Activity - Troubleshoot CloudFormation

Activity overview

In this activity, you will practice troubleshooting AWS CloudFormation deployments.

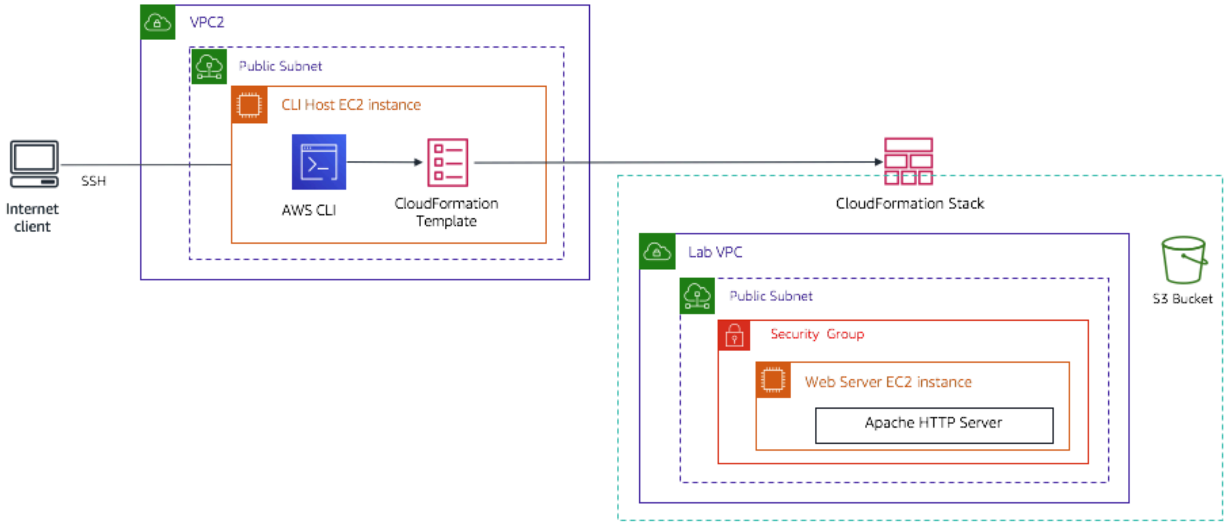
In **Task 1**, you practice querying JavaScript Object Notation (JSON)-formatted data by using JMESPath.

In **Task 2**, you start using your AWS account. The environment starts by providing you an Amazon Elastic Compute Cloud (Amazon EC2) instance named *CLI Host* that exists in a virtual private cloud (VPC) named *VPC2*. You will establish a Secure Shell (SSH) connection to the CLI Host. From the CLI Host, you use the AWS Command Line Interface (AWS CLI) to run AWS CloudFormation commands that will create a stack with resources in the AWS account. Your first attempt to create a stack will fail and will require troubleshooting.

In **Task 3**, you manually modify (outside of the context of AWS CloudFormation) a resource that was created by the AWS CloudFormation stack. You then use AWS CloudFormation to detect drift.

In **Task 4**, you attempt to delete the stack. However, issues are encountered. In the **Challenge** section, you must figure out to how successfully delete the stack, while keeping the Amazon Simple Storage Service (Amazon S3) bucket that was originally created by the stack and that now contains objects.

The architectural diagram below illustrates the setup that is used in tasks 2, 3, and 4 of this activity.



Activity objectives

After completing this activity, you will be able to:

- **Practice** using JMESPath to query JSON-formatted documents.
- **Troubleshoot** the deployment of an AWS CloudFormation stack by using the AWS CLI.
- **Analyze** log files on a Linux EC2 instance to determine the cause of a `create-stack` failure.
- **Troubleshoot** a failed `delete-stack` action.

Business case relevance

A new request from the Café leadership team

⚠

Sofîa is discussing the AWS deployment that is used by the Café with Olivia. Sofîa mentioned that Martha and Frank would like both her and Nikhil to have the skills to build infrastructure as code (IaC). Olivia suggested that as a first attempt, Sofîa should try working on a proof of concept (POC).

In this activity, you take on the role of Sofîa. See if you can create a deployment of a web server inside a custom VPC that you define, and perform this deployment by using an AWS CloudFormation template. You will also learn how to troubleshoot deployments so that you gain an understanding of how effective this technology can be.

Task 1: Practice querying JSON-formatted data by using JMESPath

In this first task, you will practice using the JMESPath JSON query language to return results from a JSON document.

3. Open a new browser window and go to jmespath.org/.
   Tip: If possible, arrange the browser windows so that you can see these instructions and also see the jmespath.org window at the same time.
4. On the JMESPath website, in the document window that currently displays the *locations* JSON document, copy the following JSON document (replacing the *locations* document):
5.
   ```
   {
     "desserts": [
       {
         "name": "Chocolate cake",
         "price": "20.00"
       },
       {
         "name": "Ice cream",
         "price": "15.00"
       },
       {
         "name": "Carrot cake",
         "price": "22.00"
       }
   ```

```
   ]
}
```

6. Resize the JSON code panel in the browser window so that you can see the entire document, or as much of it as possible.
7. In the **Expression** search box above the document, delete all the text. In the search box, enter `desserts`.
   The expression is immediately evaluated.
   Important: Do not press ENTER after you enter an expression in the search box. If you press ENTER, the entire page reloads with its original data.
   In the **Result** panel below the document, notice that all the content that is in the *desserts* part of the document is returned.
8. Add `[1]` to the expression:
9. `desserts[1]`
10. Notice that only the second dessert element is displayed. The `[]` notation is used with an index, and it enables you to refer to a specific element of an array. Because JMESPath considers the first position in an array to be 0, `1` returns the second element.
11. Retrieve only the value of the *name* attribute for the chocolate cake element. In the search box, enter:
12. `desserts[0].name`
13. The name of the first dessert element, *"Chocolate cake"*, is returned.
    Tip: The `.` notation allows you to specify the name of an attribute in the document.
14. Retrieve the values of both the *name* and *price* attributes of the chocolate cake element. In the search box, enter:
15. `desserts[0].[name,price]`
16. The name and price of the chocolate cake desert is displayed. The `[]` notation can also be used to define a list of attributes to return.
17. Return the values of the *name* attribute for all three dessert elements, but without the prices.
18. `desserts[].name`
19. An empty array index `[]` or one with an asterisk `[*]` refers to all of the elements in an array.
20. Now use a filter, instead of referring to elements by their position. Return the attributes of the carrot cake element:
21. `desserts[?name=='Carrot cake']`
22. Notice that you did not need to know the index of the element that you were searching for. The filter expression `[? <expression>]` returns the elements in the document that match the expression condition that you specify.

23. Lastly, replace the JSON document with the following document, which describes resources in an AWS CloudFormation stack:
24.
    ```
    {
        "StackResources": [
            {
                "LogicalResourceId": "VPC",
                "ResourceType": "AWS::EC2::VPC"
            },
           {
                "LogicalResourceId": "PublicSubnet1",
                "ResourceType": "AWS::EC2::Subnet"
            },
           {
                "LogicalResourceId": "CliHostInstance",
                "ResourceType": "AWS::EC2::Instance"
            }
        ]
    }
    ```
25. Can you determine the correct JSMESPath expression to retrieve the *LogicalResourceId* of the *EC2 instance* resource? Try to figure it out on your own before you use the following solution.
26.
    ```
    StackResources[?ResourceType == 'AWS::EC2::Instance'].LogicalResourceId
    ```
27. When you complete the other tasks in this activity, you will notice that the instructions often have AWS CLI commands that include `--query` or `--filter` parameters. These parameters use JMSEPath expressions to filter the output that is returned by an AWS CLI command.
    You are encouraged to practice using what you learned in this first task—throughout this activity—while you use the AWS CLI.

