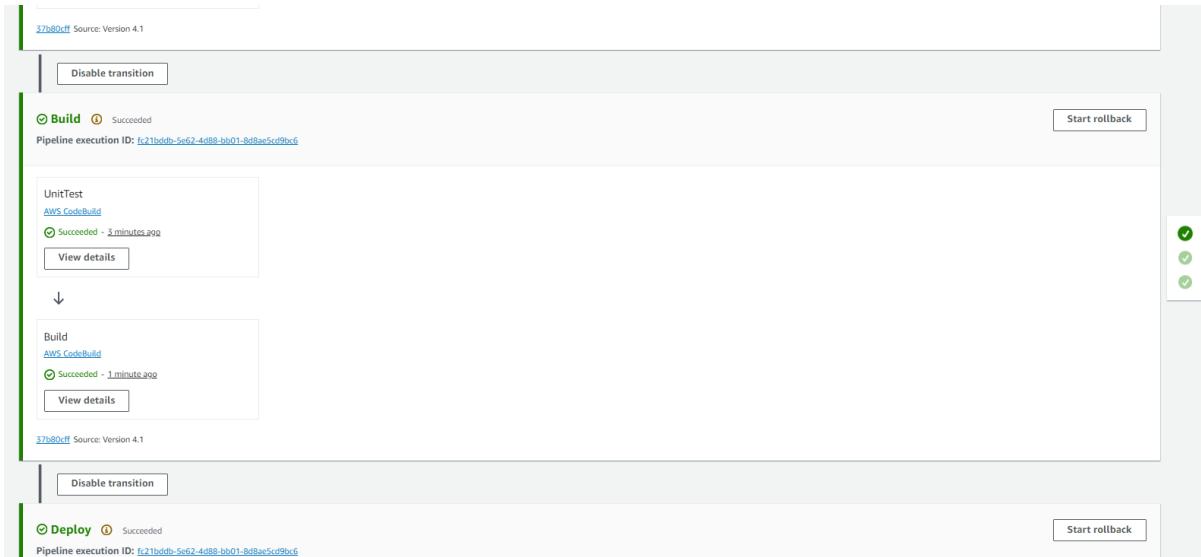
⌚ Failed Start time: 5 minutes ago Current phase: COMPLETED

```

888     at _ZoneDelegate.invoke (node_modules/zone.js/fesm2015/zone.js:368:26)
889     at ProxyZoneSpec.onInvoke (node_modules/zone.js/fesm2015/zone-testing.js:273:39)
890     at ZoneDelegate.invoke (node_modules/zone.js/fesm2015/zone.js:368:26)
891     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
892     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
893     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
894     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
895     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
896     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
897     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
898     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
899     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
900     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
901     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
902     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
903     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
904     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
905     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
906     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
907     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
908     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
909     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
910     at ZoneAwareFunctionWrapper.invoke (node_modules/zone.js/fesm2015/zone.js:454:26)
911 TOTAL: 1 FAILED, 8 SUCCESS
TOTAL: 1 FAILED, 8 SUCCESS
912
[Container] 2024/08/07 15:01:52.082375 Command did not exit successfully ng test --no-watch --no-progress --browsers=ChromeHeadlessCI exit status 1
[Container] 2024/08/07 15:01:52.087459 Phase completed: BUILD State: FAILED
[Container] 2024/08/07 15:01:52.087460 Phase context status code: COMMAND_EXECUTION_ERROR Message: Error while executing command: ng test --no-watch --no-progress --browsers=ChromeHeadlessCI. Reason: exit status 1
[Container] 2024/08/07 15:01:52.120931 Entering phase POST_BUILD
[Container] 2024/08/07 15:01:52.122878 Phase complete: POST_BUILD State: SUCCEEDED
[Container] 2024/08/07 15:01:52.122888 Phase context status code: Message:
[Container] 2024/08/07 15:01:52.250177 Set report auto-discover timeout to 5 seconds
[Container] 2024/08/07 15:01:52.250815 Expanding base directory path: .
[Container] 2024/08/07 15:01:52.258319 Assembling file list
[Container] 2024/08/07 15:01:52.258331 Expanding .
[Container] 2024/08/07 15:01:52.260499 Expanding file paths for base directory .
[Container] 2024/08/07 15:01:52.260513 Assembling file list
[Container] 2024/08/07 15:01:52.260517 Expanding */
[Container] 2024/08/07 15:01:52.260545 Found 18 file(s)
[Container] 2024/08/07 15:01:52.530039 Report auto-discover file discovery took 0.279816 seconds
[Container] 2024/08/07 15:01:52.530737 Phase complete: UPLOAD_ARTIFACTS State: SUCCEEDED
[Container] 2024/08/07 15:01:52.530761 Phase context status code: Message:
911

```

23. Also, if you go to the website and try to calculate something then you will see that it is working fine because this time there was no pipeline execution. The pipeline stopped at the Unit Test stage.
24. Now go back to VS Code, change the operation back to its original state, and push the code once again.
25. Below you can see that our build was successfully executed.



26. Now if you go to your website and refresh it then you can see the new version and your addition operator is working properly.

Sample Angular App for AWS CodePipeline Step by Step

Version: 4.0

Congratulations! You successfully built and deployed your code.

This is a simple single-page calculator app developed with Angular and Bootstrap for the build examples on the AWS CodePipeline Step by Step course.

Simple Calculator

+
Clear Calculate

Result: 40

27. Now do not delete your pipeline or the repository because we will be using our current pipeline in the next set of labs.