

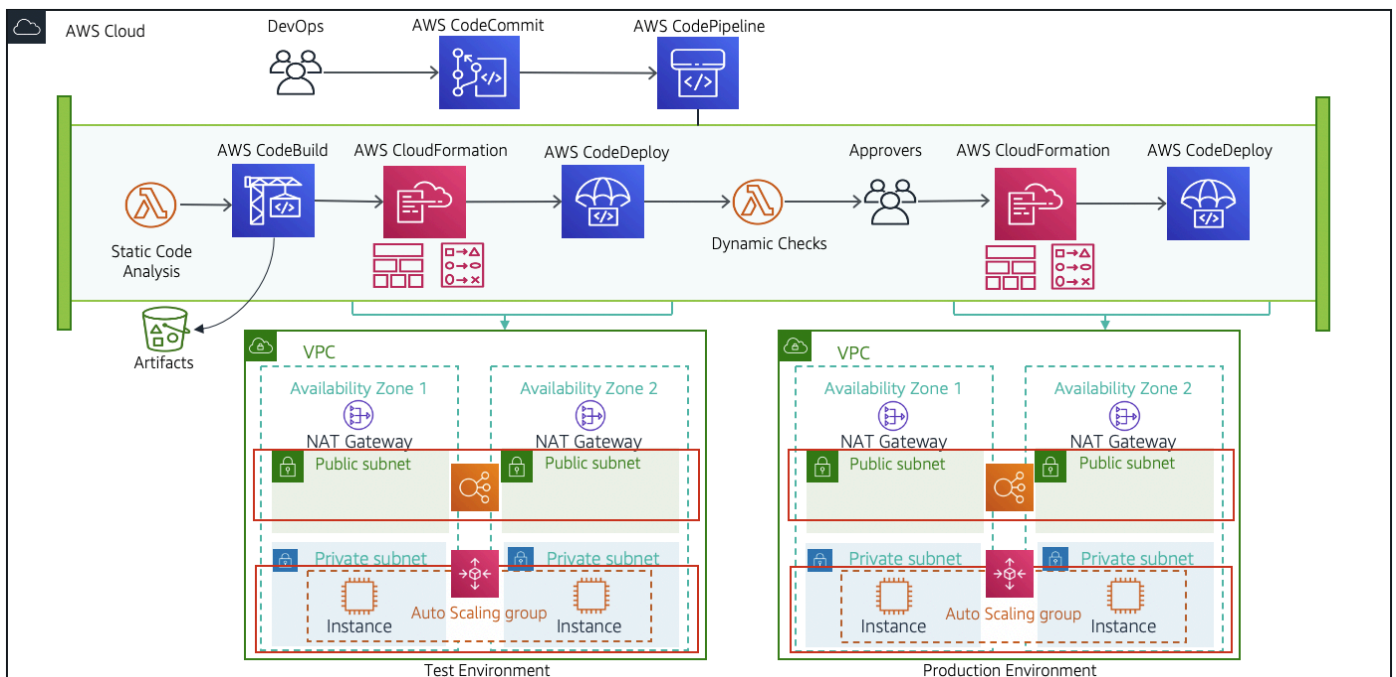
Lab 6: Using AWS DevOps tools for CI/CD pipeline automations

We work on a release pipeline that has failed in the build stage

Objectives of this lab:

1. Understand the architecture of a release pipeline.
2. Perform basic troubleshooting of failed stages in AWS CodePipeline by locating, analyzing logs, and applying fixes as needed.
3. Adjust the configurations of an AWS infrastructure based on test results from the pipeline.
4. Validate and manually approve a change between stage transitions of a pipeline.
5. Add new actions to an existing AWS CodePipeline stage.

NOTE - X-Ray provides an end-to-end view of requests as they travel through your application, making it easier to identify performance bottlenecks and errors.



An **AWS CodeCommit** repository has already been created, and the code for the demo calculator web application has already been uploaded into the AWS CodeCommit repository.

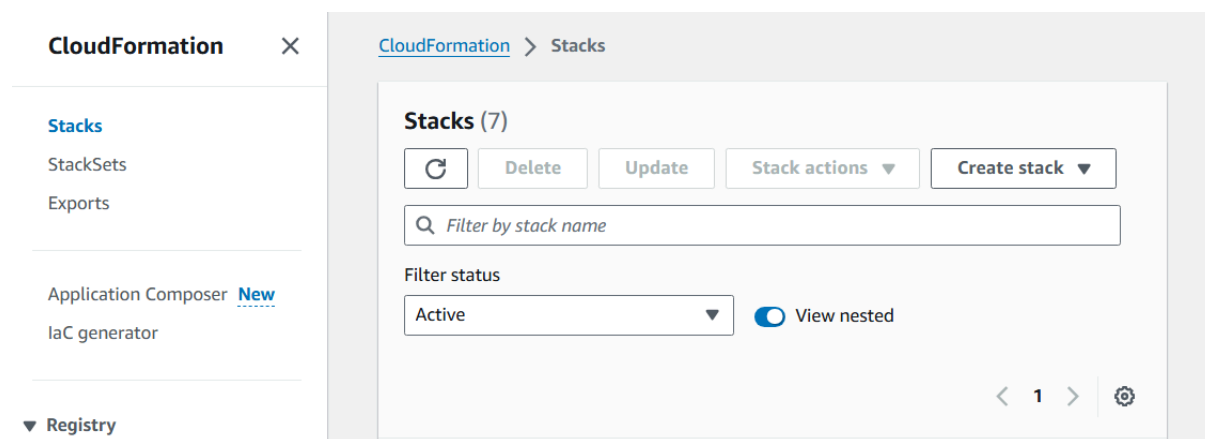
In the **Static_Check stage**, a static analysis of the code is performed using an AWS Lambda function named CFNValidateLambda. The Lambda function uses regular expression language to find patterns and identify security group policy violations, then Lambda fails the pipeline.

Build specification files, commonly referred to as build spec files, are YAML formatted files used by AWS CodeBuild to define the commands and settings for build projects.

Task 1 - Building the Pipeline

1.1 Search and open CloudFormation in AWS


AWS CloudFormation is a tool provided by Amazon Web Services (AWS) that helps you set up and manage AWS resources using code. Instead of manually configuring resources, you define what you need in a text file, and CloudFormation takes care of the rest.




1.2 Click on create standard stack

A stack is a collection of AWS resources that you can manage as a single unit. Stacks enable you to create, update, and delete related resources together, ensuring they are managed consistently.