Task 2: Troubleshooting and working with AWS CloudFormation stacks

Task 2 starts with an EC2 instance named *CLI Host*, which is already created for you. It runs in the public subnet of a VPC named *VPC2*.

You will first establish an SSH connection to the CLI Host so that you can work with the AWS CloudFormation service from there.

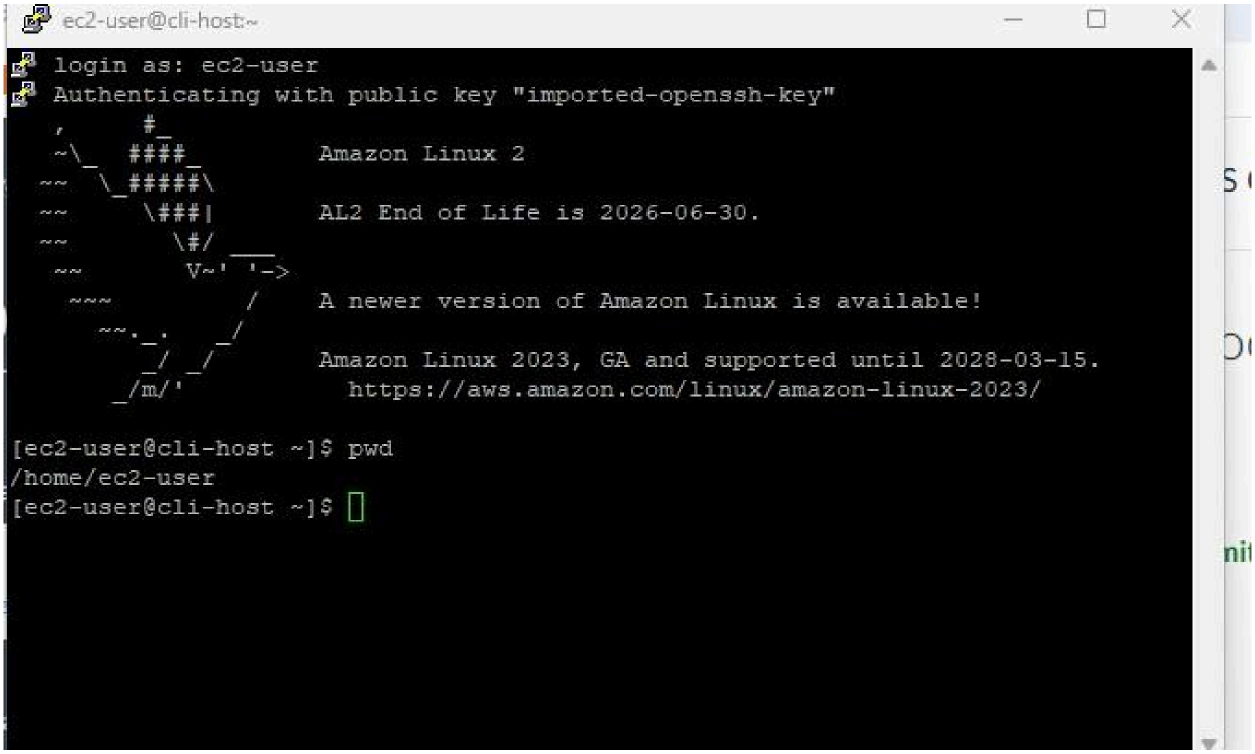
Task 2.1 for Windows: SSH to CLI Host Instance

These instructions are specifically for Windows users. If you are using macOS or Linux, skip to the next section.

14. Select the `Details` drop-down menu above these instructions you are currently reading, and then select `Show`. A Credentials window will be presented.

Select the **Download PPK** button and save the **labsuser.ppk** file.

15. *Typically your browser will save it to the Downloads directory.*
16. Make a note of the **PublicIP** address.
17. Then exit the Details panel by selecting the **X**.
18. Download **PuTTY** to SSH into the Amazon EC2 instance. If you do not have PuTTY installed on your computer, download it here.
19. Open **putty.exe**
20. Configure your PuTTY session by following the directions in the following link: Connect to your Linux instance using PuTTY
21. Windows Users: Select here to skip ahead to the next task.

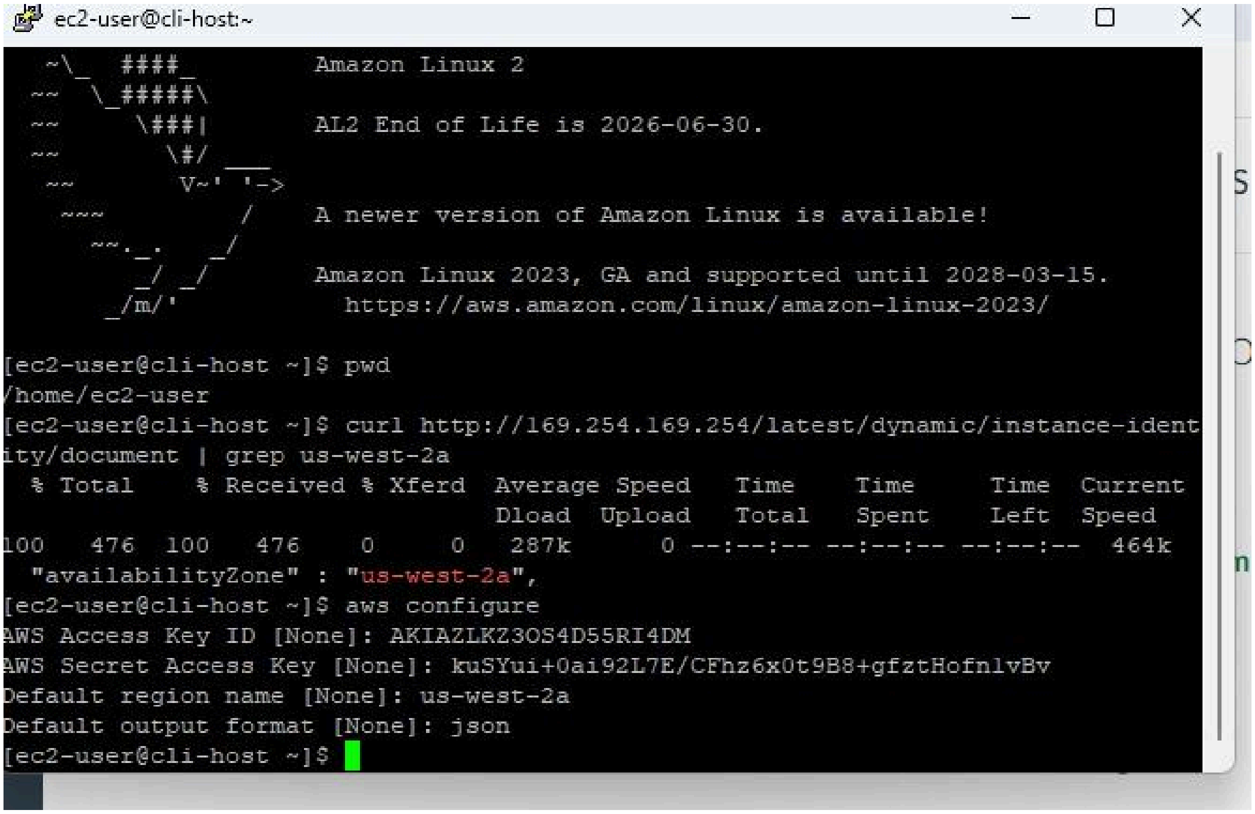


```
login as: ec2-user
Authenticating with public key "imported-openssh-key"
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@cli-host ~]$ pwd
/home/ec2-user
[ec2-user@cli-host ~]$ 
```

