

Nnadiekwe, Chiderah David

Project- 3 Stage CICD pipeline to deploy an application

- Build a three stages pipeline to deploy a given Application using CloudFormation as a deploy provider.

- Source your code from CodeCommit.

- Your application must be running on a custom VPC.

- Your application must be running on a custom VPC.

- First, I created a CodeCommit repository

The screenshot shows the AWS CodeCommit interface. On the left, a sidebar menu includes 'Source • CodeCommit', 'Getting started', 'Repositories' (which is selected), 'Approval rule templates', 'Artifacts • CodeArtifact', 'Build • CodeBuild', 'Deploy • CodeDeploy', 'Pipeline • CodePipeline', and 'Settings'. Below these are links for 'Go to resource' and 'Feedback'. The main content area is titled 'Developer Tools > CodeCommit > Repositories'. It displays a table with one row for the repository '3_Stage_CICD_Pipeline'. The table columns are 'Name', 'Description', 'Last modified', and 'Clone URL'. The repository details show: 'Name' is '3_Stage_CICD_Pipeline'; 'Description' is 'This repo will contain a CloudFormation file needed to deploy a python application on codebuild'; 'Last modified' is '43 minutes ago'; and 'Clone URL' includes 'HTTPS' and 'SSH' options. At the top right of the main area, there are buttons for 'Notify', 'Clone URL', 'View repository', 'Delete repository', and 'Create repository'.

Then I created https git credentials for AWS code commit through IAM by selecting the user and then “security credentials”

The screenshot shows the AWS IAM interface. The left sidebar includes 'Identity and Access Management (IAM)', 'Dashboard', 'Access management' (with 'User groups' and 'Users' selected), 'Access reports' (with 'Access analyzer', 'Archive rules', 'Analyzers', 'Settings', 'Credential report', and 'Organization activity'), and 'Service control policies (SCPs)'. The main content area is titled 'HTTPS Git credentials for AWS CodeCommit (1)'. It shows a table with one row for the user 'admin-at-968047596089'. The table columns are 'User name', 'Created', and 'Status'. The user details show: 'User name' is 'admin-at-968047596089'; 'Created' is 'Now'; and 'Status' is 'Active'. A red box highlights this row. Below this section is another table titled 'Credentials for Amazon Keypaces (for Apache Cassandra) (0)' with a note 'No credentials' and a 'Generate credentials' button.

The screenshot shows the AWS CodeCommit console. The left sidebar has a tree view with 'Source' expanded, showing 'CodeCommit' selected. Under 'CodeCommit', 'Code' is expanded, showing 'Pull requests', 'Commits', 'Branches', 'Git tags', and 'Settings'. Other sections like 'Artifacts', 'Build', 'Deploy', 'Pipeline', and 'Settings' are also listed. The main content area is titled '3_Stage_CICD_Pipeline'. It shows 'Connection steps' with tabs for 'HTTPS' (selected), 'SSH', and 'HTTPS (GRC)'. Below the tabs, there are three sections: 'Step 1: Prerequisites', 'Step 2: Git credentials', and 'Step 3: Clone the repository'. The 'Clone the repository' section contains a command-line input field with the URL 'git clone https://git-codecommit.us-east-1.amazonaws.com/v1/repos/3_Stage_CICD_Pipeline'. To the right of the input field is a green 'Copied' button with a checkmark icon.

Next, I cloned the CodeComit repo to my local repo using the https git credentials for codecommit

```
C: > Users > 17093 > Desktop > cloud computing > DevOps homeworks > Resume_challenge > index.html > html > h2
$ git init
Initialized empty Git repository in C:/Users/17093/Desktop/cloud computing/DevOps homeworks/3 stage CICD pipeline/.git/
17093@LAPTOP-BUUBLOVN MINGW64 ~/Desktop/cloud computing/DevOps Homeworks/3 stage CICD pipeline (master)
$ git clone https://git-codecommit.us-east-1.amazonaws.com/v1/repos/3_Stage_CICD_Pipeline
Cloning into '3_Stage_CICD_Pipeline'...
warning: You appear to have cloned an empty repository.

17093@LAPTOP-BUUBLOVN MINGW64 ~/Desktop/cloud computing/DevOps Homeworks/3 stage CICD pipeline (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    3_Stage_CICD_Pipeline

nothing added to commit but untracked files present (use "git add" to track)

17093@LAPTOP-BUUBLOVN MINGW64 ~/Desktop/cloud computing/DevOps Homeworks/3 stage CICD pipeline (master)
$ rm -rf 3_Stage_CICD_Pipeline/
17093@LAPTOP-BUUBLOVN MINGW64 ~/Desktop/cloud computing/DevOps Homeworks/3 stage CICD pipeline (master)
$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)

17093@LAPTOP-BUUBLOVN MINGW64 ~/Desktop/cloud computing/DevOps Homeworks/3 stage CICD pipeline (master)
$ ls -l
total 0
17093@LAPTOP-BUUBLOVN MINGW64 ~/Desktop/cloud computing/DevOps Homeworks/3 stage CICD pipeline (master)
$
```

Ln 11, Col 5 Spaces: 2 UTF-8 LF HTML

Successfully cloned!

- Next, I created an instance in a custom VPC to manually test the CloudFormation template using cfn-lint

The screenshot shows the 'Launch an instance' wizard in the AWS EC2 console. The 'Name and tags' section has 'local-testing-server' entered. The 'Application and OS Images (Amazon Machine Image)' section shows 'Amazon Linux 2023 AMI 2023.2.2...' selected. The 'Virtual server type (instance type)' is set to 't2.micro'. A tooltip for the free tier is visible, stating: 'Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month.' The 'Launch instance' button is at the bottom right.