Stacks (7)

 **Delete** **Update** **Stack actions ▼** **Create stack ▲**

 *Filter by stack name*

Filter status

- With new resources (standard)
- With existing resources (import resources)

1.3 For the template, choose existing stack

Create stack

Prerequisite - Prepare template

Prepare template
Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

☒ **Choose an existing template**
Upload or choose an existing template.

☐ **Use a sample template**
Choose from our sample template library.

☐ **Build from Application Composer**
Create a template using a visual builder.

1.4 Specify the template from S3 bucket, whose location is in lab manual

Specify template

A template is a JSON or YAML file that describes your stack's resources and properties.

Template source

Selecting a template generates an Amazon S3 URL where it will be stored.

- ☒ **Amazon S3 URL**
Provide an Amazon S3 URL to your template.

- ☐ **Upload a template file**
Upload your template directly to the console.

- ☐ **Sync from Git - new**
Sync a template from your Git repository.

Amazon S3 URL

<https://ap-south-1-tcprod.s3.amazonaws.com/courses/ILT-TF-200-DEVOPS/>

Amazon S3 template URL

S3 URL: <https://ap-south-1-tcprod.s3.amazonaws.com/courses/ILT-TF-200-DEVOPS/v3.6.4.prod-bb9fae11/lab-6-DevOpsTools/scripts/releasePipelineTemplate.yaml>

[View in Application Composer](#)

1.5 Click next

1.6 Give the stack name, and next

Specify stack details

Provide a stack name

Stack name

Stack name must be 1 to 128 characters, start with a letter, and only contain alphanumeric characters. Character count: 15/128.

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

No parameters

There are no parameters defined in your template

Cancel

Previous

Next

1.7 Review the inputs, and proceed

Review and create

Step 1: Specify template

[Edit](#)

Prerequisite - Prepare template

Template

Template is ready

Template

Template URL

`https://ap-south-1-tcprod.s3.amazonaws.com/courses/ILT-TF-200-DEVOPS/v3.6.4.prod-bb9fae11/lab-6-DevOpsTools/scripts/releasePipelineTemplate.yaml`

Stack description

-

Step 2: Specify stack details

[Edit](#)

Provide a stack name

Stack name

releasePipeline

1.8 CloudFormation has finished and the pipeline is ready to use.

releasePipeline



2024-05-25 17:38:22 UTC+0530



CREATE_COMPLETE

Task 2 - Fixing the build stage

2.1 Open CodePipeline, and choose release pipeline link

[Developer Tools](#) > [CodePipeline](#) > Pipelines

Pipelines Info

Notify ▼

View history

Release change

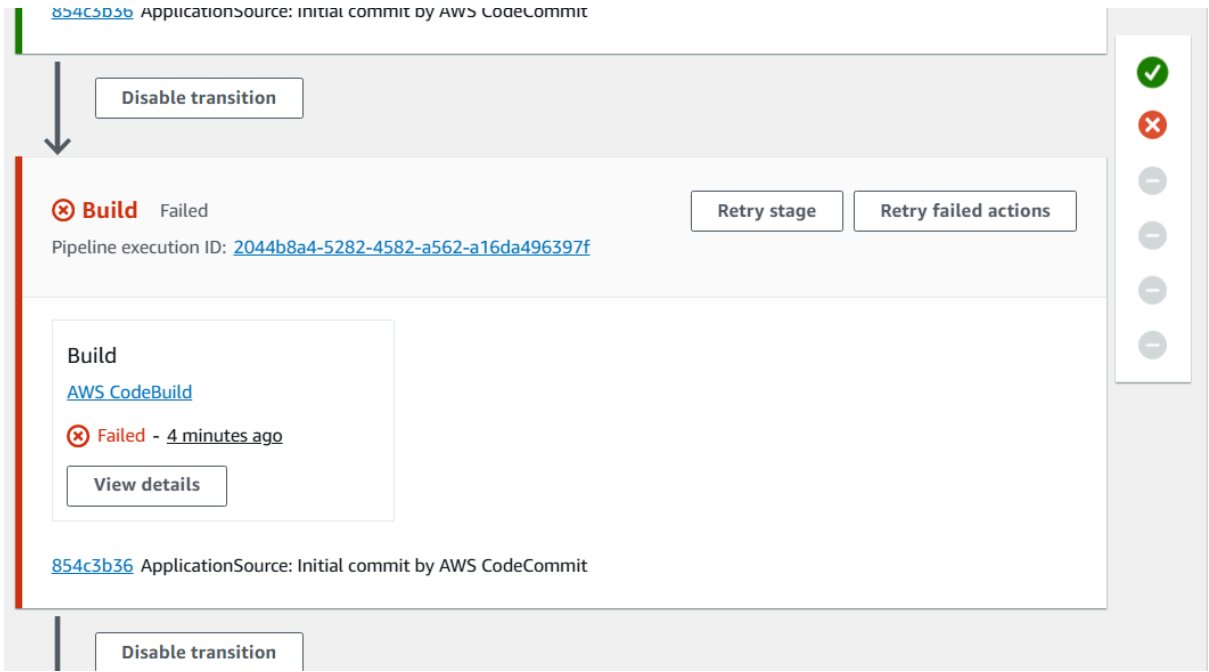
Delete pipeline

Create pipeline

< 1 >

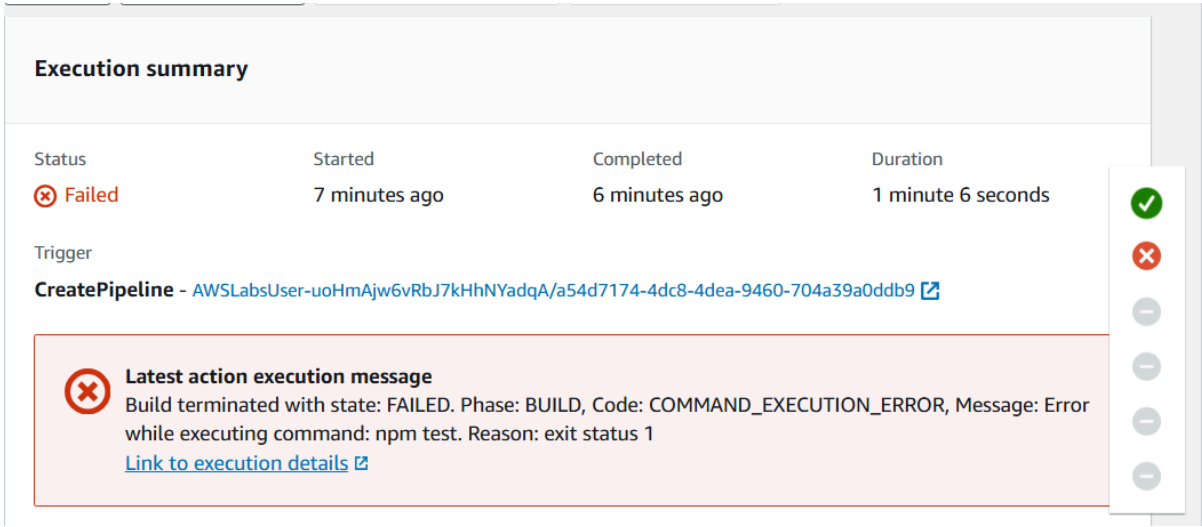
	Name	Latest execution status	Latest source revisions	Latest execution started	Most recent executions
	<div>releasePipeline</div> <div>(Type: V1 Execution mode: SUPERSEDED)</div>	<div> Failed</div>	<div>Application Source – 854c3b36:</div> <div>Initial commit by AWS CodeCommit</div>	<div>4 minutes ago</div>	<div> View details</div>