Task 2.2: Configure the AWS CLI

29. Discover the region in which the CLI Host instance is running:
30. `curl http://169.254.169.254/latest/dynamic/instance-identity/document | grep region`
31. You will use this region information in a moment.
32. Update the AWS CLI software with the credentials.
33. `aws configure`
34. At the prompts, enter the following information:
    - **AWS Access Key ID**: Click on the Details drop down menu above these instructions, and then click Show. Copy the **AccessKey** value and paste it into the terminal window.
    - **AWS Secret Access Key**: Copy and paste the **SecretKey** value from the same Credentials screen.
    - **Default region name**: Type in the name of the region where your EC2 instances are running, which you just discovered a moment ago. For example, `us-east-1` or `eu-west-2`.
    - **Default output format**: `json`



```
  ~\_  ####_        Amazon Linux 2
 ~~  \_#####\
 ~~     \###|       AL2 End of Life is 2026-06-30.
 ~~       \#/ ___
  ~~       V~' '->
   ~~~         /    A newer version of Amazon Linux is available!
     ~~._.   _/
        _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
      _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@cli-host ~]$ pwd
/home/ec2-user
[ec2-user@cli-host ~]$ curl http://169.254.169.254/latest/dynamic/instance-ident
ity/document | grep us-west-2a
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   476  100   476    0     0   287k      0 --:--:-- --:--:-- --:--:--  464k
  "availabilityZone" : "us-west-2a",
[ec2-user@cli-host ~]$ aws configure
AWS Access Key ID [None]: AKIAZLKZ3OS4D55RI4DM
AWS Secret Access Key [None]: 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
Default region name [None]: us-west-2a
Default output format [None]: json
[ec2-user@cli-host ~]$ 
```

Task 2.3: Attempt to create an AWS CloudFormation stack

In this task, you will try to create an AWS CloudFormation stack from a template that is given to you. You will use the AWS CLI to do this task.

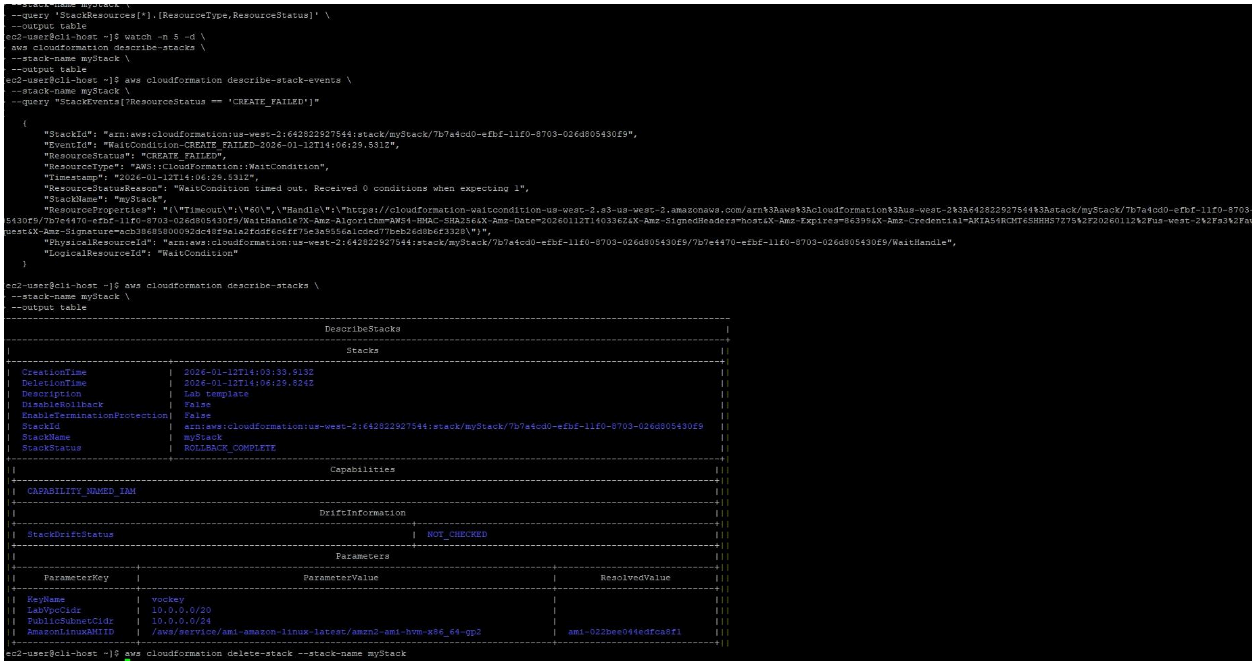
32. Run the following command to first observe the AWS CloudFormation template that you will use:
33. `less template1.yaml`
34. Scroll through the template contents and observe the contents:
    Tip: Press RETURN (or ENTER) to scroll though the contents of the file.
    - The **Parameters** section has a few items, including the *AmazonLinuxAMIID* and *KeyName* parameters that are referenced in the **Resources** section. The **Resources** section is lower in the template, and it is where you can create an EC2 instance.
    - The **Resources** section contains most of the lines in this template. It creates the following resources:
        - **VPC** with supporting resources, including a public subnet.

        - **EC2 instance** named *Web Server*.
        - **WaitCondition** and **WaitHandle** resources that wait for the userdata script (which is specified in the EC2 instance details) to complete.
        - **S3 bucket**.
        - **Security Group** resource named *WebServerSG*.
    - The **Outputs** section writes out the *name of the S3 bucket* and the *public IP address of the EC2 instance* that are created.

35. When you are done observing the contents of the template, exit the less Linux utility by entering `q`.
36. Run the following command to create a stack:
37.
```
aws cloudformation create-stack \
--stack-name myStack \
--template-body file://template1.yaml \
--capabilities CAPABILITY_NAMED_IAM \
--parameters ParameterKey=KeyName,ParameterValue=vockey
```
38. Check the status of each resource that is created by this stack:
39.
```
watch -n 5 -d \
aws cloudformation describe-stack-resources \
--stack-name myStack \
--query 'StackResources[*].[ResourceType,ResourceStatus]' \
--output table
```
40. **Note:** In the previous command, the *watch* Linux utility is used to invoke the `describe-stack-resources` command. It runs the same command every 5 seconds, and it briefly highlights changes as they occur. The command above also uses the `--output table` parameter to make reading the results easier.
41. Observe the progress of resource creation. This step will take approximately 3–5 minutes.
    You will not see all resources display immediately because some resources have dependencies on other resources, which must be created first.
    The WaitCondition resource is the last resource that will be run. It has a 60 second timeout from when it was invoked, which does not occur until after some other resources are created.
    Important: Notice that after *almost* all resources are created, they start being deleted.
    Something is definitely wrong. You must do some troubleshooting.
42. To exit the watch Linux utility, press CTRL+C on your keyboard and go to the next step.
43. To see the stack status and other details, run the `describe-stacks` command:
44.
```
watch -n 5 -d \
aws cloudformation describe-stacks \
--stack-name myStack \
--output table
```

45. Depending on when you ran the previous command, the output of `describe-stacks` will either show a status of *CREATE_FAILED* or it will enter a status of *ROLLBACK_IN_PROGRESS* followed by a status of *ROLLBACK_COMPLETE*.
    This is expected behavior.
46. Exit the watch Linux utility by pressing CTRL+C.
39. Analyze the issue by running the `describe-stack-events` command with a query that returns only the *CREATE_FAILED* events:
40.
    ```
    aws cloudformation describe-stack-events \
    --stack-name myStack \
    --query "StackEvents[?ResourceStatus == 'CREATE_FAILED']"
    ```
41. Read the output of the command.
    If you followed the steps above, the output of this command—in the *ResourceStatusReason* element—should show that the WaitCondition timed out. You might recall—from looking at the AWS CloudFormation template—that the wait condition waits for a signal at the end of the **userdata** section of the EC2 instance resource. If you would like to confirm this situation, review the contents of the template again.
    This situation could indicate that there is an issue in the **userdata** section of the EC2 instance resource that the template attempted to create.
42. Consider accessing the userdata logs for the EC2 instance.
    The next logical step is to look at the logs that capture any errors that were thrown when the userdata script ran. Unfortunately, AWS CloudFormation deleted the EC2 instance that contained the log.
    By default, AWS CloudFormation deletes all resources if *any* of the resources that are defined in the template cannot be successfully created. Because the wait condition resource failed, the entire stack failed and all changes were rolled back.
43. Run the `describe-stacks` command one more time.
44.
    ```
    aws cloudformation describe-stacks \
    --stack-name myStack \
    --output table
    ```
45. The StackStatus element indicates that the status of the stack is now *ROLLBACK_COMPLETE*.
    This status confirms that the EC2 instance and the other resources that were created by the stack were deleted.
46. Delete the stack with the status *ROLLBACK_COMPLETE*.
    Though the stack failed, and the stack resources were all deleted, the stack object itself was not deleted.
    Use the following command to delete the stack:

47. `aws cloudformation delete-stack --stack-name myStack`
48. The stack will be deleted quickly because it does not contain resources that must be rolled back.