The instance should have SSH enabled

The screenshot shows the 'Instances' page in the AWS EC2 console. It lists three instances: 'local-testing-server' (running, t2.micro), 'i-0b79f9957c1bb2950' (terminated, t3.micro), and 'i-06efcd792df5b6a6f' (terminated, t3.micro). The 'local-testing-server' instance is selected. The 'Details' tab for this instance shows its public IPv4 address (34.200.234.222), private IPv4 address (10.0.1.31), and private IP DNS name (ip-10-0-1-31.ec2.internal). The 'Actions' dropdown menu is open, showing options like 'Stop', 'Start', 'Reboot', and 'Terminate'.

Next, I SSH into the instance using session manager to install cfn-lint. Cfn-lint requires python 3.7 and above to run

```
Session ID: admin-0ae6cb78dad0af1c2 Instance ID: i-0039d6e8f66c89353 Terminate

sh-5.2$ sudo su - ec2-user
[ec2-user@ip-10-0-1-31 ~]$ python3 --version
Python 3.9.16
[ec2-user@ip-10-0-1-31 ~]$
```

As seen, the python version in the Linux instance is 3.9.16 which meets the requirement to install cfn lint. So, we're good to go!

Next, we checked if pip3 is installed. Pip3 is the package manager of python. I checked using the command "pip3 –version". As seen, it says the "command is not found" hence, I had to install the pip3 using the command "yum install python3-pip" as a super user (sudo).

```
Session ID: admin-0ae6cb78dad0af1c2 Instance ID: i-0039d6e8f66c89353 Terminate

[ec2-user@ip-10-0-1-31 ~]$ pip3 --version
-bash: pip3: command not found
[ec2-user@ip-10-0-1-31 ~]$ yum install python3-pip
Error: This command has to be run with superuser privileges (under the root user on most systems).
[ec2-user@ip-10-0-1-31 ~]$ sudo su
[root@ip-10-0-1-31 ec2-user]# yum install python3-pip
Last metadata expiration check: 0:18:28 ago on Mon Oct  9 03:17:50 2023.
Dependencies resolved.
=====
 Package          Architecture Version      Repository   Size
=====
Installing:
 python3-pip      noarch      21.3.1-2.amzn2023.0.5 amazonlinux 1.8 M
Installing weak dependencies:
 libxcrypt-compat x86_64     4.4.33-7.amzn2023 amazonlinux 92 k
=====
Transaction Summary
Install 2 Packages

Total download size: 1.9 M
Installed size: 11 M
Is this ok [y/N]: y
Downloading Packages:
(1/2): libxcrypt-compat-4.4.33-7.amzn2023.x86_64.rpm 1.1 MB/s | 92 kB    00:00
(2/2): python3-pip=21.3.1-2.amzn2023.0.5.noarch.rpm 13 MB/s | 1.8 MB    00:00
=====
9.7 MB/s | 1.9 MB    00:00

Total
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing : 1/1
  Installing : libxcrypt-compat-4.4.33-7.amzn2023.x86_64 1/2
```

Now, I recheck the pip3 version using the "pip3 –version" command

```

libcrypt-compat           x86_64          4.4.33-7.amzn2023      amazonlinux          92 kB
Transaction Summary
Install 2 Packages
Total download size: 1.9 M
Installed size: 11 M
Is this ok [y/N]: y
Downloading Packages:
(1/2) libcrypt-compat-4.4.33-7.amzn2023.x86_64.rpm          1.1 MB/s |  92 kB   00:00
(2/2) python3-pip-21.3.1-2.amzn2023.0.5.noarch.rpm          13 MB/s | 1.8 MB   00:00
Total                                         9.7 MB/s | 1.9 MB   00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing                                           : 1/1
  Installing  libcrypt-compat-4.4.33-7.amzn2023.x86_64        1/2
  Installing  python3-pip-21.3.1-2.amzn2023.0.5.noarch       2/2
  Running scriptlet: python3-pip-21.3.1-2.amzn2023.0.5.noarch 2/2
  Verifying    libcrypt-compat-4.4.33-7.amzn2023.x86_64        1/2
  Verifying    python3-pip-21.3.1-2.amzn2023.0.5.noarch       2/2
Installed:
  libcrypt-compat-4.4.33-7.amzn2023.x86_64                  python3-pip-21.3.1-2.amzn2023.0.5.noarch
Complete!
[ec2-user@ip-10-0-1-31 ~]$ sudo su - ec2-user
Last login: Mon Oct  9 03:25:17 UTC 2023 on pts/0
[ec2-user@ip-10-0-1-31 ~]$ pip3 --version
pip 21.3.1 from /usr/lib/python3.9/site-packages/pip (python 3.9)
[ec2-user@ip-10-0-1-31 ~]$ 