2.2 We see that the build phase for the pipeline has failed.





2.3 Open pipeline execution id, last snapshot

2.4 Select link to exec. Details



CodeBuild page opens and the CodeBuild page shows what happened during the stage's action.

Build status		
Status	Initiator	Build ARN
 Failed	codepipeline/releasePipeline	 <code>arn:aws:codebuild:ap-south-1:773946498214:build/Lab6-BuildProject:ee718346-d3cd-45cc-a489-085d2f73d645</code>
Resolved source version	Start time	End time
<code>854c3b36d0004374991b63f319592edc9827230a</code>	May 25, 2024 5:38 PM (UTC+5:30)	May 25, 2024 5:39 PM (UTC+5:30)
Build number		
1		

2.5 Scroll down to see the build logs

Entries are timestamped, indicate what ran, and where it ran.

Build logsPhase detailsReportsEnvironment variablesBuild detailsResource utilization

Showing the last 98 lines of the build log. [View entire log](#)

Tail logs

No previous logs

1 [Container] 2024/05/25 12:08:34.024489 Running on CodeBuild On-demand

2 [Container] 2024/05/25 12:08:34.024500 Waiting for agent ping

3 [Container] 2024/05/25 12:08:34.226303 Waiting for DOWNLOAD_SOURCE

4 [Container] 2024/05/25 12:08:35.454114 Phase is DOWNLOAD_SOURCE

5 [Container] 2024/05/25 12:08:35.455237 CODEBUILD_SRC_DIR=/codebuild/output/src3184449181/src

6 [Container] 2024/05/25 12:08:35.455666 YAML location is

7 [Container] 2024/05/25 12:08:35.457783 Setting HTTP client timeout to higher timeout for S3 source

8 [Container] 2024/05/25 12:08:35.457883 Processing environment variables

9 [Container] 2024/05/25 12:08:35.630652 Selecting 'nodejs' runtime version '12' based on manual

2.6 Scroll down to the very bottom, it shows which state failed

```
81 npm ERR! Test failed. See above for more details.
82
83 [Container] 2024/05/25 12:09:27.654242 Command did not exit successfully npm test exit status 1
84 [Container] 2024/05/25 12:09:27.659005 Phase complete: BUILD State: FAILED
```

2.7 Scroll just up, and see the cause of the error

Here, it is an assertion error (4==0)

```

66 1) Calculator Tests
67   Multiply Tests
68     returns 0 * 4 = 0:
69
70   AssertionError [ERR_ASSERTION]: 4 == 0
71     + expected - actual
72
73     -4
74     +0
75
76   at Context.<anonymous> (test/calculator_spec.js:37:11)
77   at processImmediate (internal/timers.js:461:21)

```

Another way to phrase what the Assertion Error in the log is saying is, The expected output of the test and the actual output from the test when it ran, are not equivalent to each other.

2.8 Note down the following:

1. Who caused the error: Calculator multiplication tests returned an error during the build process.
2. What the type of error was: Assertion Error in Node.js.
3. When in the build process the error occurred: During the build phase (timestamps are available, if needed).
4. Where the error occurred in the test code: The file path and file name of the test definitions is test/calculator_spec.js. The line number and character location is 37:11.
5. Why this error occurred: The root cause of the error was a mismatch between the value returned from the test and the expected value defined for the test.
6. How this error occurred: Likely a human error, such as a typo, occurred when creating the test code.

Task 3 - Fixing error in the test code

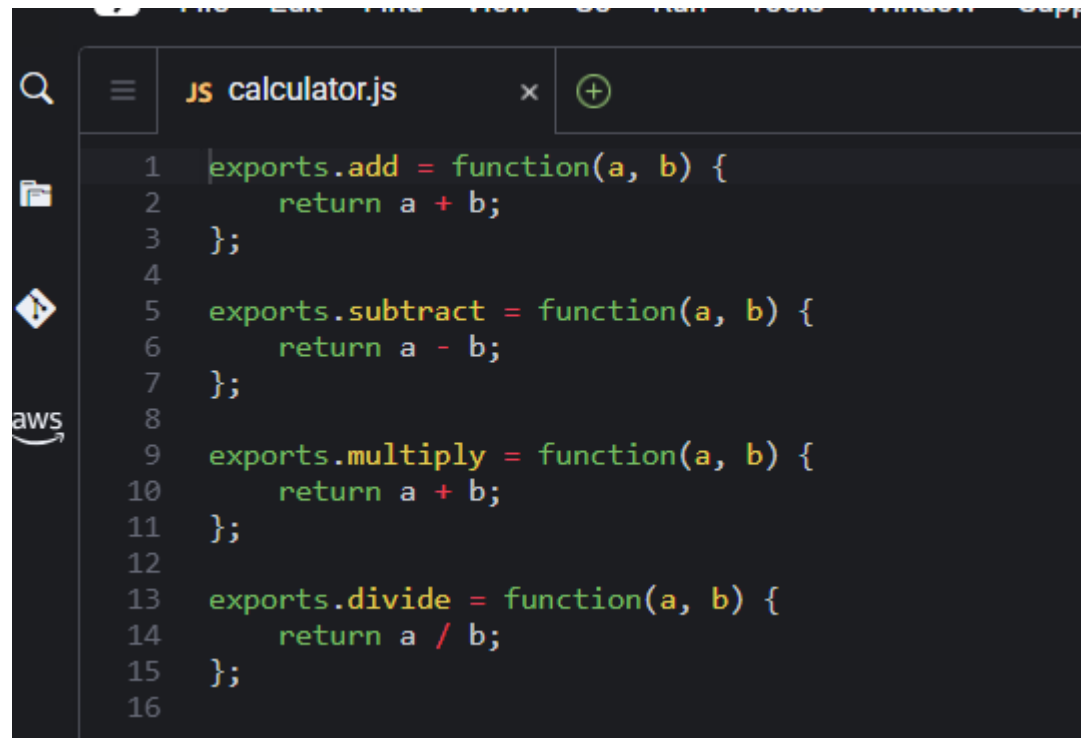
3.1 Open cloud9, and clone the repo in IDE

```

AWSLabsUser-uoHmAjw6vRbJ7kHhNYadqA:~/environment $ /tmp/git-cloning-runner-1716640093512-012920289612.sh
Cloning into '/home/ec2-user/environment/Lab6'...