```
> --stack-name myStack \
> --query 'StackResources[*].[ResourceType,ResourceStatus]' \
> --output table
[ec2-user@cli-host ~]$ watch -n 5 -d \
> aws cloudformation describe-stacks \
> --stack-name myStack \
> --output table
[ec2-user@cli-host ~]$ aws cloudformation describe-stack-events \
> --stack-name myStack \
> --query "StackEvents[?ResourceStatus == 'CREATE_FAILED']"
[
    {
        "StackId": "arn:aws:cloudformation:us-west-2:642822927544:stack/myStack/7b7a4cd0-efbf-11f0-8703-026d805430f9",
        "EventId": "WaitCondition-CREATE_FAILED-2026-01-12T14:06:29.531Z",
        "ResourceStatus": "CREATE_FAILED",
        "ResourceType": "AWS::CloudFormation::WaitCondition",
        "Timestamp": "2026-01-12T14:06:29.531Z",
        "ResourceStatusReason": "WaitCondition timed out. Received 0 conditions when expecting 1",
        "StackName": "myStack",
        "ResourceProperties": "{\"Timeout\":\"60\",\"Handle\":\"https://cloudformation-waitcondition-us-west-2.s3-us-west-2.amazonaws.com/arn%3Aaws%3Acloudformation%3Aus-west-2%3A642822927544%3Astack/myStack/7b7a4cd0-efbf-11f0-8703-
05430f9/7b7e4470-efbf-11f0-8703-026d805430f9/WaitHandle?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=20260112T140336Z&X-Amz-SignedHeaders=host&X-Amz-Expires=86399&X-Amz-Credential=AKIA54RCMT6SHHHS7Z75%2F20260112%2Fus-west-2%2Fs3%2Faw
quest&X-Amz-Signature=acb38685800092dc48f9a1a2fddf6c6ff75e3a9556a1cded77beb26d8b6f3328\"}",
        "PhysicalResourceId": "arn:aws:cloudformation:us-west-2:642822927544:stack/myStack/7b7a4cd0-efbf-11f0-8703-026d805430f9/7b7e4470-efbf-11f0-8703-026d805430f9/WaitHandle",
        "LogicalResourceId": "WaitCondition"
    }

[ec2-user@cli-host ~]$ aws cloudformation describe-stacks \
> --stack-name myStack \
> --output table
-------------------------------------------------------------------------------------------------------------------------------------
|                                                          DescribeStacks                                                           |
+-----------------------------------------------------------------------------------------------------------------------------------+
|                                                              Stacks                                                              ||
+-----------------------------+----------------------------------------------------------------------------------------------------+|
|  CreationTime               |  2026-01-12T14:03:33.913Z                                                                          ||
|  DeletionTime               |  2026-01-12T14:06:29.824Z                                                                          ||
|  Description                |  Lab template                                                                                      ||
|  DisableRollback            |  False                                                                                             ||
|  EnableTerminationProtection|  False                                                                                             ||
|  StackId                    |  arn:aws:cloudformation:us-west-2:642822927544:stack/myStack/7b7a4cd0-efbf-11f0-8703-026d805430f9  ||
|  StackName                  |  myStack                                                                                           ||
|  StackStatus                |  ROLLBACK_COMPLETE                                                                                 ||
+-----------------------------+----------------------------------------------------------------------------------------------------+|
||                                                          Capabilities                                                          |||
|+--------------------------------------------------------------------------------------------------------------------------------+||
||  CAPABILITY_NAMED_IAM                                                                                                          |||
|+--------------------------------------------------------------------------------------------------------------------------------+||
||                                                        DriftInformation                                                        |||
|+-------------------------------------------------------------------------+------------------------------------------------------+||
||  StackDriftStatus                                                       |  NOT_CHECKED                                         |||
|+-------------------------------------------------------------------------+------------------------------------------------------+||
||                                                           Parameters                                                           |||
|+----------------------+---------------------------------------------------------------------------+-----------------------------+||
||     ParameterKey     |                              ParameterValue                               |        ResolvedValue        |||
|+----------------------+---------------------------------------------------------------------------+-----------------------------+||
||  KeyName             | vockey                                                                    |                             |||
||  LabVpcCidr          | 10.0.0.0/20                                                               |                             |||
||  PublicSubnetCidr    | 10.0.0.0/24                                                               |                             |||
||  AmazonLinuxAMIID    | /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2             | ami-022bee044edfca8f1       |||
|+----------------------+---------------------------------------------------------------------------+-----------------------------+||
[ec2-user@cli-host ~]$ aws cloudformation delete-stack --stack-name myStack
```

Task 2.4: Avoid rollback on an AWS CloudFormation stack

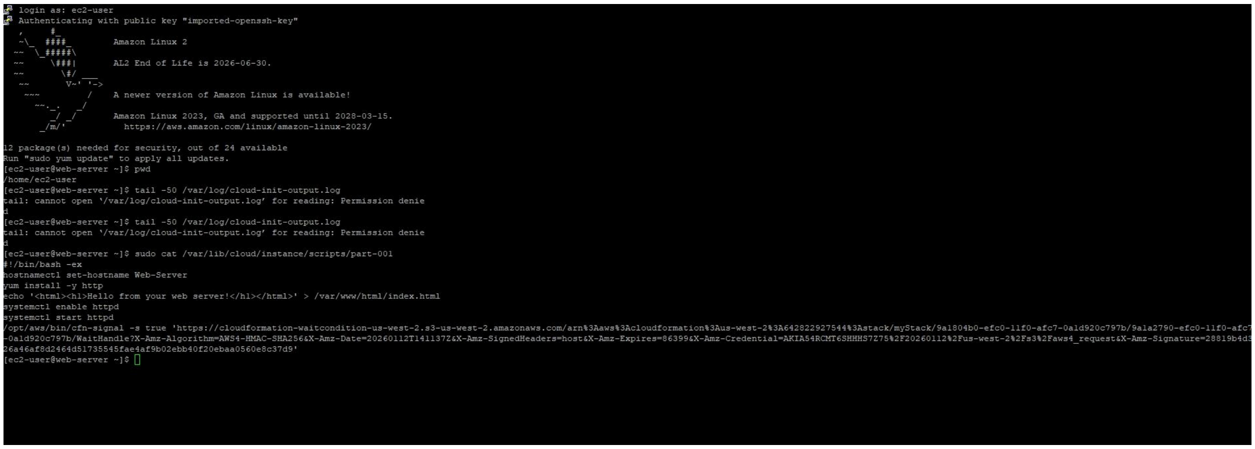
44. Now, run the `create-stack` command again. Give the stack the same name, but this time, specify that there should be no rollback on failure:
45.
    ```
    aws cloudformation create-stack \
    --stack-name myStack \
    --template-body file://template1.yaml \
    --capabilities CAPABILITY_NAMED_IAM \
    --on-failure DO_NOTHING \
    --parameters ParameterKey=KeyName,ParameterValue=vockey
    ```
46. In the command, notice the parameter `--on-failure DO_NOTHING`. This parameter will ensure that a rollback will not occur this time. Without a rollback, you will have time to introspect the EC2 instance logs.
47. Run the `describe-stack-resources` command again.

48.
   ```
   watch -n 5 -d \
   aws cloudformation describe-stack-resources \
   --stack-name myStack \
   --query 'StackResources[*].[ResourceType,ResourceStatus]' \
   --output table
   ```
49. Wait until there are no more stack resources with a status of *CREATE_IN_PROGRESS*.
   Notice that after the WaitCondition attains the status of *CREATE_FAILED*, the other resources keep their *CREATE_COMPLETE* status.
   Tip: To exit the watch utility, press CTRL+C.
50. Run the `describe-stacks` command.
51.
   ```
   aws cloudformation describe-stacks \
   --stack-name myStack \
   --output table
   ```
52. The output should indicate that the status of the stack is now *CREATE_FAILED*. Significantly, however, AWS CloudFormation did not roll back the stack this time.
53. Analyze the latest details of the *CREATE_FAILED* event and verify that it is the same issue as before.
54.
   ```
   aws cloudformation describe-stack-events \
   --stack-name myStack \
   --query "StackEvents[?ResourceStatus == 'CREATE_FAILED']"
   ```
55. The output of the above command should again confirm that the timeout of the WaitCondition is the issue.
56. After the stack fails, use SSH to connect to the *Web Server* EC2 instance that was created by the stack. To do this:
   - In the terminal window where you are connected to the CLI Host, run a `describe-instances` command to get the public IP address:
   -
     ```
     aws ec2 describe-instances \
     --filters "Name=tag:Name,Values='Web Server'" \
     --query
     'Reservations[].Instances[].[State.Name,PublicIpAddress]'
     ```
   - Copy the IP address that was returned by the previous command.
   - Leave this CLI Host terminal window active. You will return to it later.
   - Open a new terminal window or tab on your computer.
     Tip: For details on how to connect to the instance, refer back to Task 2.1 for your operating system. This time, make sure that you connect to the new IP address. After you are connected, the terminal prompt should contain the hostname of *web-server*.
   - Connect to the Web Server instance.

49. Analyze the cloud-init-output.log file.

    - In the **web-server** SSH terminal window, run the following command to view the last 50 lines of the cloud-init log output:
    - `tail -50 /var/log/cloud-init-output.log`
    - Notice the line in the log (approximately 5–10 lines from the bottom) that states: *No package http available*.
    - Also, notice the message: *util.py[WARNING]: Failed running /var/lib/cloud/instance/scripts/part-001*. You will look at the util.py file next.

50. Analyze the part-001 script.
    - Run the following command to view the part-001 script:
    - `sudo cat /var/lib/cloud/instance/scripts/part-001`
    - Notice that the part-001 script contains all the contents of *userdata* that were specified in the AWS CloudFormation template.
    - In particular, notice the first line, which starts with the #! characters. This line includes the `-e` parameter. This parameter signals that if any command in the script fails, the whole script should immediately stop running with a non-zero status.
    - In summary, because no package named *http* could be found, the userdata script failed. Therefore, the wait condition never received the success signal, and after 2 minutes, the wait condition timed out. This reason is why the stack failed.
51. In the terminal window where you are connected to the web server, enter `exit` to disconnect from the Web Server instance.
52. Close the terminal window.



```
login as: ec2-user
Authenticating with public key "imported-openssh-key"
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

12 package(s) needed for security, out of 24 available
Run "sudo yum update" to apply all updates.
[ec2-user@web-server ~]$ pwd
/home/ec2-user
[ec2-user@web-server ~]$ tail -50 /var/log/cloud-init-output.log
tail: cannot open '/var/log/cloud-init-output.log' for reading: Permission denie
d
[ec2-user@web-server ~]$ tail -50 /var/log/cloud-init-output.log
tail: cannot open '/var/log/cloud-init-output.log' for reading: Permission denie
d
[ec2-user@web-server ~]$ sudo cat /var/lib/cloud/instance/scripts/part-001
#!/bin/bash -ex
hostnamectl set-hostname Web-Server
yum install -y http
echo '<html><h1>Hello from your web server!</h1></html>' > /var/www/html/index.html
systemctl enable httpd
systemctl start httpd
/opt/aws/bin/cfn-signal -s true 'https://cloudformation-waitcondition-us-west-2.s3-us-west-2.amazonaws.com/arn%3Aaws%3Acloudformation%3Aus-west-2%3A642822927544%3Astack/myStack/9a1804b0-efc0-11f0-afc7-0a1d920c797b/9a1a2790-efc0-11f0-afc7
-0a1d920c797b/WaitHandle?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=20260112T141137Z&X-Amz-SignedHeaders=host&X-Amz-Expires=86399&X-Amz-Credential=AKIA54RCMT6SHHHS7Z75%2F20260112%2Fus-west-2%2Fs3%2Faws4_request&X-Amz-Signature=28819b4d3
26a46af8d2464d51735545fae4af9b02ebb40f20ebaa0560e8c37d9'
[ec2-user@web-server ~]$ 
```

Task 2.5: Fix the issue and successfully create the AWS CloudFormation stack

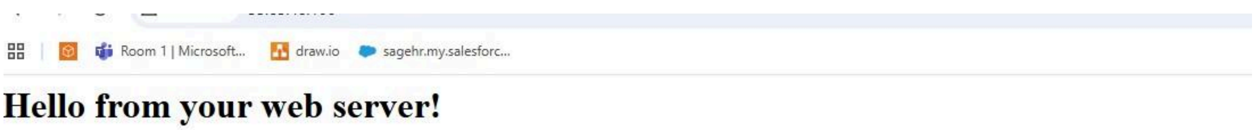
53. Back in the terminal window where you are connected to the CLI Host instance, update the AWS CloudFormation template:
54. `vim template1.yaml`
    - Use the DOWN arrow key to scroll down (or enter `:128` and press ENTER) to scroll to the part of the template that must be updated.
    - On line 128, in the **UserData section** of the EC2 resource, change *"http"* to `"httpd"` (which is the actual name of the Apache web server).
    - **Tip:** Use the UP and DOWN arrow keys to position your cursor. Enter `a` to enter edit mode in the vi editor, and then make the change. Press ESC to exit edit mode. Finally, to write the change to disk and quit the vi editor, enter `:wq` and press ENTER.
55. Confirm that the file was updated by running this command:
56. `cat template1.yaml | grep httpd`
57. The output of the command should return three lines of the file.
    The first line should show the `yum install -y httpd` line that you modified.
    Important: *If the yum line does not appear*, your changes might not have been saved. If you do not see the yum line, repeat the previous step.
58. Delete the failed stack:
59. `aws cloudformation delete-stack --stack-name myStack`
60. It might take a minute or two to delete the stack because many resources must be deleted.
61. Run the `describe-stacks` command.
62.
    ```
    watch -n 5 -d \
    aws cloudformation describe-stacks \
    --stack-name myStack \
    --output table
    ```
63. If you ran the previous command not long after running the `delete-stack` command, a table should display. The table should show that the StackStatus is *DELETE_IN_PROGRESS*.
    However, when the table disappears and the output indicates that the stack *does not exist*, exit the Linux watch utility by pressing CTRL+C.
64. Now that you resolved the issue in the template, run the `create-stack` command again:
65.
    ```
    aws cloudformation create-stack \
    --stack-name myStack \
    --template-body file://template1.yaml \
    --capabilities CAPABILITY_NAMED_IAM \
    --on-failure DO_NOTHING \
    --parameters ParameterKey=KeyName,ParameterValue=vockey
    ```

66. Run the `describe-stack-resources` command again and wait until all resources are created:
67.
    ```
    watch -n 5 -d \
    aws cloudformation describe-stack-resources \
    --stack-name myStack \
    --query 'StackResources[*].[ResourceType,ResourceStatus]' \
    --output table
    ```
68. Wait until there are no more stack resources with a status of *CREATE_IN_PROGRESS*.
    Tip: To exit the watch utility, press CTRL+C.
69. Run the `describe-stacks` command.
70.
    ```
    aws cloudformation describe-stacks \
    --stack-name myStack \
    --output table
    ```
71. This time, your stack should be created successfully (without errors), and it should have a StackStatus of *CREATE_COMPLETE*.
    Also notice that the **Outputs** section includes the PublicIP address of the web server and the name of the S3 bucket that was created.
72. Test the web server.
    - Copy the public IP address from the output of the previous command.
    - To load the server webpage, open a browser tab and enter the IP address.
    - A *Hello from your web server!* message should display.

Congratulations! You figured out why the stack was failing. You successfully discovered the root cause of the problem by looking at log files on the EC2 instance.

You completed the necessary troubleshooting and then updated your AWS CloudFormation template so that it successfully created a set of resources.

You used the *WaitCondition* in combination with the `-e` parameter that was specified in the line that starts with the #! characters at the top of the **userdata** section in your template. By using the WaitCondition with the `-e` parameter, you ensured that every command in the userdata script ran without error.



Hello from your web server!

Task 3: Make manual modifications and detect drift

In this task, you will intentionally modify a resource that was created by AWS CloudFormation, but you will modify the resource manually in the AWS Management Console.

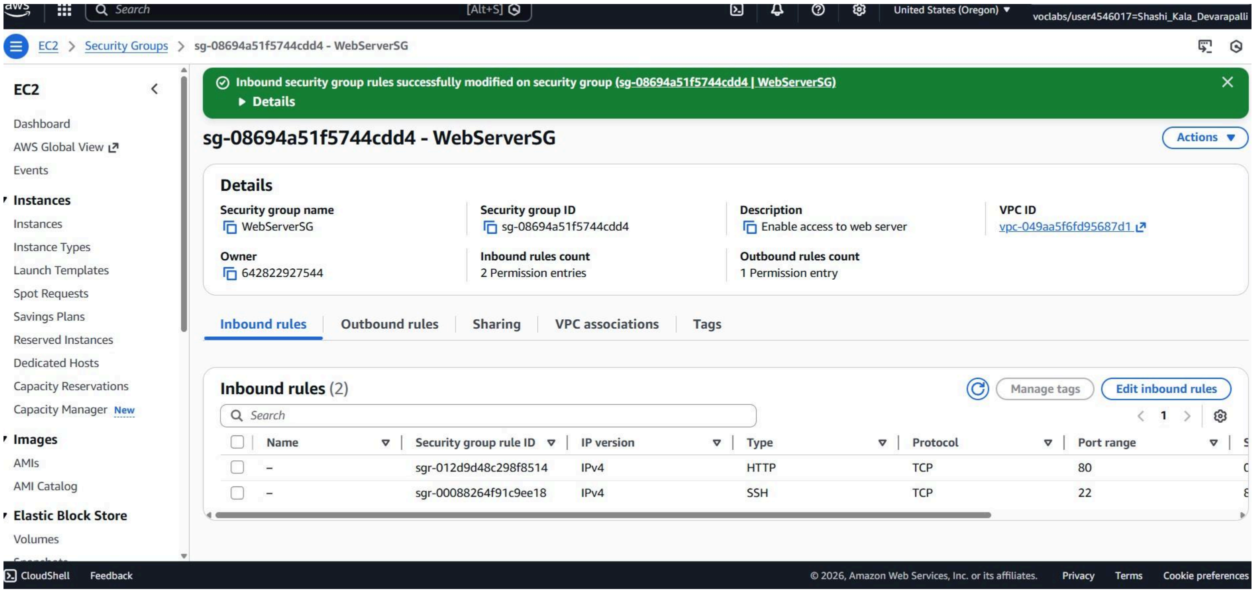
When you must modify resources that are created by AWS CloudFormation, it is a best practice to update the AWS CloudFormation template and then run the `update-stack` command. However, it is a common situation that other AWS account users might not know this best practice. They also might forget that it is important to follow this best practice. This task will give you hands-on practice noticing when these situations have occurred.

Task 3.1: Make manual modifications to the security groups

61. At the top of these instructions, click AWS
    This will open the AWS Management Console in a new browser tab. The system will automatically log you in.
    Tip: If a new browser tab does not open, there will typically be a banner or icon at the top of your browser indicating that your browser is preventing the site from

opening pop-up windows. Click on the banner or icon and choose "Allow pop ups."

62. Arrange the AWS Management Console tab so that it displays along side these instructions. Ideally, you will be able to see both browser tabs at the same time, to make it easier to follow the lab steps.
63. From the **Services** menu, choose **EC2**.
64. Click **Instances** and then select **Web Server**.
65. Click the **Security** tab, followed by the **WebServerSG** security group.
66. Click the **Inbound rules** tab and then click on **Edit inbound rules**.
67. Modify the existing SSH inbound rule.
    To modify the rule, in the **Source** column of the row that applies to port 22, click **Custom** and select **My IP**.
68. Click **Save rules**.



Inbound security group rules successfully modified on security group (sg-08694a51f5744cdd4 | WebServerSG)

▶ Details

sg-08694a51f5744cdd4 - WebServerSG

Details

Security group name: WebServerSG
Security group ID: sg-08694a51f5744cdd4
Description: Enable access to web server
VPC ID: vpc-049aa5f6fd95687d1
Owner: 642822927544
Inbound rules count: 2 Permission entries
Outbound rules count: 1 Permission entry

Inbound rules (2)

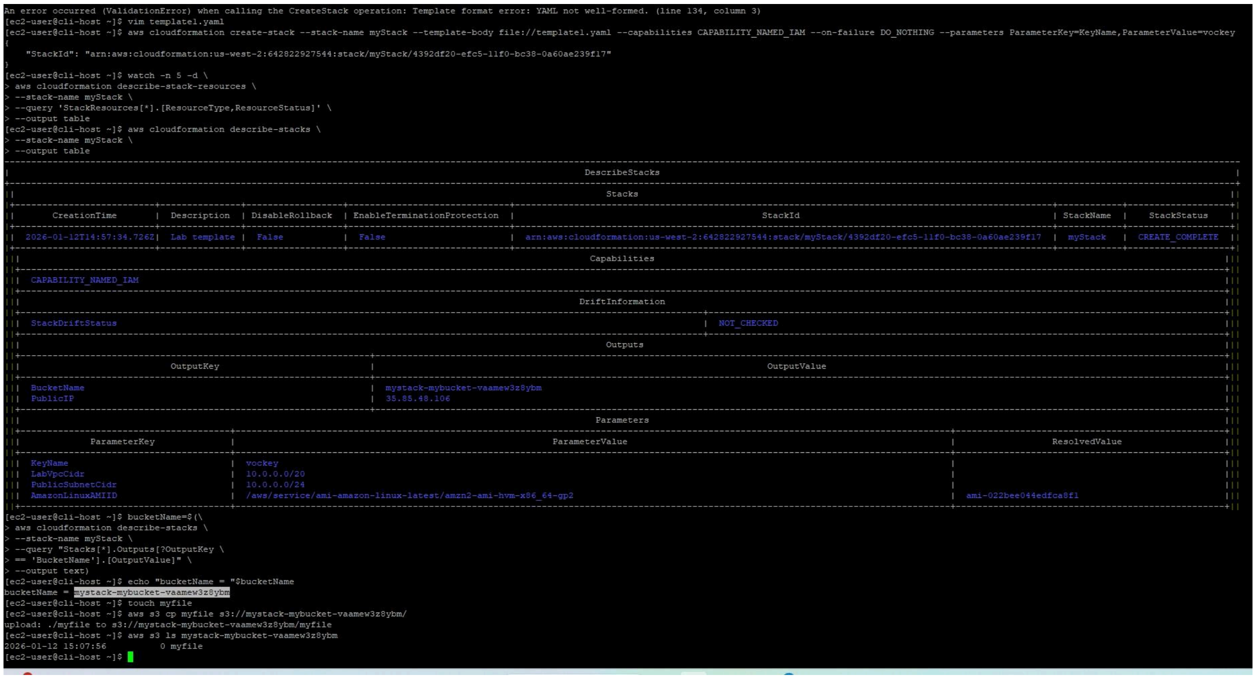
□	Name	Security group rule ID	IP version	Type	Protocol	Port range
□	–	sgr-012d9d48c298f8514	IPv4	HTTP	TCP	80
□	–	sgr-00088264f91c9ee18	IPv4	SSH	TCP	22

Task 3.2: Add an object to the S3 bucket.

69. From the terminal that is connected to the CLI Host, query the bucket name, assign it to a variable named *bucketName*, and echo result to the terminal by running the following command:
70.

```
bucketName=$(\
aws cloudformation describe-stacks \
--stack-name myStack \
--query "Stacks[*].Outputs[?OutputKey \
== 'BucketName'].[OutputValue]" \
--output text)
echo "bucketName = "$bucketName
```

71. Create an empty file.
72. `touch myfile`
73. Copy the file to the bucket by using the following command, which uses the *bucketName* variable that you defined:
74. `aws s3 cp myfile s3://$bucketName/`
75. Verify that the file is in the bucket:
76. `aws s3 ls $bucketName/`



```
An error occurred (ValidationError) when calling the CreateStack operation: Template format error: YAML not well-formed. (line 134, column 3)
[ec2-user@cli-host ~]$ vim template1.yaml
[ec2-user@cli-host ~]$ aws cloudformation create-stack --stack-name myStack --template-body file://template1.yaml --capabilities CAPABILITY_NAMED_IAM --on-failure DO_NOTHING --parameters ParameterKey=KeyName,ParameterValue=vockey
{
    "StackId": "arn:aws:cloudformation:us-west-2:642822927544:stack/myStack/4392df20-efc5-11f0-bc38-0a60ae239f17"
}
[ec2-user@cli-host ~]$ watch -n 5 -d \
> aws cloudformation describe-stack-resources \
> --stack-name myStack \
> --query 'StackResources[*].[ResourceType,ResourceStatus]' \
> --output table
[ec2-user@cli-host ~]$ aws cloudformation describe-stacks \
> --stack-name myStack \
> --output table
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
|                                                                       DescribeStacks                                                                                        |
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
||                                                                         Stacks                                                                                            ||
|+-----------------------------+---------------+------------------+------------------------------+------------------------------------------------------------------------------------------------+-----------+---------------------+|
||        CreationTime         |  Description  | DisableRollback  | EnableTerminationProtection  |                                                   StackId                                      | StackName |    StackStatus      ||
|+-----------------------------+---------------+------------------+------------------------------+------------------------------------------------------------------------------------------------+-----------+---------------------+|
||  2026-01-12T14:57:34.726Z|  Lab template |  False           |  False                       |  arn:aws:cloudformation:us-west-2:642822927544:stack/myStack/4392df20-efc5-11f0-bc38-0a60ae239f17  |  myStack  |  CREATE_COMPLETE  ||
|+-----------------------------+---------------+------------------+------------------------------+------------------------------------------------------------------------------------------------+-----------+---------------------+|
|||                                                                     Capabilities                                                                                         |||
||+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------+||
|||  CAPABILITY_NAMED_IAM                                                                                                                                                      |||
||+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------+||
|||                                                                    DriftInformation                                                                                      |||
||+-----------------------------------------------------------------------------------------------------------+----------------------------------------------------------------+||
|||  StackDriftStatus                                                                                         |  NOT_CHECKED                                                   |||
||+-----------------------------------------------------------------------------------------------------------+----------------------------------------------------------------+||
|||                                                                        Outputs                                                                                           |||
||+-------------------------------------------------------------+------------------------------------------------------------------------------------------------------------+||
|||                         OutputKey                           |                                        OutputValue                                                         |||
||+-------------------------------------------------------------+------------------------------------------------------------------------------------------------------------+||
|||  BucketName                                                 |  mystack-mybucket-vaamew3z8ybm                                                                             |||
|||  PublicIP                                                   |  35.85.48.106                                                                                              |||
||+-------------------------------------------------------------+------------------------------------------------------------------------------------------------------------+||
|||                                                                       Parameters                                                                                         |||
||+------------------------------------+-------------------------------------------------------------------------------------------------+-----------------------------------+||
|||            ParameterKey            |                                         ParameterValue                                          |          ResolvedValue            |||
||+------------------------------------+-------------------------------------------------------------------------------------------------+-----------------------------------+||
|||  KeyName                           |  vockey                                                                                         |                                   |||
|||  LabVpcCidr                        |  10.0.0.0/20                                                                                    |                                   |||
|||  PublicSubnetCidr                  |  10.0.0.0/24                                                                                    |                                   |||
|||  AmazonLinuxAMIID                  |  /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2                                  |  ami-022bee044edfca8f1            |||
||+------------------------------------+-------------------------------------------------------------------------------------------------+-----------------------------------+||
[ec2-user@cli-host ~]$ bucketName=$(\
> aws cloudformation describe-stacks \
> --stack-name myStack \
> --query "Stacks[*].Outputs[?OutputKey \
> == 'BucketName'].[OutputValue]" \
> --output text)
[ec2-user@cli-host ~]$ echo "bucketName = "$bucketName
bucketName = mystack-mybucket-vaamew3z8ybm
[ec2-user@cli-host ~]$ touch myfile
[ec2-user@cli-host ~]$ aws s3 cp myfile s3://mystack-mybucket-vaamew3z8ybm/
upload: ./myfile to s3://mystack-mybucket-vaamew3z8ybm/myfile
[ec2-user@cli-host ~]$ aws s3 ls mystack-mybucket-vaamew3z8ybm
2026-01-12 15:07:56          0 myfile
[ec2-user@cli-host ~]$ 
```

Task 3.3: Detect drift

73. To start drift detection on your stack, run the following command:
74. `aws cloudformation detect-stack-drift --stack-name myStack`
75. The command should return a StackDriftDetectionId.
76. Monitor the status of the drift detection by running the following command (replace <*driftId*> with the actual value of *StackDriftDetectionId*):
77. `aws cloudformation describe-stack-drift-detection-status \`
    `--stack-drift-detection-id driftId`

78. Notice that the output shows *"StackDriftStatus": "DRIFTED"*
79. Finally, describe the resources that drifted by running the following `describe-stack-resource-drifts` command:
80.
    ```
    aws cloudformation describe-stack-resource-drifts \
    --stack-name myStack
    ```
81. The output from the command is extensive. Try a different approach.
82. Run a `describe-stack-resources` command with a query parameter that will return only the resource type, resource status, and drift status.
    The following command outputs the results as a table:
83.
    ```
    aws cloudformation describe-stack-resources \
    --stack-name myStack \
    --query
    'StackResources[*].[ResourceType,ResourceStatus,DriftInformation.StackResourceDriftStatus]' \
    --output table
    ```
84. This output is easier to read because of the query parameter, which is written in JMESPath.
    Notice that not all resources are checked for drift. However, the resources that are checked for drift show a status.
    On this stack, all checked resources have a status of *IN_SYNC*, except for the security group that you manually modified, which has a status of *MODIFIED*.
    Also notice that though you placed an object in the S3 bucket, the bucket still shows a status of *IN_SYNC*. If you had modified some property of the bucket, then the bucket would show a status of *MODIFIED*. However, only adding files to a bucket does not register as drift in AWS CloudFormation.
85. Retrieve the specific details of the drift for the resource that has a StackResourceDriftStatus of *MODIFIED*:
86.
    ```
    aws cloudformation describe-stack-resource-drifts \
    --stack-name myStack \
    --stack-resource-drift-status-filters MODIFIED
    ```
87. Notice the **PropertyDifferences** section of the output. It should show that port 22 is now open only to your IP address, instead of the 0.0.0.0/0 Classless Inter-Domain Routing (CIDR) block that is defined in the AWS CloudFormation template.
88. Try updating the stack:
89.
    ```
    aws cloudformation update-stack \
    --stack-name myStack \
    --template-body file://template1.yaml \
    --parameters ParameterKey=KeyName,ParameterValue=vockey
    ```

90. The output should indicate that an error occurred. *This is expected.*
    The `update-stack` command will not automatically resolve drift, though drift has occurred. You must manually resolve these issues to eliminate the drift.

Task 4: Attempt to delete the stack

There might be occasions when you want to completely delete a stack, such as when you finish running tests in a test environment that you no longer need. Or perhaps you want to save on costs if you don't need to use the environment resources for a while. In such situations, you know that if you need these resources again, you can re-create them by using your proven template to create a new stack.

In this last task, you will try to delete the stack. The attempt will fail. You will then be given a challenge to resolve the issue.

79. Try deleting the stack by running the following command:
80.
    ```
    aws cloudformation delete-stack --stack-name myStack
    ```
81. Observe the results by running the `describe-stack-resources` command:
82.
    ```
    watch -n 5 -d \
    aws cloudformation describe-stack-resources \
    --stack-name myStack \
    --query 'StackResources[*].[ResourceType,ResourceStatus]' \
    --output table
    ```
83. Observe how the status of each resource changes.
    Notice that most of the resources are successfully deleted. However, there is one resource that fails to delete. It is the S3 bucket.
    Once all resources have a status if either DELETE_COMPLETE or DELETE_FAILED, use the `Ctrl-C` keyboard combination to exit the watch command.
84. Run the describe-stacks command to see the stack status:
85.
    ```
    aws cloudformation describe-stacks \
    --stack-name myStack \
    --output table
    ```
86. Notice that the "StackStatus" shows "DELETE_FAILED"
    Also notice the "StackStatusReason": "The following resource(s) failed to delete: [MyBucket]. "
    CloudFormation will not delete a bucket that has objects in it. This is to help guard against accidental data loss.

Challenge: Keep the file in the S3 bucket, but Still Delete the Stack

One approach to the issue you face would be to manually delete or move the file object that is in the S3 bucket and then run the delete-stack command again. However, that approach may not be appropriate if people in the organization have already starting storing a large number of files in the bucket and other systems now depend on the bucket name and location not changing.

Your challenge is to:

- Figure out how to keep the bucket and keep the file in it, but still successfully delete the stack so that the status of the stack becomes DELETE_COMPLETE.
- Use the AWS CLI to solve the problem (avoid using the AWS Management Console)

Tips:

- **TIP #1:** Make use of the AWS CLI Command Reference. Do any options exist to keep resources created by the stack, when running the `delete-stack` command?
- **TIP #2:** What aws cloudformation command would provide you with resource logical IDs? Can you determine the logical ID by which CloudFormation knows the S3 bucket it created?

OPTIONAL BONUS CHALLENGE: can you compose a JMESpath expression and use it in a `--query` parameter, so that the AWS CLI command returns *only* the LogicalResourceId? Recall what you practiced in Task 1. Suggestion: surround the query part of the command with double quotes (not single quotes).

- **TIP #3:** Oftentimes the solution to a challenge requires you to use the output from one command as input to another command, to achieve success.

If you successfully manage to delete the CloudFormation stack using the AWS CLI, without removing the file from the S3 bucket or modifying the S3 bucket in any way, then you have succeeded.

Congratulations, you successfully did some troubleshooting and resolved the failed delete-stack command that left a stack in an undesirable state.

Update from Café

When Sofîa went into work the next morning at the Café, she was excited to tell Nikhil and the others about the proof of concept work she had done the night before, using CloudFormation.

She explained how this "Infrastructure as Code" approach to creating cloud deployments was causing her to completely rethink the cafe's current approach to how they manage the AWS resources that support the cafe website and other business applications.

Nikhil was very interested in what she was describing. They starting discussing how they could use this technology to reliably create matching but separate *development* and *production* environments, and how useful that would be for new feature development. They also saw the value provided by powerful features such as drift detection and the ability to tear down and then rebuild complex cloud infrastructures consisting of resources from multiple AWS services.