```

Now, it shows that the version installed is pip 21.3.1

Next, I install the cfn-lint on the instance using the command “pip3 install cfn-lint”

```

Session ID: admin-0ae6cb78ada0af1c2           Instance ID: i-0039d6e8f66c89353      Terminate
[ec2-user@ip-10-0-1-31 ~]$ pip3 install cfn-lint
Defaulting to user installation because normal site-packages is not writeable
Collecting cfn-lint
  Downloading cfn_lint-0.81.0-py3-none-any.whl (3.5 MB)
    |
    |██████████| 3.5 MB 6.2 MB/s
Collecting networkx<4,>=2.4
  Downloading networkx-3.1-py3-none-any.whl (2.1 MB)
    |
    |██████████| 2.1 MB 18.5 MB/s
Collecting sympy>=1.0.0
  Downloading sympy-1.12-py3-none-any.whl (5.7 MB)
    |
    |██████████| 5.7 MB 50.6 MB/s
Collecting regex==2021.7.1
  Downloading regex-2023.10.3-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (773 kB)
    |
    |██████████| 773 kB 57.6 MB/s
Requirement already satisfied: jsonpatch in /usr/lib/python3.9/site-packages (from cfn-lint) (1.21)
Requirement already satisfied: pyyaml>5.4 in /usr/lib64/python3.9/site-packages (from cfn-lint) (5.4.1)
Collecting sarif-om=<1.0.4
  Downloading sarif_om-1.0.4-py3-none-any.whl (30 kB)
Collecting jschema-to-python=<1.2.3
  Downloading jschema_to_python-1.2.3-py3-none-any.whl (10 kB)
Collecting aws-sam-translator=<1.75.0
  Downloading aws_sam_translator-1.77.0-py3-none-any.whl (377 kB)
    |
    |██████████| 377 kB 59.2 MB/s
Collecting junit-xml=<1.9
  Downloading junit_xml-1.9-py2.py3-none-any.whl (7.1 kB)
Requirement already satisfied: jsonschema<5,>=3.0 in /usr/lib/python3.9/site-packages (from cfn-lint) (3.2.0)
Collecting typing_extensions<5,>=4.4
  Downloading typing_extensions-4.0.0-py3-none-any.whl (31 kB)
Collecting pydantic<3,>=1.8
  Downloading pydantic-2.4.2-py3-none-any.whl (395 kB)
    |
    |██████████| 395 kB 55.5 MB/s
Collecting boto3==1.*,>=1.19.5
  Downloading boto3-1.28.62-py3-none-any.whl (135 kB)
    |
    |██████████| 135 kB 57.9 MB/s
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in /usr/lib/python3.9/site-packages (from boto3==1.*,>=1.19.5->aws-sam-translator>=1.75.0->cfn-lint) (0.10.0)

```

Verifying the cfn-lint installation

```

[ec2-user@ip-10-0-1-31 ~]$ cfn-lint --version
cfn-lint 0.81.0
[ec2-user@ip-10-0-1-31 ~]$ 

```

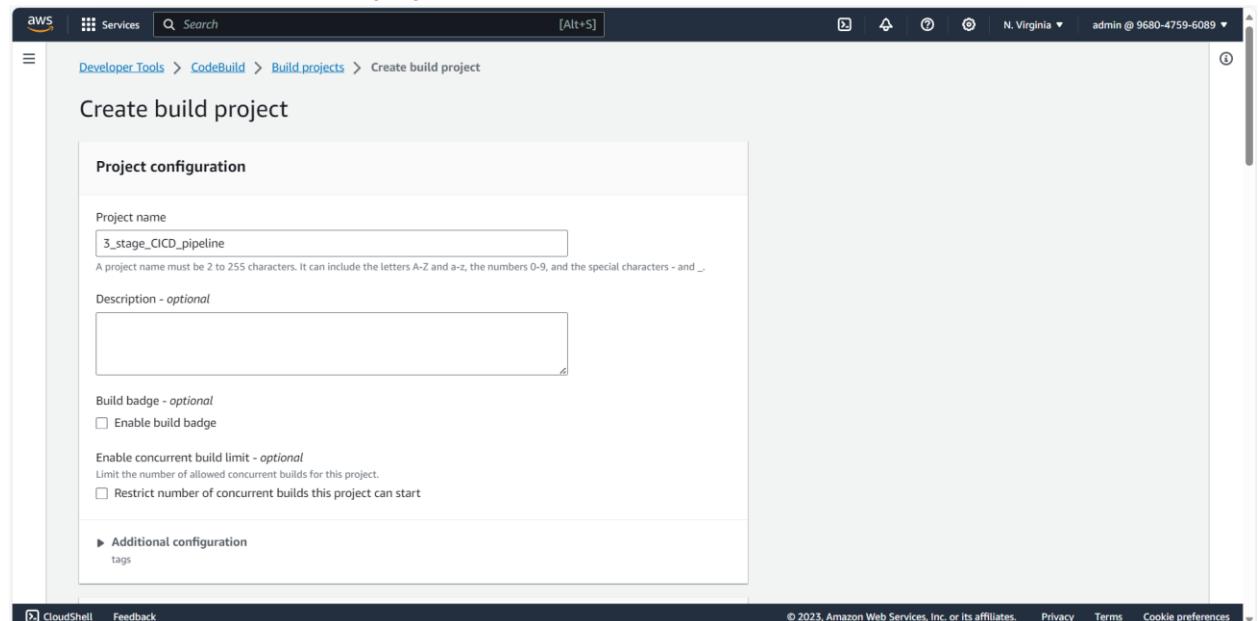
Testing my cloudformation template using cfn-lint in the ec2. My cloudformation template is "launch_instance_in_custom_vpc.yml"

```
Outputs:
  VPCID:
    Description: VPC ID
    Value: !Ref VPC
    Export:
      Name: !Sub '${AWS::StackName}-VPCID'
  PublicSubnet:
    Description: The subnet ID to use for public web servers
    Value: !Ref PublicSubnet
    Export:
      Name: !Sub '${AWS::StackName}-SubnetID' #replacing a string
  WebServerSecurityGroup:
    Description: The security group ID to use for public web servers
    Value: !GetAtt
      - WebserverSecurityGroup
      - GroupId
    Export:
      Name: !Sub '${AWS::StackName}-SecurityGroupID'

[root@ip-10-0-1-31 bin]# vim launch_instance_in_custom_vpc.yml
[root@ip-10-0-1-31 bin]# cfn-lint launch_instance_in_custom_vpc.yml
E2002 Parameter AvailabilityZone has invalid type AWS::EC2::AvailabilityZone::us-east-1a
launch_instance_in_custom_vpc.yml:19:5
E3008 Property "AvailabilityZone" can Ref to parameter of types (String, AWS::SSM::Parameter::Value<String>, AWS::EC2::AvailabilityZone::Name, AWS::SSM::Parameter::Value<AWS::EC2::AvailabilityZone::Name>) at Resources/PublicSubnet/Properties/AvailabilityZone/Ref
launch_instance_in_custom_vpc.yml:38:7

[root@ip-10-0-1-31 bin]# vim launch_instance_in_custom_vpc.yml
[root@ip-10-0-1-31 bin]# cfn-lint launch_instance_in_custom_vpc.yml
[root@ip-10-0-1-31 bin]#
```