```

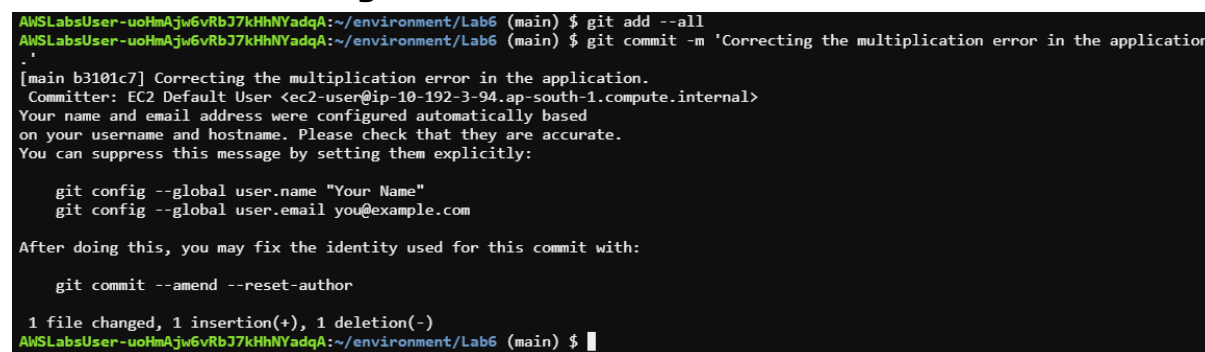
3.2 Open the calculator file code using the file explorer



```
1 exports.add = function(a, b) {
2   return a + b;
3 };
4
5 exports.subtract = function(a, b) {
6   return a - b;
7 };
8
9 exports.multiply = function(a, b) {
10  return a + b;
11 };
12
13 exports.divide = function(a, b) {
14   return a / b;
15 };
16
```

3.3 Change line 10 to a*b

3.4 Commit the changes



```
AWSLabsUser-uoHmAjw6vRbJ7kHhNYadqA:~/environment/Lab6 (main) $ git add --all
AWSLabsUser-uoHmAjw6vRbJ7kHhNYadqA:~/environment/Lab6 (main) $ git commit -m 'Correcting the multiplication error in the application'
[main b3101c7] Correcting the multiplication error in the application.
Committer: EC2 Default User <ec2-user@ip-10-192-3-94.ap-south-1.compute.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 1 insertion(+), 1 deletion(-)
AWSLabsUser-uoHmAjw6vRbJ7kHhNYadqA:~/environment/Lab6 (main) $
```

3.5 OPen codepipeline again, we see build stage has passed, but static check has failed

The screenshot displays the AWS CodePipeline console for a pipeline execution with ID `c1f50556-9b16-4a13-b97b-df7d960e6cf8`. The 'Build' stage, using 'AWS CodeBuild', is marked as 'Succeeded' and completed 2 minutes ago. Below it, a message indicates an 'ApplicationSource' error: 'Correcting the multiplication error in the application.' A 'Disable transition' button is visible. The 'Static_Check' stage, using 'AWS Lambda', is marked as 'Failed' and completed 1 minute ago. It includes 'Retry stage' and 'Retry failed actions' buttons. A vertical bar on the right side of the console shows the status of stages: green checkmarks for successful stages and a red 'X' for failed stages. The footer contains links for 'Shell', 'Feedback', and copyright information for Amazon Web Services, Inc.

Build Succeeded
Pipeline execution ID: [c1f50556-9b16-4a13-b97b-df7d960e6cf8](#)

Build
[AWS CodeBuild](#)
Succeeded - 2 minutes ago
[View details](#)

[b3101c7f](#) ApplicationSource: Correcting the multiplication error in the application.

Disable transition


Static_Check Failed
Pipeline execution ID: [c1f50556-9b16-4a13-b97b-df7d960e6cf8](#)

Retry stage Retry failed actions

stack_validate
[AWS Lambda](#)
Failed - 1 minute ago
[View details](#)

Shell Feedback © 2024 Amazon Web Services, Inc. or its affiliates Privacy Terms Cookie preferences

Task 4: Fixing the Lambda Static check

 **Consider:** You may be wondering why this didn't fail the first time the pipeline ran. The code is intentionally written to not process any rules the first time in order to change the order of your troubleshooting. By instructing you to fix the build stage first, you can examine the Lambda function logs while you wait for the initial test environment deployment.

4.1 Click on the Pipeline execution ID

⊗ **Static_Check** Failed

Pipeline execution ID: [c1f50556-9b16-4a13-b97b-df7d960e6cf8](#)

4.2 error:

The Action execution failed message dialog box displays the error returned from the Lambda CFNValidateLambda function. The reason for the failure is that the function found global access to the SSH network ingress ports.

Status	Started	Completed	Duration
⊗ Failed	7 minutes ago	5 minutes ago	2 minutes 8 seconds

Trigger
CloudWatchEvent - [releasePipeline-SourceEvent](#) [🔗](#)

⊗ **Latest action execution message**
Function exception: Failed filters ['IngressOpenToWorld', 'SSHOpenToWorld']
[Link to execution details](#) [🔗](#)

Pipeline execution ID
📄 c1f50556-9b16-4a13-b97b-df7d960e6cf8

4.3 Return to Cloud9

4.4 Open application.json file from the left pane explorer

```
{} application.json x +
29
30     {
31         "Ref": "AWS::StackName"
32     },
33     "security-group"
34 ]
35 },
36 "GroupDescription": "Security group that allows SSH ingress from all over the world!",
37 "VpcId": {
38     "Fn::ImportValue": {
39         "Fn::Sub": "${SharedResourceStack}:VPC"
40     }
41 },
42 "SecurityGroupIngress": [
43     {
44         "IpProtocol": "tcp",
45         "FromPort": 22,
46         "ToPort": 22,
47         "CidrIp": "0.0.0.0/0"
48     }
49 ]
50 },
51 },
52 "DeploymentGroup": {
53     "Type": "AWS::CodeDeploy::DeploymentGroup",
54     "Properties": {
55         "DeploymentGroupName": {
56             "Fn::Sub": "${SharedResourceStack}:DeploymentGroup"
57         }
58     }
59 },
60 "Tags": [
61     {
62         "Key": "Name",
63         "Value": "${AWS::StackName}-SecurityGroup"
64     }
65 ]
66 },
67 "Outputs": {
68     "SecurityGroupId": {
69         "Value": {"Ref": "SecurityGroup"},
70         "Export": "true"
71     }
72 }
73 }
```

(7 Bytes) 47:43

```
bash - "ip-10-192-3-94.a x Immediate x bash - "ip-10-192-3-94.a x +
git config --global user.name "Your Name"
git config --global user.email you@example.com
```

4.5 On Line 47, change "CidrIp": "0.0.0.0/0" so that the value is your own private IP address instead, to allow access only from your IP address

```
},
"SecurityGroupIngress": [
    {
        "IpProtocol": "tcp",
        "FromPort": 22,
        "ToPort": 22,
        "CidrIp": "152.58.76.163/32"
    }
]
```

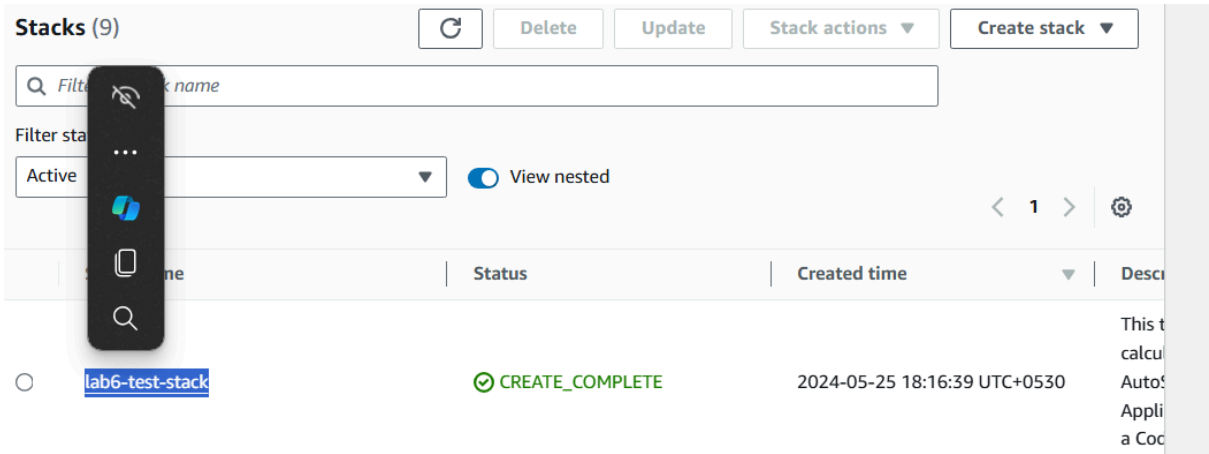
4.6 Save and push the changes

Task 5: Validating in the Test Environment

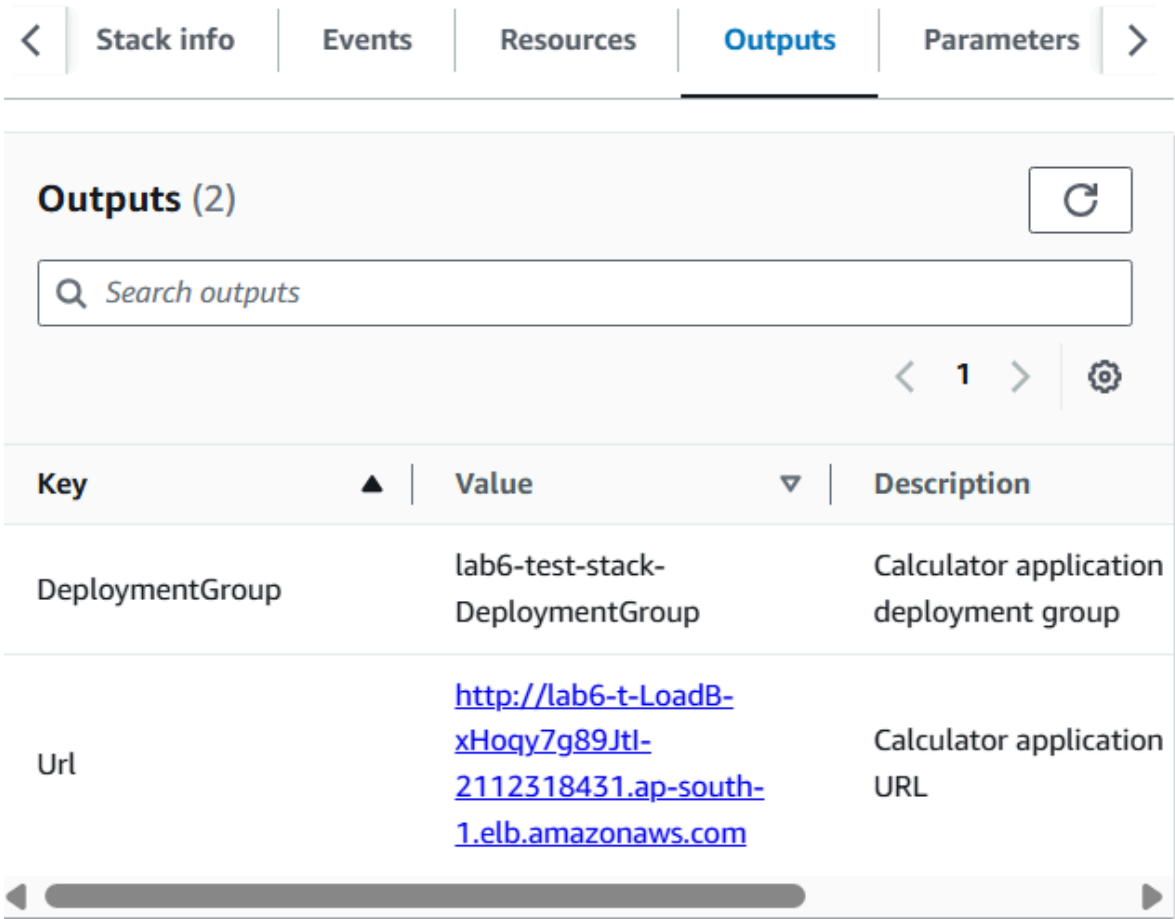
Your pipeline uses these deployment action providers -

- 1. AWS CloudFormation** for deploying immutable infrastructure
- 2. AWS CodeDeploy** for deploying applications.