- **Next, I created the CodeBuild project**



aws Services Search [Alt+S] N. Virginia admin @ 9680-4759-6089

Source

Source 1 - Primary

Source provider: AWS CodeCommit

Repository:

► Additional configuration
Git clone depth, Git submodules

Environment

Environment image:

Managed image Use an image managed by AWS CodeBuild

Custom image Specify a Docker image

Operating system:

Privileged

CloudShell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

For this project, I utilized Ubuntu Environment

aws Services Search [Alt+S] N. Virginia admin @ 9680-4759-6089

Environment

Environment image:

Managed image Use an image managed by AWS CodeBuild

Custom image Specify a Docker image

Operating system: Ubuntu

Runtime(s): Standard

Image: aws/codebuild/standard:5.0

Image version: Always use the latest image for this runtime version

Environment type: Linux EC2

Privileged

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

CloudShell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

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: codebuild-3_Stage_CICD_Pipeline-service-role

Type your service role name

► 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

CloudShell Feedback

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Developer Tools **CodeBuild**

- Source • CodeCommit
- Artifacts • CodeArtifact
- Build • CodeBuild
 - Getting started
 - Build projects
 - Build project**
 - Settings
 - Build history
 - Report groups
 - Report history
 - Account metrics
 - Deploy • CodeDeploy
 - Pipeline • CodePipeline
 - Settings

Go to resource Feedback

CloudShell Feedback

Project created
You have successfully created the following project: 3_Stage_CICD_Pipeline

Create a notification rule for this project

Developer Tools > CodeBuild > Build.projects > 3_Stage_CICD_Pipeline

3_Stage_CICD_Pipeline

Notify Share Edit Delete build project Start build with overrides Start build

Configuration

Source provider AWS CodeCommit	Primary repository 3_Stage_CICD_Pipeline	Artifacts upload location -	Build badge Enabled
Public builds Disabled	Copy badge URL		

Build history Batch history Build details Build triggers Metrics

Build history

Stop build View artifacts View logs Delete builds Retry build

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

The CodeBuild project has been built successfully!

- **Next, I created the BuildSpec**

I created a buildspec for the build stage of the pipeline “CodeBuild”. I created a BuildSpec YAML file which contains the command to test install cfn-lint and test a cloudformation template. In order to successfully achieve creating the BuildSpec devoid of errors, I referenced the AWS documentation <https://docs.aws.amazon.com/codebuild/latest/userguide/build-spec-ref.html>

```
version: 0.2
phases:
  install:
    on-failure: ABORT
    runtime-versions:
      python: 3.8 #installing python version 3.8
  pre_build:
    on-failure: CONTINUE
    commands:
      - echo ****Checking if python installed successfully****
      - python3 --version #will return the version of python that has been
installed
      - echo ***Checking if pip3 is installed***
      - pip3 --version
      - echo ***installing cfn-lint***
      - pip3 install cfn-lint
      - echo ***checking if cfn-lint is installed***
      - cfn-lint --version
  build:
    on-failure: ABORT
    commands:
      - echo ***Build started at `date` ***
      - pwd
      - cfn-lint launch_instance_in_custom_vpc.yml
  post_build:
    on-failure: ABORT
    commands:
      - echo ***Build complete***
      - ls -l
artifacts:
  files:
    - '**/*'
```

- Next, I created the CodePipeline

Step 1
Choose pipeline settings

Step 2
Add source stage

Step 3
Add build stage

Step 4
Add deploy stage

Step 5
Review

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

Step 2
Add source stage

Step 3
Add build stage

Step 4
Add deploy stage

Step 5
Review

Source

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

Repository name
Choose a repository that you have already created where you have pushed your source code.

Branch name
Choose a branch of the repository

Change detection options
Choose a detection mode to automatically start your pipeline when a change occurs in the source code.

Amazon CloudWatch Events (recommended)
Use Amazon CloudWatch Events to automatically start my pipeline when a change occurs

AWS CodePipeline
Use AWS CodePipeline to check periodically for changes

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.

The screenshot shows the 'Add build stage' configuration page. On the left, a sidebar lists steps: Step 1 (Choose pipeline settings), Step 2 (Add source stage), Step 3 (Add build stage - currently selected), Step 4 (Add deploy stage), and Step 5 (Review). The main area is titled 'Add build stage' and contains the following fields:

- Build - optional**: A dropdown for 'Build provider' set to 'AWS CodeBuild'. A note says: 'This is the tool of your build project. Provide build artifact details like operating system, build spec file, and output file names.'
- Region**: Set to 'US East (N. Virginia)'.
- Project name**: '3_Stage_CICD_Pipeline' with a 'Create project' button.
- Environment variables - optional**: A note: 'Choose the key, value, and type for your CodeBuild environment variables. In the value field, you can reference variables generated by CodePipeline.' A 'Learn more' link is provided.
- Add environment variable**: A button to add new variables.
- Build type**: Two options: 'Single build' (selected) and 'Batch build'.

The build stage failed because I have NOT yet pushed the buildspec to the codecommit repository!