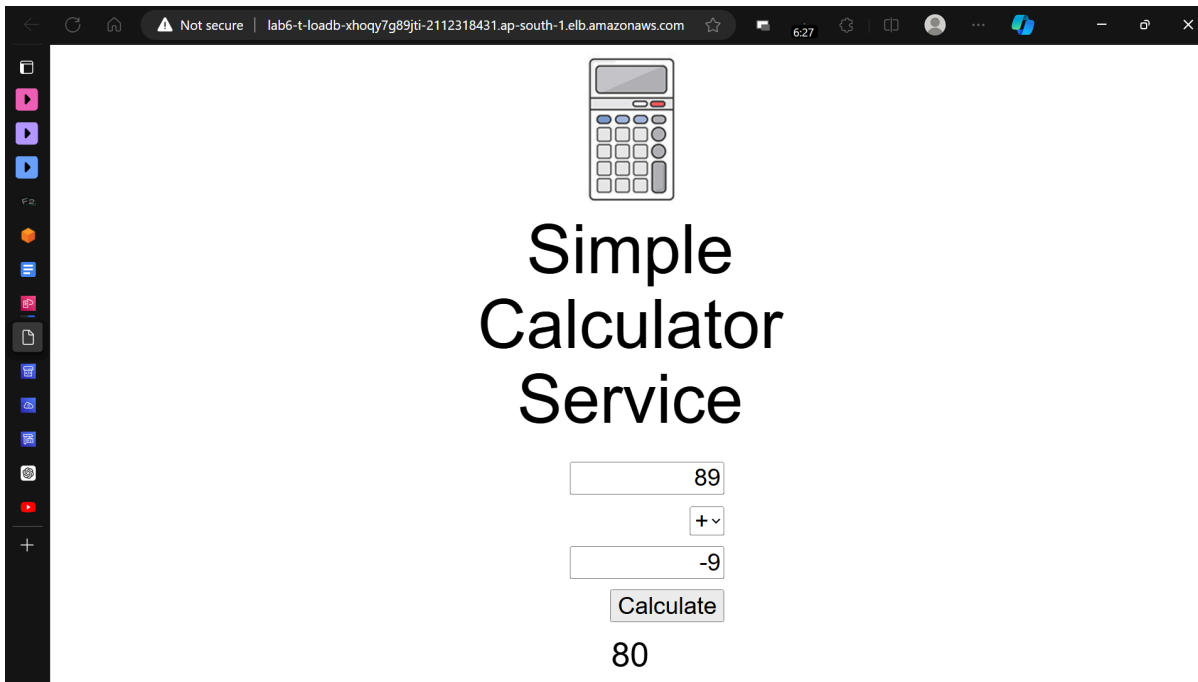
5.1 Open CloudFormation, and select test-stack



5.2 Open Outputs tab, and open the URL in new tab



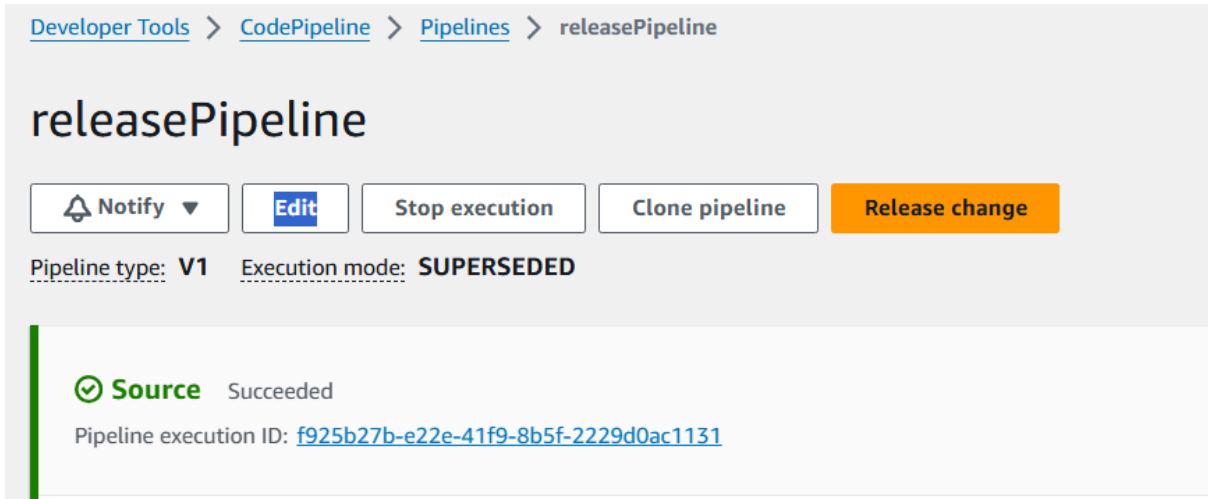
5.3 Our app opens, and it works!!!



Task 6: Automating the testing of the test environment

In a previous task, we manually validated and tested that the web application site was responsive. In this task, we use an AWS Lambda function in the release pipeline to automate the validation step for the test environment.