The screenshot shows the AWS CodePipeline console for the pipeline '3_Stage_CICD_pipeline_install_python_application'. The left sidebar shows the pipeline structure:

- Source: CodeCommit
- Artifacts: CodeArtifact
- Build: CodeBuild
- Deploy: CodeDeploy
- Pipeline: CodePipeline
- Getting started
- Pipelines
- History
- Settings
- Settings
- Go to resource
- Feedback

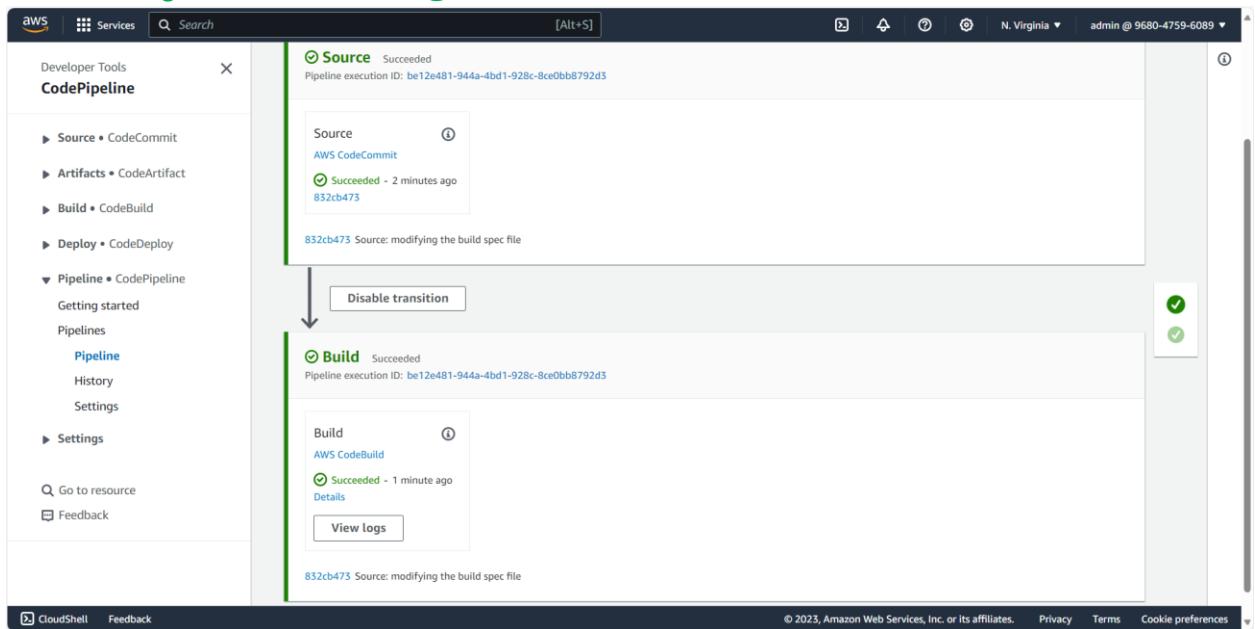
The main view shows the execution details:

- Source**: Succeeded. Pipeline execution ID: cd957cb1-3dbf-45e8-966c-e6387516d7a4. Status: Succeeded - 1 minute ago 95f5b68b.
- Build**: Failed. Pipeline execution ID: cd957cb1-3dbf-45e8-966c-e6387516d7a4. Status: Failed.

A red arrow points from the Source stage to the Build stage. A 'Disable transition' button is located between them. There are also 'Notify', 'Edit', 'Stop execution', 'Clone pipeline', and 'Release change' buttons at the top of the pipeline view.

Next, I did a git push from the local repo to the code commit repository! I pushed the buildspec file "buildspec.yml" and the cloudformation template "launch_instance_in_custom_vpc.yml" to the codecommit repo.

The Build Stage is now successful! 😊



- **The Cloudformation packaging for deployment**

In order to deploy the Cloudformation template, we need to install a cloudformation package during the build stage. Hence, we need to modify the buildspec file.

I referenced this aws documentation to achieve this

<https://docs.aws.amazon.com/cli/latest/reference/cloudformation/package.html>

In this stage, I first manually tested the command below in the linux instance to be sure it works before passing the command into the buildspec

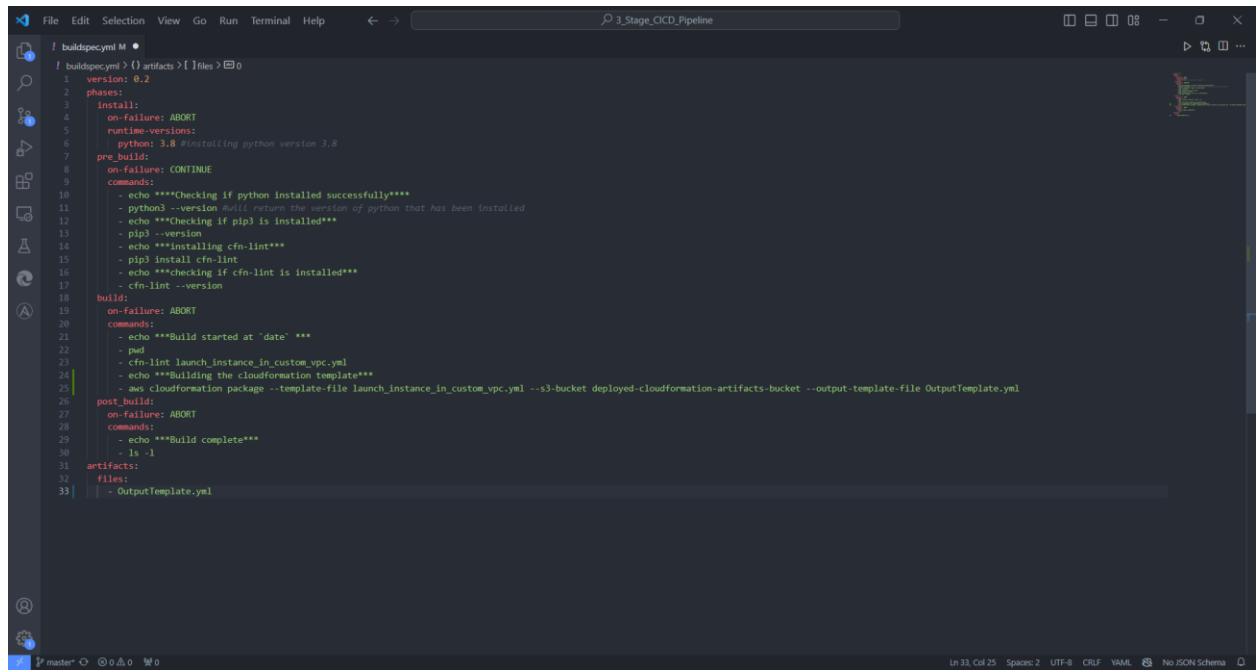
"aws cloudformation package --template-file launch_instance_in_custom_vpc.yml --s3-bucket deployed-cloudformation-artifacts-bucket --output-template-file OutputTemplate.yml"

This command will launch an s3 bucket where it will store the artifact ready to be deployed.

Session ID: admin-0762dd88ec34779f5 Instance ID: i-0039d6e8f66c89353 Terminate

```
[root@ip-10-0-1-31 bin]# aws cloudformation package --template-file launch_instance_in_custom_vpc.yaml --s3-bucket deployed-cloudformation-artifacts-bucket --output-template-file OutputTemplate.yml
Successfully packaged artifacts and wrote output template to file OutputTemplate.yml.
Execute the following command to deploy the packaged template
aws cloudformation deploy --template-file /usr/bin/OutputTemplate.yml --stack-name <YOUR STACK NAME>
[root@ip-10-0-1-31 bin]#
```

Since the command is successful, I then passed the command to the BuildSpec



```
! buildspec.yml M
1 buildspec.yml > () artifacts > [ ] files > □ o
2
3 phases:
4   install:
5     on-failure: ABORT
6     runtime-versions:
7       python: 3.8 #installing python version 3.8
8   pre_build:
9     on-failure: CONTINUE
10    commands:
11      - echo ***Checking if python installed successfully***#
12      - python3 --version #will return the version of python that has been installed
13      - echo ***Checking if pip3 is installed**
14      - pip3 -version
15      - echo ***installing cfn-lint***
16      - pip3 install cfn-lint
17      - echo ***checking if cfn-lint is installed***#
18      - cfn-lint --version
19   build:
20     on-failure: ABORT
21     commands:
22       - echo ***Build started at `date` ***
23       - pip3
24       - cfn-lint launch.instance_in_custom_vpc.yaml
25       - echo ***Building the cloudformation template***
26       - aws cloudformation package --template-file launch_instance_in_custom_vpc.yaml --s3-bucket deployed-cloudformation-artifacts-bucket --output-template-file OutputTemplate.yml
27   post_build:
28     on-failure: ABORT
29     commands:
30       - echo ***Build complete**#
31   artifacts:
32     files:
33       - OutputTemplate.yml
```

The screenshot shows a terminal window with the title "3_Stage_CI_CD_Pipeline". The command "aws cloudformation package" was run and completed successfully, producing the file "OutputTemplate.yml". This file is then included in the buildspec.yml as a template artifact.

- Adding the third stage of the pipeline- The Deploy stage

I created an IAM role using Cloudformation as the principal to enable cloudformation launch resources on my behalf.

The screenshot shows the AWS IAM Role creation interface. It consists of two main sections: Step 1: Select trusted entities and Step 2: Add permissions.

Step 1: Select trusted entities

Role details:

- Role name:** CloudFormation-service-role-for-pipeline
- Description:** Allows CloudFormation to create and manage AWS stacks and resources on your behalf.

Trust policy:

```

1 ~ [{
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Sid": "",
6       "Effect": "Allow",
7       "Principal": {
8         "Service": "cloudformation.amazonaws.com"
9       }
10      },
11    ],
12  ],
13 ]
  
```

Step 2: Add permissions

Permissions policy summary:

Policy name	Type	Attached as
AmazonEC2FullAccess	AWS managed	Permissions policy
AmazonSSMAutomationRole	AWS managed	Permissions policy
IAMFullAccess	AWS managed	Permissions policy

Next, I edited the Pipeline and added the deploy stage

The screenshot shows the AWS IAM service role configuration for the CloudFormation service role. The left sidebar lists various IAM management options like User groups, Roles, Policies, and Access reports. The main panel displays the role's summary, including its creation date (October 09, 2023, 04:14 UTC-02:30), last activity (none), ARN (arn:aws:iam::968047596089:role/CloudFormation-service-role-for-pipeline), and maximum session duration (1 hour). Below the summary is a 'Permissions' tab showing three attached policies: AmazonEC2FullAccess, AmazonSSMAutomationRole, and IAMFullAccess, all of which are AWS managed policies.

The screenshot shows the 'Edit action' configuration for a 'DeployStack' action within a pipeline. The left sidebar lists pipeline stages and actions. The main form includes fields for Action name (DeployStack), Action provider (AWS CloudFormation), Region (US East (N. Virginia)), Input artifacts (BuildArtifact), Action mode (Create or update a stack), Stack name (Deployed-using-CICD-Pipeline-Stack), and Template (specifying the template URL). A note at the bottom states: "When you update an existing stack, the update is permanent. When you use a change set, the result provides a diff of the updated stack and the original stack before you choose to execute the change."

Stack name
If you are updating an existing stack, choose the stack name.

Template
Specify the template you uploaded to your source location.
Artifact name File name Template file path

Template configuration - optional
Specify the configuration file you uploaded to your source location.
 Use configuration file

Artifact name File name Template configuration file path

Capabilities - optional
Specify whether you want to allow AWS CloudFormation to create IAM resources on your behalf.