6.1 Go to CodePipeline and choose edit



6.2 Choose edit stage in test_stack

Edit: Test_Stack

Edit stage

GenerateChangeSet
AWS CloudFormation

↓

DeployChangeSet
AWS CloudFormation

↓

Deploy
[AWS CodeDeploy](#)

☐ Configure automatic rollback on stage failure

6.3 Click on edit in DeployChangeSet

Edit: Test_Stack

Cancel

Delete

Done

+ Add action group

GenerateChangeSet
AWS CloudFormation

+ Add action

↓

DeployChangeSet
AWS CloudFormation

+ Add action

↓

Deploy
[AWS CodeDeploy](#)

+ Add action

Edit action

6.4 Fill this value

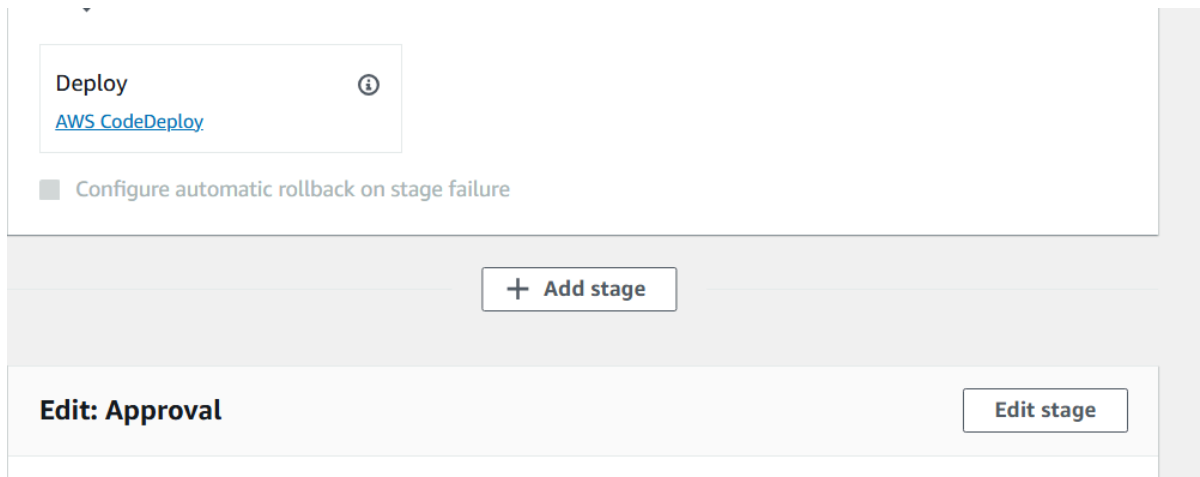
Variable namespace - *optional*

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](#)

TestStackOutput

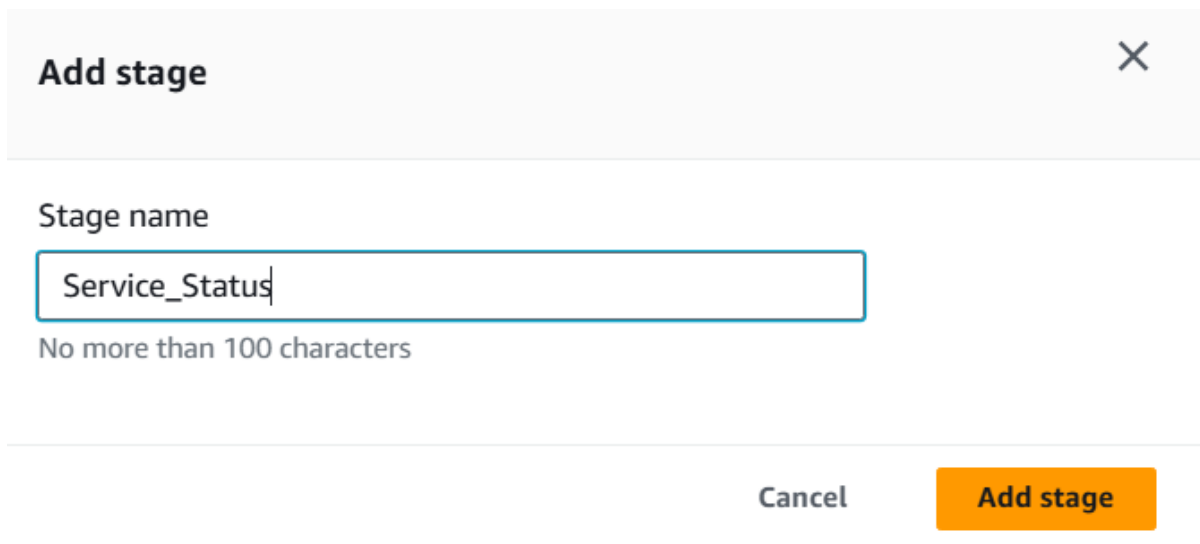
6.5 choose next

6.6 Click on add stage option after



The screenshot shows a configuration interface for a stage. At the top, there is a box labeled "Deploy" with a link to "AWS CodeDeploy" and an information icon. Below this is a checkbox labeled "Configure automatic rollback on stage failure". In the center, there is a button labeled "+ Add stage". At the bottom, there is a section labeled "Edit: Approval" with an "Edit stage" button.

6.7



The screenshot shows a dialog box titled "Add stage" with a close button (X) in the top right corner. Inside the dialog, there is a label "Stage name" above a text input field. The input field contains the text "Service_Status". Below the input field, there is a note: "No more than 100 characters". At the bottom of the dialog, there are two buttons: "Cancel" and "Add stage".

6.8 Choose add action grp

Edit: Service_Status

CancelDeleteDone

+ Add action group

☐ Configure automatic rollback on stage failure

6.9 Following values

For **Action name** enter

Test_Status

For **Action provider** choose **AWS Lambda**

For **Region** , keep the default value. It should be the same AWS Region where your lab was launched or the same AWS Region as your pipeline.

For **Function name** , choose the function name that contains the string **servicestatus** .

For **User parameters** - *optional*, enter

#{TestStackOutput.Url}

Choose

Done

 .

Click on done

6.10 Save the edits in pipeline

[Developer Tools](#) > [CodePipeline](#) > [Pipelines](#) > [releasePipeline](#) > Edit releasePipeline

Editing: releasePipeline

DeleteCancelSave

To test the new pipeline stage with the Lambda function, release the most recent change through the pipeline.

6.11 Click on release Pipeline

Release change



Releasing a change will detect the most recent change in each location configured in your source action(s), and run that change through the pipeline. Do you want to continue?

► **Source revision overrides**

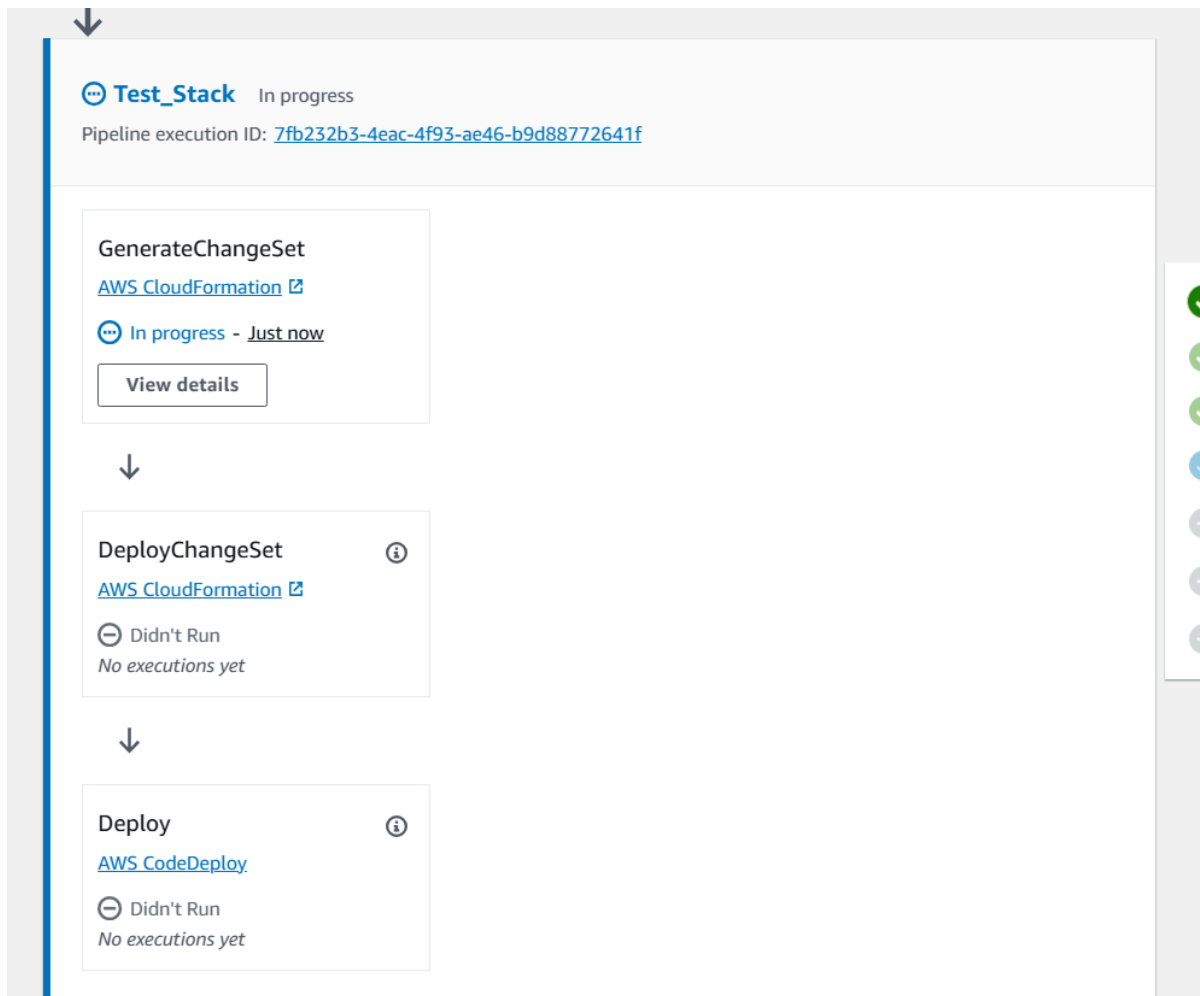
A source revision is the version of the source artifact with all the changes to your application code for the pipeline execution.

Cancel

Release

6.12

The most recent change will re-run through the pipeline. It might take a few moments for the status of the run to show in the pipeline view.



NOTE - We used the CodePipeline console to configure the DeployChangeSet action in the Test_Stack stage to export its CloudFormation outputs as variables. You then added a new stage in the release pipeline and added an action in that stage and tested the new stage.

Task 7: Approving the deletion of the Test environment and build the Production environment

7.1 On the CodePipeline console, locate the Approval stage of the pipeline.

Approval Pending

Pipeline execution ID: [7fb232b3-4eac-4f93-ae46-b9d88772641f](#)

Purge_Test

Manual approval

⌚ Waiting for approval -

Review

↓

Delete_Test_Stack ⓘ

[AWS CloudFormation](#) ↗

⌚ Didn't Run

No executions yet

[db3f72d6](#) ApplicationSource: updating TestSecurityGroup ingress rule from public to private address.

It is in pending state

7.2 In the Purge_Test action, choose Review .

Purge_Test

Manual approval

⌚ Waiting for approval -

Review

7.3 choose approve and submit

Decision

☒ Approve

Approving will resume the pipeline execution.

☐ Reject

Rejecting will stop the pipeline execution with a failed status.

Comments - *optional*



Preview markdown

[Learn more](#)

Cancel

Submit

7.4 The pipeline performs actions with CloudFormation to delete the test environment stack.

Delete_Test_Stack

[AWS CloudFormation](#)

In progress - Just now

View details

Task 9: Verify the application deployment to Production environment

✓ Verify:

- Test environment was deleted to avoid the unnecessary resource consumption.
- CloudFormation provisioned the Production environment.
- CodeDeploy deployed the web application to Production environment.

We have to verify:

9.1 Open CloudFormation again, and we no longer see a stack that contains the string test-stack.

Test-stack shows as deleted;
Only prod-stack is available

The screenshot displays the AWS CloudFormation console. On the left, the 'Stacks (9)' list shows three stacks: 'lab6-prod-stack' (status: CREATE_COMPLETE), 'releasePipeline' (status: CREATE_COMPLETE), and 'aws-cloud9-Lab-6-'. The 'lab6-test-stack' is not visible in this list. On the right, the 'lab6-test-stack' details page is shown, indicating its status as 'DELETE_COMPLETE'. The 'Overview' section provides details about the stack, including its ID, description, and status.

Stacks (9)

Filter by stack name

Filter status: Active View nested

Stacks

Stack	Created	Status
lab6-prod-stack	2024-05-25 18:48:24 UTC+0530	CREATE_COMPLETE
releasePipeline	2024-05-25 17:38:22 UTC+0530	CREATE_COMPLETE
aws-cloud9-Lab-6-		

lab6-test-stack

Delete Update Stack actions Create stack

Stack info Events Resources Outputs Parameters

Overview

Property	Value
Stack ID	arn:aws:cloudformation:ap-south-1:773946498214:stack/lab6-test-stack/d48a2540-1a94-11ef-98bb-06dbc649f672
Description	This template sets up the sample calculator application on an AutoScaling group, behind an Application Load Balancer, with a CodeDeploy application.
Status	DELETE_COMPLETE
Detailed status	-
Status reason	-
Root stack	-
Parent stack	-
Created time	-

9.2 Open the url from prod-stack, to see the calculator once again