Role name

Output file name

File generated by this action

Advanced

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](#)

Next, I passed the keypair name from the pipeline into the cloudformation template

Capabilities - optional
Specify whether you want to allow AWS CloudFormation to create IAM resources on your behalf.

Role name

Output file name

File generated by this action

Advanced

Parameter overrides

```
{
  "KeyName": "local-testing-keypair"
}

{
  "parameterName": "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](#)

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

No more than 100 characters

For this pipeline, I will be performing a “continuous Delivery” which requires approval before the resources are deployed. Hence, I setup another Action in the deploy stage to approve deployments by leveraging SNS.

The screenshot shows the AWS CodePipeline console. On the left, there's a sidebar with navigation links like 'Source', 'Artifacts', 'Build', 'Deploy', 'Pipeline', 'Getting started', 'Pipelines', 'Pipeline', 'History', 'Settings', 'Feedback', and 'Go to resource'. The main area displays a pipeline stage named 'Deploy' with a status of 'In progress'. A modal window titled 'Review' is open over the pipeline stage, containing the message 'Please approve my pipeline'. It also includes a 'Comments - optional' text area, a 'Preview markdown' toggle, and 'Learn more' link. At the bottom of the modal are 'Cancel', 'Reject', and 'Approve' buttons.

The screenshot shows an iPhone screen with a text message from '+1 (562) 265-3590'. The message reads: 'Text Message Today 4:36 AM This is your One Time Password: 641868'. Below this, four identical messages are shown, each reading: 'CICD-Cloudformation-Approval> APPROVAL NEEDED: AWS CodePipeline 3_Stage_CICD_pipeline_install_python_ap... for action Continuous-Delivery-Approval'. At the bottom of the text message screen, it says 'The sender is not in your contact list.' and 'Report Junk'.

Next, I set up the deploy to production stage

NB: -This is only available in **US-WEST-1** region

- Since Keyname is regional, you need to select a key in the US-WEST-1 region
- You can use the same Role as you used in the deploy-to-test stage

The screenshot shows the 'Edit action' configuration page for the 'Deploy-to-Prod' step in a pipeline. The left sidebar lists various stages and actions. The main form is titled 'Edit action' and contains the following fields:

- Action name:** Deploy-to-Prod
- Action provider:** AWS CloudFormation
- Region:** US West (N. California)
- Input artifacts:** BuildArtifact
- Action mode:** Create or update a stack
- Stack name:** deployed-using-CI_CD-stack

Below this, there is another configuration section for the 'Template' field:

- Artifact name:** BuildArtifact
- File name:** OutputTemplate.yml
- Template file path:** BuildArtifact:OutputTemplate.yml

Further down, there are sections for 'Template configuration' and 'Capabilities'.

At the bottom, the 'Role name' is specified as arn:aws:iam::968047596089:role/CloudFormation-service-role-for-pipeline.

Specify whether you want to allow AWS CloudFormation to create IAM resources on your behalf.

CAPABILITY_IAM

Role name: arn:aws:iam::968047596089:role/CloudFormation-service-role-for-pipeline

Output file name:

File generated by this action:

Advanced

Parameter overrides: {"KeyName": "cicd-production-key"}

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](#)

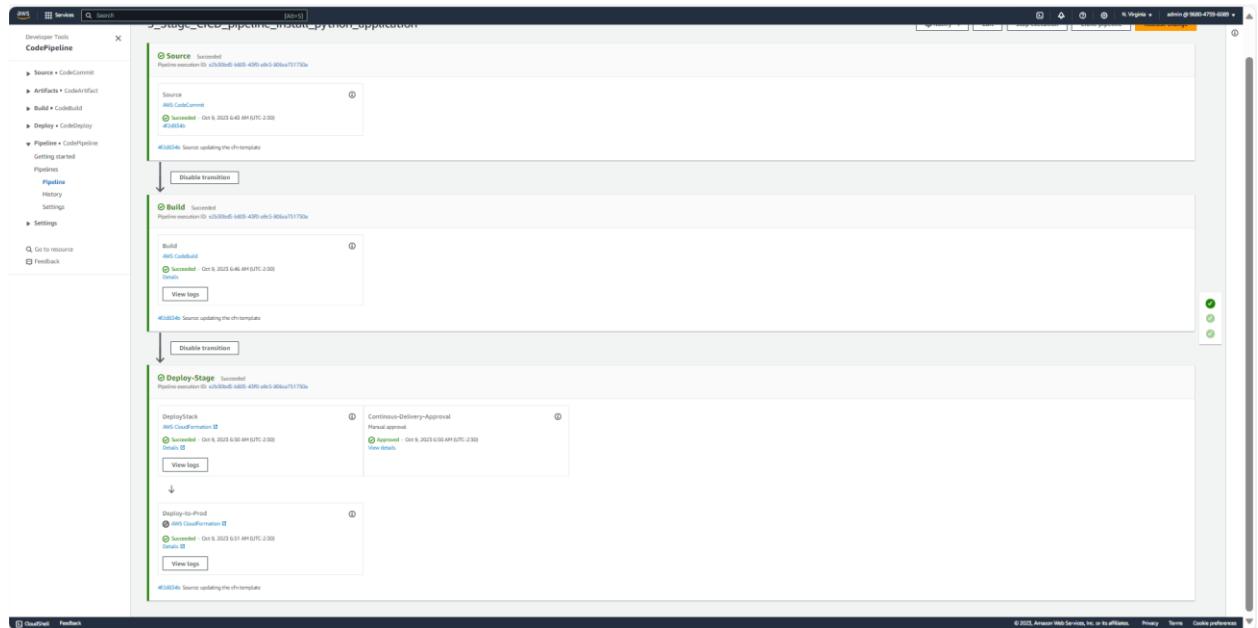
Output artifacts: Choose a name for the output of this action. No more than 100 characters.

Done

After everything is set and I approved the delivery, the pipeline moved to the Production stage as seen below;

The screenshot shows the AWS CodePipeline console with the following details:

- Left Sidebar:** Developer Tools, Services, Search, Pipeline, History, Settings, Go to resource, Feedback.
- Middle Area:**
 - Deploy Stage:** Succeeded, Pipeline execution ID: e2b30bd5-b805-43f0-a9c5-806ca751750a. It includes a DeployStack step (AWS CloudFormation) and a Continuous-Delivery... step (Manual approval). The DeployStack step is successful, and the Continuous-Delivery... step is approved.
 - Downward Arrow:** Indicates the flow to the next stage.
 - Deploy-to-Prod Stage:** Starting, AWS CloudFormation. It shows a Succeeded step (Oct 9, 2023 6:51 AM UTC-2:30) and a Details link.
- Bottom Status Bar:** 4f2d854b Source: updating the cfn template, © 2023, Amazon Web Services, Inc. or its affiliates., Privacy, Terms, Cookie preferences.



Pipeline was deployed successfully!

Results:

The instance launched in us-east-1 (Testing)

The screenshot shows the AWS EC2 Instances page with the URL "https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#instances;v=3;\$case=tags:true%5C;client:false;\$regex=tags:false%5C;client:false". The page displays a table of instances. One instance is listed: "My3StageCICDInstance" with Instance ID "i-01dfebbd5124a71cc", State "Running", Type "t2.micro", and Availability Zone "us-east-1a". The "Actions" column for this instance has a "Launch instances" button.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
My3StageCICDInstance	i-01dfebbd5124a71cc	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a

Instance: i-01dfebbd5124a71cc (My3StageCICDInstance)

Private IP: 172.31.1.112	Public IP: 18.209.255.165 [open address]	Private IP DNS name (IPv4 only): ip-10-0-1-112.ec2.internal
IPv6 address: -	Instance state: Running	Instance type: t2.micro
Hostname type: IP name: ip-10-0-1-112.ec2.internal	Private IP: 10.0.1.112	Elastic IP addresses: 18.209.255.165 [Public IP]
Answer private resource DNS name: -	Public IP: ec2-18-209-255-165.compute-1.amazonaws.com [open address]	AWS Compute Optimizer finding: Opt-in to AWS Compute Optimizer for recommendation
Auto-assigned IP address: -		

The static website in us-east-1

You successfully accessed Nnadiekwe Chiderah David web page launched via a CloudFormation template in US-EAST-1 region

The instance launched in US-WEST-1 (production)

Instances (1/1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Pub
My3StageCI_CD...	i-03023bf2c248755b5	Running	t2.micro	2/2 checks passed	No alarms	us-west-1a	ec2-13-57-109-189.us...	13.5

Instance: i-03023bf2c248755b5 (My3StageCI_CDInstance)

Details | Security | Networking | Storage | Status checks | Monitoring | Tags

Instance summary

Instance ID i-03023bf2c248755b5 (My3StageCI_CDInstance)	Public IPv4 address 13.57.109.189 [open address]
IPv6 address -	Instance state Running
Hostname type IP name: ip-10-0-1-16.us-west-1.compute.internal	Private IP DNS name (IPv4 only) ip-10-0-1-16.us-west-1.compute.internal
Answer private resource DNS name -	Instance type t2.micro
Auto-assigned IP address -	VPC ID vpc-0ffe6a7629d19413a (3StageCI_CDVPC) [

Private IPv4 addresses
10.0.1.16

Public IPv4 DNS
ec2-13-57-109-189.us-west-1.compute.amazonaws.com [open address]

Elastic IP addresses
13.57.109.189 [Public IP]

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

The static webpage in US-WEST-1

3 STAGE CICD PIPELINE WAS USED TO DEPLOY THIS WEBPAGE

You successfully accessed Nnadiekwe Chiderah David web page launched via a CloudFormation template in US-WEST-1 region

The deployed stack

The screenshot shows the AWS CloudFormation console interface. On the left, the navigation pane includes 'CloudFormation', 'Stacks' (selected), 'Drifts', 'StackSets', 'Exports', 'Designer', 'Registry' (with 'Public extensions', 'Activated extensions', and 'Publisher' listed), 'Spotlight', and 'Feedback'. The main content area displays the 'Stacks' list for the selected stack, which is named 'Deployed-using-CICD-pipeline-stack' and was created on '2023-10-09 06:49:31 UTC-0230'. The status is 'UPDATE_COMPLETE'. To the right, the 'Resources (11)' section lists the following resources:

Logical ID	Physical ID	Type	Status
InternetGateway	igw-0a397d742c4de9746	AWS::EC2::InternetGateway	CREATE_COMPLETE
My3StageCICDInstance	it-01dfebbd5124a71cc	AWS::EC2::Instance	CREATE_COMPLETE
myEIP	18.209.255.165	AWS::EC2::EIP	CREATE_COMPLETE
PublicRoute	rtb-0733a986742fd6818 0.0/0	AWS::EC2::Route	CREATE_COMPLETE
PublicRouteTable	rtb-0733a986742fd6818	AWS::EC2::RouteTable	CREATE_COMPLETE
PublicSubnet	subnet-0aa83ad5687bcae22	AWS::EC2::Subnet	CREATE_COMPLETE
PublicSubnetNetworkAclAssociation	aclassoc-0c74fcf736d9fcc02	AWS::EC2::SubnetNetworkAclAssociation	CREATE_COMPLETE
PublicSubnetRouteTableAssociation	rtbassoc-0696135f5f53db488	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE