



Vivekanand Education Society's

Institute of Technology

An Autonomous Institute Affiliated to University of Mumbai,, Approved by AICTE & Recognized by Govt. of Maharashtra
Hashu Advani Memorial Complex, Collector Colony, Chembur East, Mumbai - 400074.

Department of Information Technology

A.Y. 2024-25

Advance DevOps Lab

Assignment 02

Aim: Deploying AWS Infrastructure Using Terraform: A Hands-On Approach with S3, SQS, and Lambda Integration

Roll No.	44
Name	Ganesh Sanjay Pandhre
Class	D15B
Subject	Advance DevOps Lab
LO Mapped	LO6: To engineer a composition of nano services using AWS Lambda and Step Functions with the Serverless Framework
Grade:	

Aim : Deploying AWS Infrastructure Using Terraform: A Hands-On Approach with S3, SQS, and Lambda Integration Guidelines

Theory :

Infrastructure as Code (IaC)

Infrastructure as Code (IaC) automates the management of IT infrastructure through code rather than manual processes, enabling consistent and repeatable deployments.

Overview of Terraform

Terraform is an open-source IaC tool that allows users to define cloud infrastructure using HashiCorp Configuration Language (HCL). Key features include:

- **Declarative Configuration:** Users specify the desired state of the infrastructure.
- **Execution Plan:** Terraform generates a plan detailing the actions needed to achieve that state.
- **Resource Management:** It manages the lifecycle of cloud resources.

Amazon S3 (Simple Storage Service)

Amazon S3 is a scalable storage solution that allows users to store and retrieve data from anywhere. Key features include:

- **Buckets:** Containers for organizing data.
- **Object Storage:** Stores data as objects with unique identifiers.
- **Use Cases:** Backup, data archiving, and serving static content.

Amazon SQS (Simple Queue Service)

Amazon SQS is a managed message queuing service that decouples application components, allowing for asynchronous communication. Key features include:

- **Queues:** Store messages for processing by consumers.
- **Message Retention:** Retains messages for a configurable time.
- **Use Cases:** Event-driven architectures and inter-service communication.

AWS Lambda

AWS Lambda is a serverless computing service that runs code without managing servers. Key features include:

- **Event-Driven Execution:** Triggered by AWS services like S3 and SQS.
- **Pay-as-You-Go Pricing:** Users pay only for the compute time used.
- **Use Cases:** Data processing and responding to events.

Integration of S3, SQS, and Lambda

Integrating these services enables powerful workflows. For example, an object uploaded to S3 can trigger a Lambda function, which processes the data and sends a message to SQS, allowing other services to react asynchronously.

Start the Learner Lab and export the credentials from the CLI.

ALLv2EN... > Modules > AWS Acad...
 > Launch AWS Academy Learner Lab

AWS Used \$0.3 of \$50 03:54 ▶ Start Lab ■ End Lab ⓘ AWS Details ⓘ Readme ↺ Reset ✕

```

eee_N_3428291@runweb148728:~$ export AWS_ACCESS_KEY_ID="ASIA5KXGVM8K2M5VBZ75"
eee_N_3428291@runweb148728:~$ export AWS_SECRET_ACCESS_KEY="Q+wyIVDy0SsG%myrAFzRHdH1annHjBjTUMvM"
eee_N_3428291@runweb148728:~$ export AWS_SESSION_TOKEN="Iqo3b3j2Z1uX2VjEj//////////wEaCXVzLXdlc3QthHjHMEUCIDE+rHA
2+MrEv72wQUHw56L8+031TD1L1t+0J57VcYMAIEApAqrbKJNaG1c31YuDn0bJezFhC84kj+QJrRVstIQgtgIIMRABGwSMYzODUIMTISMTciDK
10eZ1lneyFH4C2SxqTat8nQmN8rXhKkxQF5jY02Nma6JaC1VxD+LAsrr5usRTK4j4+1bwj22jd4j/YX41kwiCXUeKEJNRkpZjropv4Hfbue7
TN21Ryq2p08S5eiekDQ1j8tH511C7r9FxoXmcBP0R1FcXy3jzj6K60pRyM+vJ4AobtdgV08PU+CGgC6UV/K3/yzHeB+ZYvY8ZRO/2oTYmkkYH1C1
gwgNu468DwvKTAk9P5PCFUSnzaorIId0d3bIdapFRk9AMQFQ0kHx+R1HemJ1HwkE9Ij8a9J4ckVhIdR3EmdCFQV1833so31k0ktsCQCDPpa312y1A9M
TSb01vY1K19p+QBTm1X2mUB1mSM3e4carR8YmTVUkM0LnxLgG0p0B+0057HmIV9JRq1asc5j6oFZpq7q484wZn1mP403Hzfndp2wBU001cw7XDA648
h0P6NbYYx93JhzKKEN14D8Epm1Rf3gG08o/Ne5g0j9bV2YG0rRDeONFco31UYSF9pnXWb71okWZJz/E+ITIXJ1bpumVHs4aAAwIpj6ZFwkbcW
x/BWzWfEfa2j+bt2jKpFeR5+EV5ktIiA=="
eee_N_3428291@runweb148728:~$
  
```

In the Learner Lab, there is usually a predefined IAM role that you can use. This role should already have the necessary permissions to interact with AWS services (like Lambda and S3).

www.google.com - Search | aws learner lab - Search | Launch AWS Academy Learner | Console Home | Console Home | +

https://us-east-1.console.aws.amazon.com/console/home?region=us-east-1#

Services Search [Alt+S] N. Virginia voclabs/user3389100=2022.ganesh.pandhre@ves.ac.in @ 6161-5127...

Roles (21) Info [Refresh] [Delete] [Create role]

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Search

<input type="checkbox"/>	Role name	Trusted entities	Last activity
<input type="checkbox"/>	LabRole	Account: 916385512917	12 hours ago

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

www.google.com - Search | aws learner lab - Search | Launch AWS Academy Learner | Console Home | Console Home | +

https://us-east-1.console.aws.amazon.com/console/home?region=us-east-1#

Services Search [Alt+S] N. Virginia voclabs/user3389100=2022.ganesh.pandhre@ves.ac.in @ 6161-5127...

LabRole Info [Delete] [Edit]

Summary

Creation date
August 07, 2024, 09:06 (UTC+05:30)

Last activity
12 hours ago

Maximum session duration
1 hour

ARN copied
arn:aws:iam::916385512917:role/LabRole

Link to switch roles in console
https://signin.aws.amazon.com/switchrole?roleName=LabRole&account=916385512917

Instance profile ARN
arn:aws:iam::916385512917:instance-profile/LabInstanceProfile

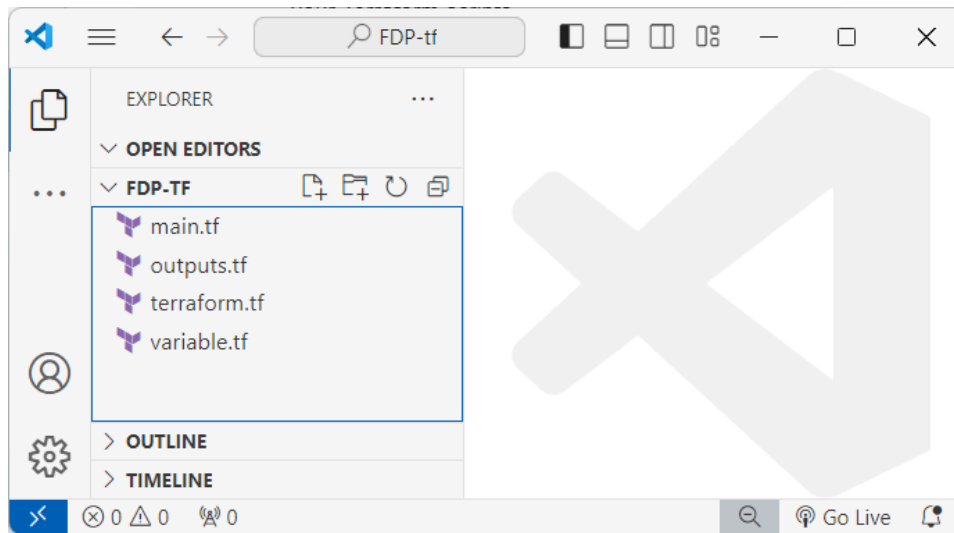
CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Create a Folder for the Project:

- Create a new folder on your local machine (for example: **FDP-tf**) where you will store your Terraform scripts.

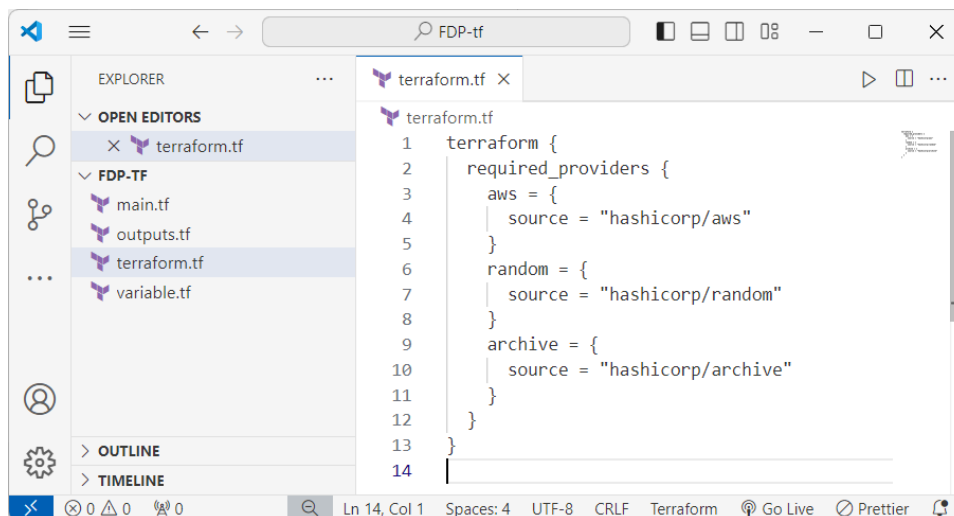
Set Up Terraform Configuration:

- Inside your folder, create four files:
 - terraform.tf
 - main.tf
 - variable.tf
 - outputs.tf



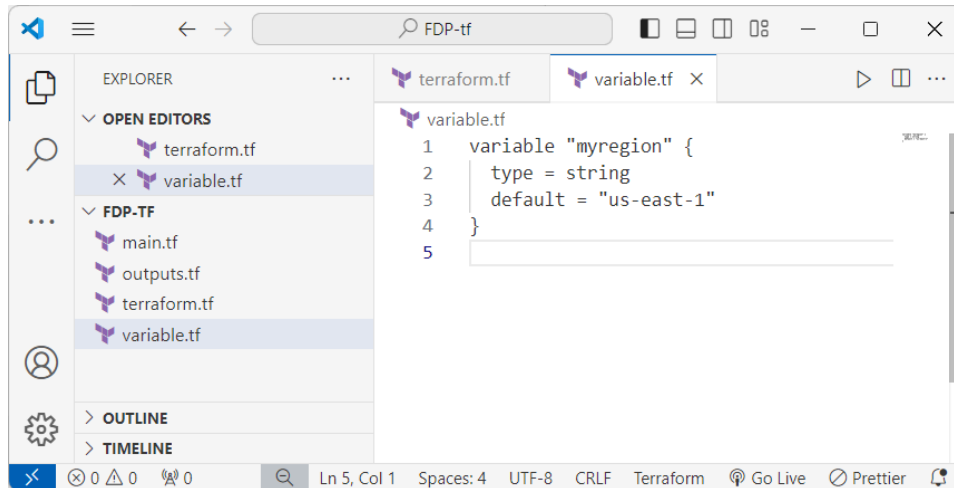
terraform.tf

```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
    }
    random = {
      source = "hashicorp/random"
    }
    archive = {
      source = "hashicorp/archive"
    }
  }
}
```



variable.tf

```
variable "myregion" {  
  type = string  
  default = "us-east-1"  
}
```



main.tf

```
provider "aws" {  
  access_key = "YOUR_ACCESS_KEY"  
  secret_key = "YOUR_SECRET_KEY"  
  token      = "YOUR_TOKEN"  
  region     = var.myregion  
}  
  
resource "random_pet" "bucketname" {  
  length = 3  
  prefix = "fdp"  
}  
  
resource "aws_s3_bucket" "mybucket" {  
  bucket = random_pet.bucketname.id  
}  
  
resource "aws_sqs_queue" "myqueue" {  
  name = "mySQSqueue"  
}  
  
data "archive_file" "zip" {  
  type = "zip"  
  source_file = "lambda_function.py"  
  output_path = "lambda_function.zip"  
}  
  
resource "aws_lambda_function" "mylambda" {
```

```

function_name = "SqsToS3Function"
runtime = "python3.8"
filename = data.archive_file.zip.output_path
source_code_hash = filebase64sha256("lambda_function.zip")
handler = "lambda_function.handler"
role = "arn:aws:iam::YOUR_IAM_ROLE"
environment {
  variables = {
    S3_BUCKET = random_pet.bucketname.id
  }
}
}

resource "aws_lambda_event_source_mapping" "SqsToLambda" {
  event_source_arn = aws_sqs_queue.myqueue.arn
  function_name    = aws_lambda_function.mylambda.arn
  batch_size      = 1
}

```

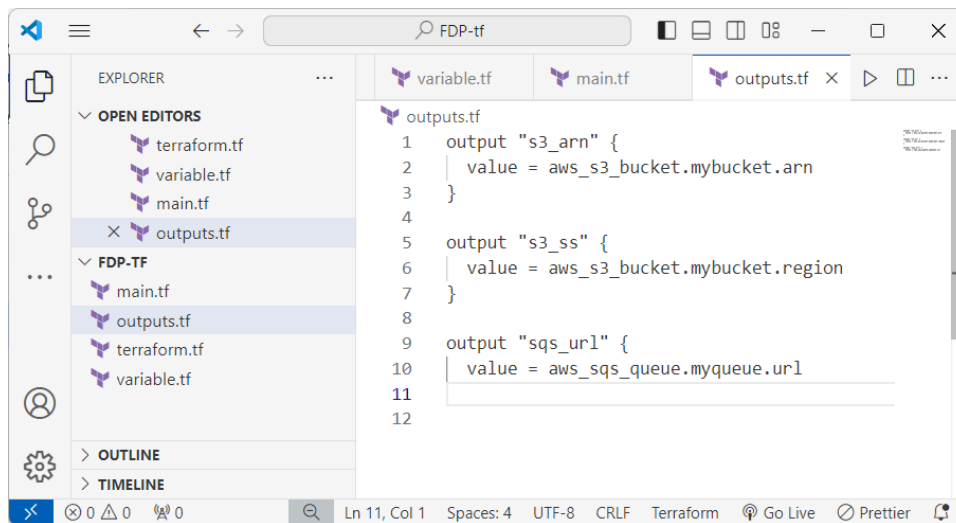
```

1  provider "aws" {
2    access_key = "ASIA5KXGVHXX2M5YBZ75"
3    secret_key = "Q+wyNYDWy0SsGMxmyrAFzRHdH1anmHIjBwTUVmVm"
4    token      = "IQoJb3JpZ2luX2VjEMj////////wEaCXVzLXdlc3QtMiJHMEUCI
5    region    = var.myregion
6  }
7
8  resource "random_pet" "bucketname" {
9    length = 3
10   prefix = "fdp"
11 }
12
13 resource "aws_s3_bucket" "mybucket" {
14   bucket = random_pet.bucketname.id
15 }
16
17 resource "aws_sqs_queue" "myqueue" {
18   name = "mySQSqueue"
19 }
20
21 data "archive_file" "zip" {
22   type = "zip"
23   source_file = "lambda_function.py"
24   output_path = "lambda_function.zip"
25 }
26
27 resource "aws_lambda_function" "mylambda" {
28   function_name = "SqsToS3Function"
29   runtime       = "python3.8"
30   filename      = data.archive_file.zip.output_path
31   source_code_hash = filebase64sha256("lambda_function.zip")
32   handler       = "lambda_function.handler"
33   role          = "arn:aws:iam::916385512917:role/LabRole"
34   environment {
35     variables = {
36       S3_BUCKET = random_pet.bucketname.id
37     }
38   }
39 }

```

outputs.tf

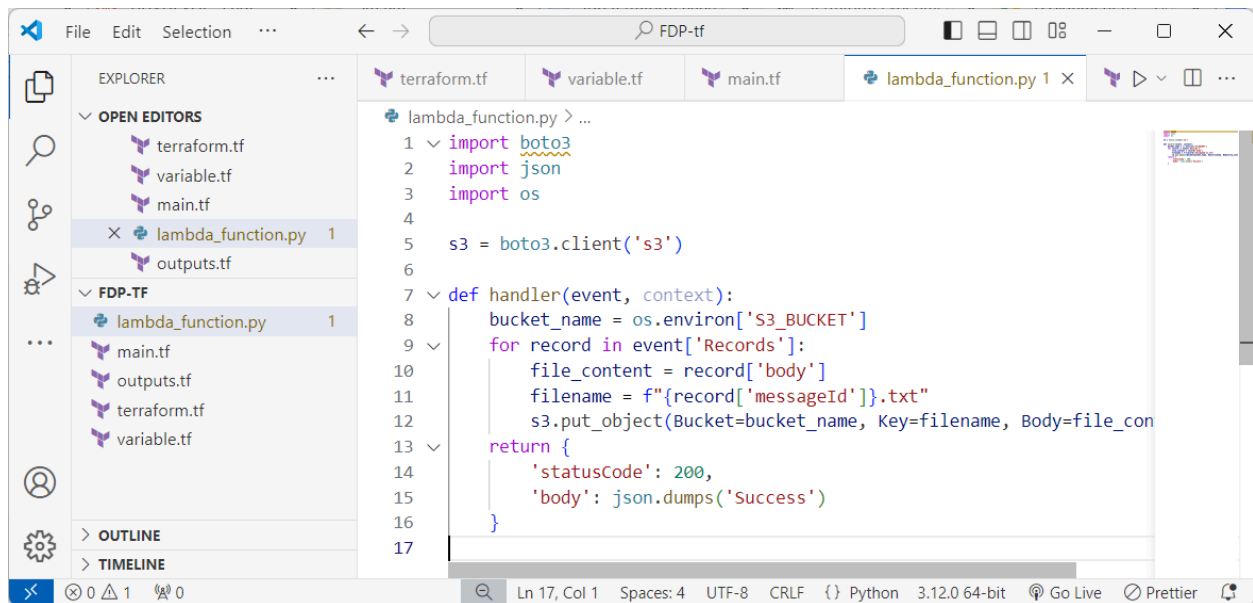
```
output "s3_arn" {  
  value = aws_s3_bucket.mybucket.arn  
}  
  
output "s3_ss" {  
  value = aws_s3_bucket.mybucket.region  
}  
  
output "sqs_url" {  
  value = aws_sqs_queue.myqueue.url  
}
```



Create Lambda Python File:

- In the same directory, create a file named `lambda_function.py` and paste the following code:

```
import boto3  
import json  
import os  
  
s3 = boto3.client('s3')  
  
def handler(event, context):  
    bucket_name = os.environ['S3_BUCKET']  
    for record in event['Records']:  
        file_content = record['body']  
        filename = f"{record['messageId']}.txt"  
        s3.put_object(Bucket=bucket_name, Key=filename, Body=file_content)  
    return {  
        'statusCode': 200,  
        'body': json.dumps('Success')  
    }
```

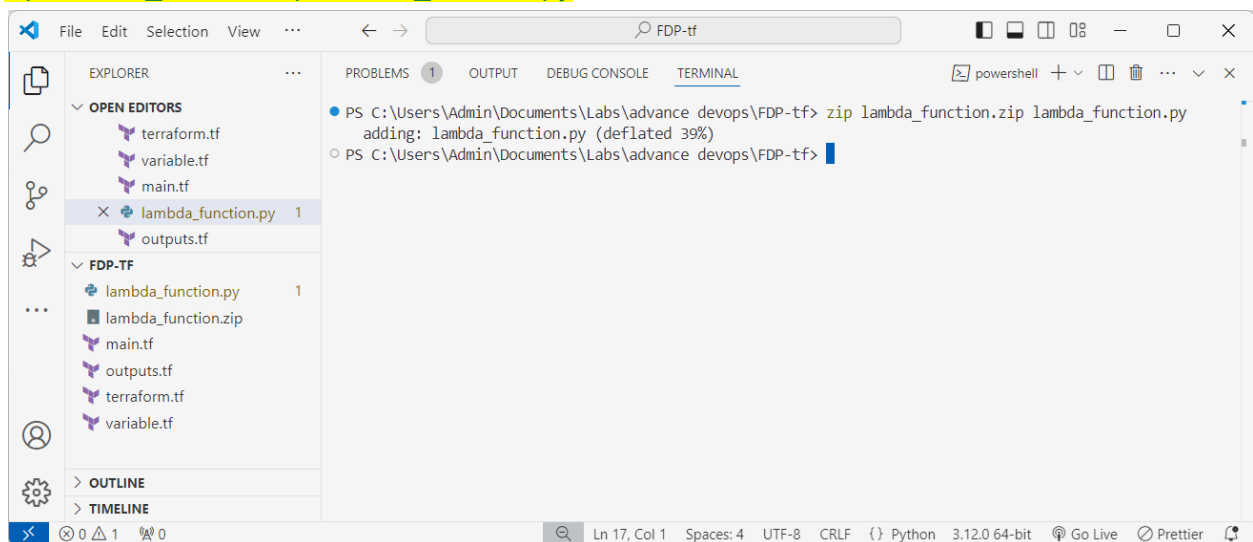


The screenshot shows the Visual Studio Code editor with the file `lambda_function.py` open. The Explorer sidebar on the left shows the project structure with files `terraform.tf`, `variable.tf`, `main.tf`, `lambda_function.py`, and `outputs.tf`. The `lambda_function.py` file is selected, and its content is displayed in the editor. The code is a Python lambda function that uses `boto3` to upload a file to an S3 bucket. The status bar at the bottom indicates the file is at line 17, column 1, with 4 spaces, UTF-8 encoding, and CRLF line endings. The status bar also shows the Python interpreter is 3.12.0 64-bit and that Prettier is installed.

```
1 import boto3
2 import json
3 import os
4
5 s3 = boto3.client('s3')
6
7 def handler(event, context):
8     bucket_name = os.environ['S3_BUCKET']
9     for record in event['Records']:
10         file_content = record['body']
11         filename = f"{record['messageId']}.txt"
12         s3.put_object(Bucket=bucket_name, Key=filename, Body=file_content)
13     return {
14         'statusCode': 200,
15         'body': json.dumps('Success')
16     }
17
```

Open the terminal or command prompt in the same directory and run:

`zip lambda_function.zip lambda_function.py`

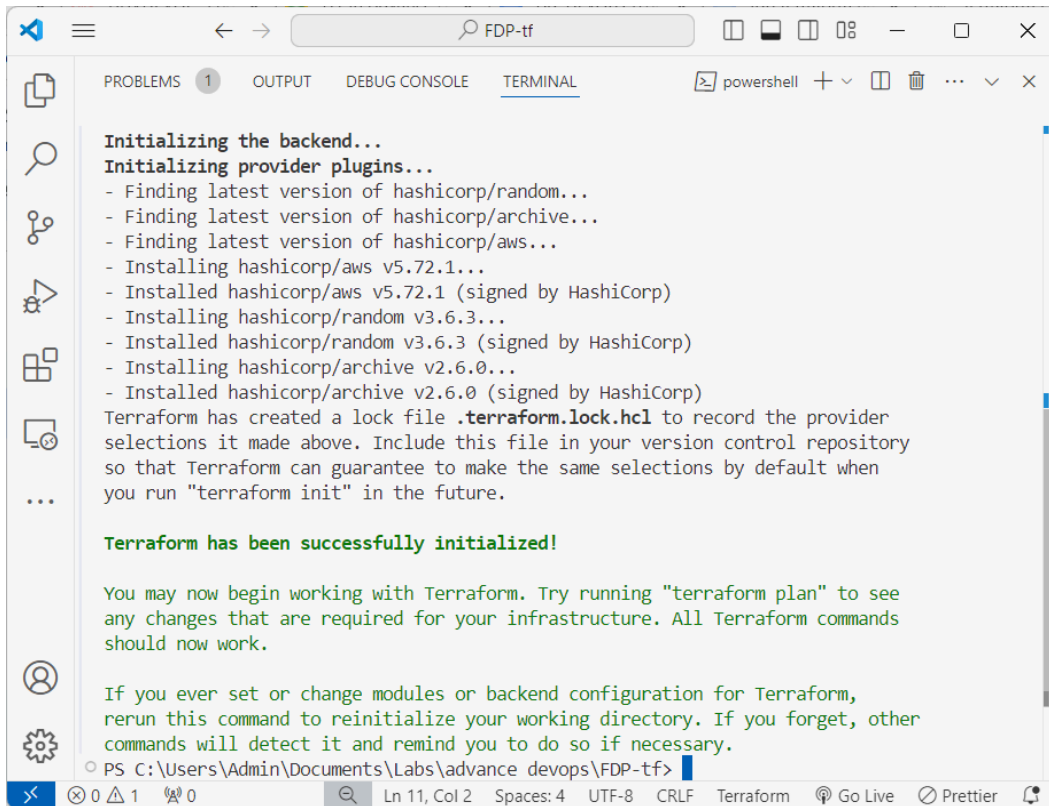


The screenshot shows the Visual Studio Code editor with the `TERMINAL` tab active. The terminal output shows the command `zip lambda_function.zip lambda_function.py` being executed. The output indicates that the file `lambda_function.py` was added to the zip file, deflated by 39%. The Explorer sidebar on the left shows the project structure with files `terraform.tf`, `variable.tf`, `main.tf`, `lambda_function.py`, and `outputs.tf`. The `lambda_function.py` file is selected, and its content is displayed in the editor. The status bar at the bottom indicates the file is at line 17, column 1, with 4 spaces, UTF-8 encoding, and CRLF line endings. The status bar also shows the Python interpreter is 3.12.0 64-bit and that Prettier is installed.

```
PS C:\Users\Admin\Documents\Labs\advance devops\FDP-tf> zip lambda_function.zip lambda_function.py
adding: lambda_function.py (deflated 39%)
PS C:\Users\Admin\Documents\Labs\advance devops\FDP-tf>
```

To Initialize Terraform, run the command:

`terraform init`



```
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/random...
- Finding latest version of hashicorp/archive...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.72.1...
- Installed hashicorp/aws v5.72.1 (signed by HashiCorp)
- Installing hashicorp/random v3.6.3...
- Installed hashicorp/random v3.6.3 (signed by HashiCorp)
- Installing hashicorp/archive v2.6.0...
- Installed hashicorp/archive v2.6.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

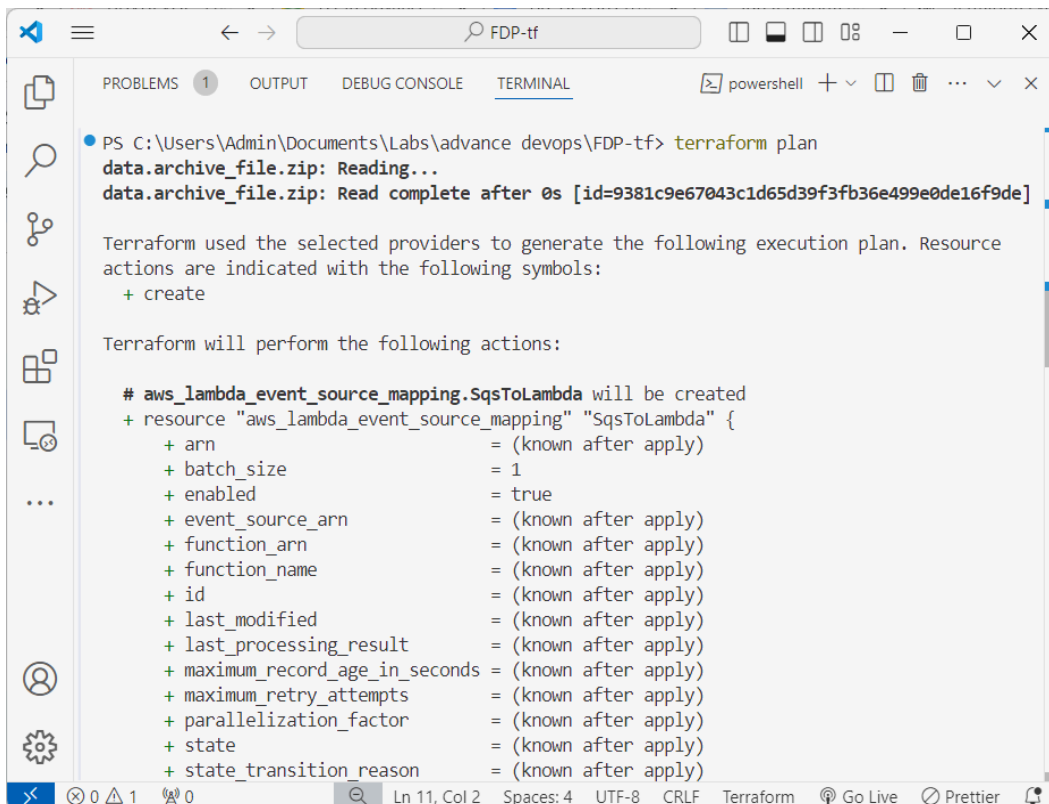
Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\Admin\Documents\Labs\advance devops\FDP-tf>
```

To Plan the Infrastructure, Run the command

terraform plan

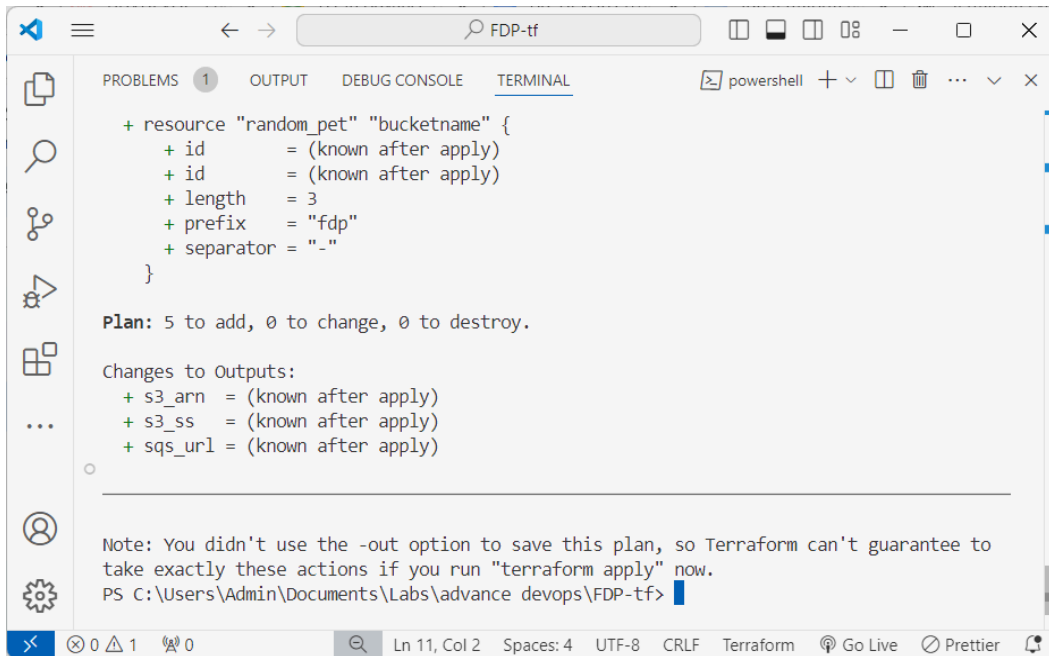


```
PS C:\Users\Admin\Documents\Labs\advance devops\FDP-tf> terraform plan
data.archive_file.zip: Reading...
data.archive_file.zip: Read complete after 0s [id=9381c9e67043c1d65d39f3fb36e499e0de16f9de]

Terraform used the selected providers to generate the following execution plan. Resource
actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# aws_lambda_event_source_mapping.SqsToLambda will be created
+ resource "aws_lambda_event_source_mapping" "SqsToLambda" {
  + arn                = (known after apply)
  + batch_size         = 1
  + enabled             = true
  + event_source_arn   = (known after apply)
  + function_arn       = (known after apply)
  + function_name      = (known after apply)
  + id                 = (known after apply)
  + last_modified      = (known after apply)
  + last_processing_result = (known after apply)
  + maximum_record_age_in_seconds = (known after apply)
  + maximum_retry_attempts = (known after apply)
  + parallelization_factor = (known after apply)
  + state              = (known after apply)
  + state_transition_reason = (known after apply)
}
```



```
PS C:\Users\Admin\Documents\Labs\advance devops\FDP-tf> terraform plan

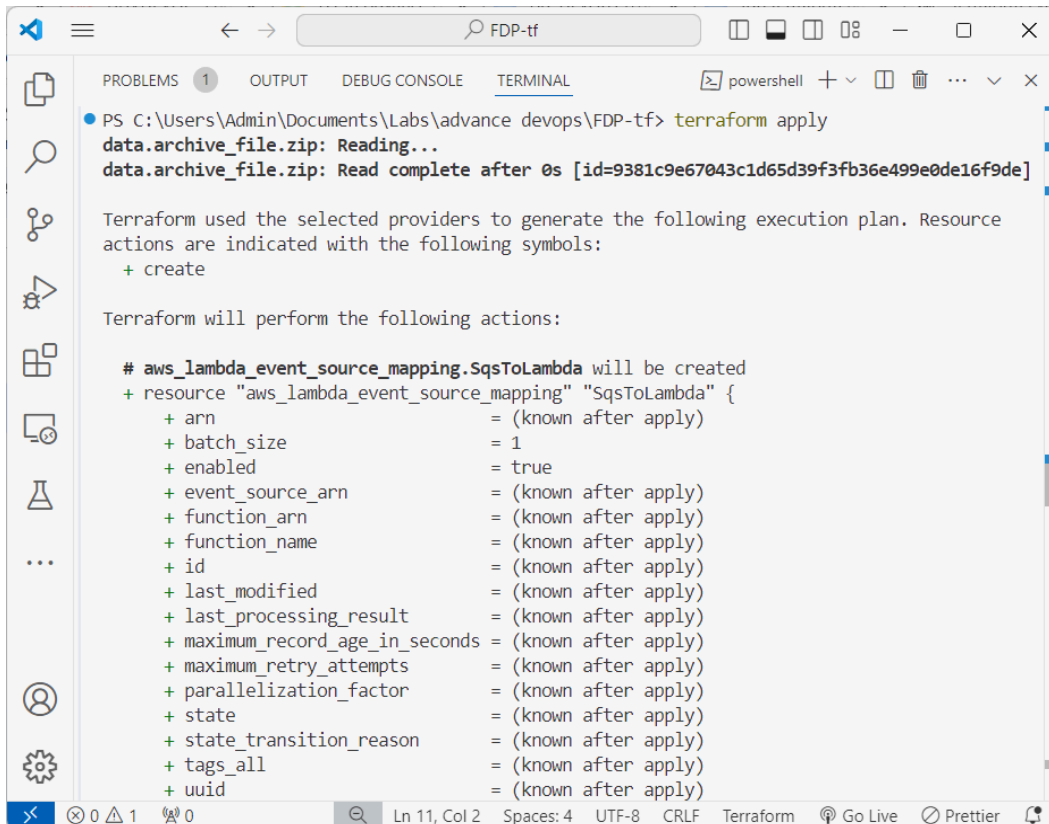
+ resource "random_pet" "bucketname" {
+   id       = (known after apply)
+   id       = (known after apply)
+   length   = 3
+   prefix   = "fdp"
+   separator = "-"
+ }

Plan: 5 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ s3_arn = (known after apply)
+ s3_ss  = (known after apply)
+ sqs_url = (known after apply)

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to
take exactly these actions if you run "terraform apply" now.
PS C:\Users\Admin\Documents\Labs\advance devops\FDP-tf>
```

If everything looks good, apply the plan by running **terraform apply**
(Enter **yes** when prompted. This will create the resources.)

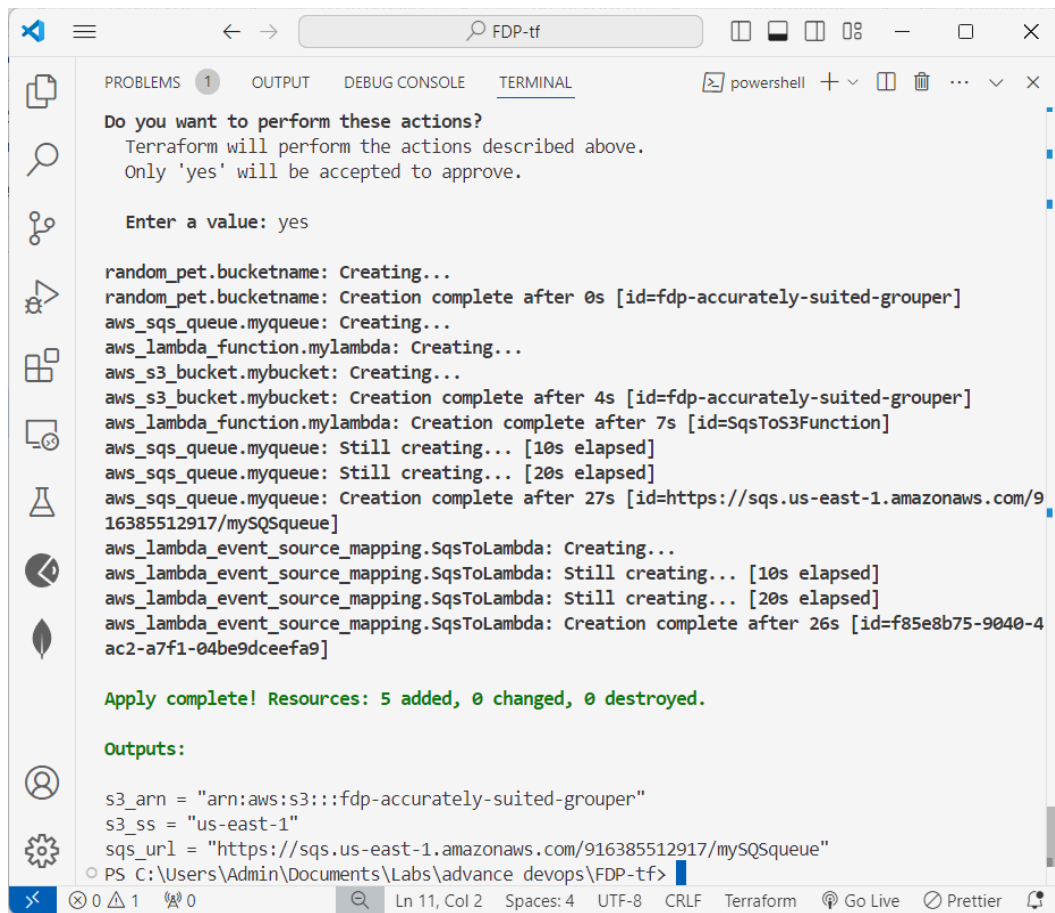


```
PS C:\Users\Admin\Documents\Labs\advance devops\FDP-tf> terraform apply
data.archive_file.zip: Reading...
data.archive_file.zip: Read complete after 0s [id=9381c9e67043c1d65d39f3fb36e499e0de16f9de]

Terraform used the selected providers to generate the following execution plan. Resource
actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# aws_lambda_event_source_mapping.SqsToLambda will be created
+ resource "aws_lambda_event_source_mapping" "SqsToLambda" {
+   arn              = (known after apply)
+   batch_size       = 1
+   enabled          = true
+   event_source_arn = (known after apply)
+   function_arn     = (known after apply)
+   function_name    = (known after apply)
+   id               = (known after apply)
+   last_modified    = (known after apply)
+   last_processing_result = (known after apply)
+   maximum_record_age_in_seconds = (known after apply)
+   maximum_retry_attempts = (known after apply)
+   parallelization_factor = (known after apply)
+   state            = (known after apply)
+   state_transition_reason = (known after apply)
+   tags_all         = (known after apply)
+   uuid             = (known after apply)
+ }
```



```
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

random_pet.bucketname: Creating...
random_pet.bucketname: Creation complete after 0s [id=fdp-accurately-suited-grouper]
aws_sqs_queue.myqueue: Creating...
aws_lambda_function.mylambda: Creating...
aws_s3_bucket.mybucket: Creating...
aws_s3_bucket.mybucket: Creation complete after 4s [id=fdp-accurately-suited-grouper]
aws_lambda_function.mylambda: Creation complete after 7s [id=SqsToS3Function]
aws_sqs_queue.myqueue: Still creating... [10s elapsed]
aws_sqs_queue.myqueue: Still creating... [20s elapsed]
aws_sqs_queue.myqueue: Creation complete after 27s [id=https://sqs.us-east-1.amazonaws.com/916385512917/mySQSqueue]
aws_lambda_event_source_mapping.SqsToLambda: Creating...
aws_lambda_event_source_mapping.SqsToLambda: Still creating... [10s elapsed]
aws_lambda_event_source_mapping.SqsToLambda: Still creating... [20s elapsed]
aws_lambda_event_source_mapping.SqsToLambda: Creation complete after 26s [id=f85e8b75-9040-4ac2-a7f1-04be9dceefa9]

Apply complete! Resources: 5 added, 0 changed, 0 destroyed.

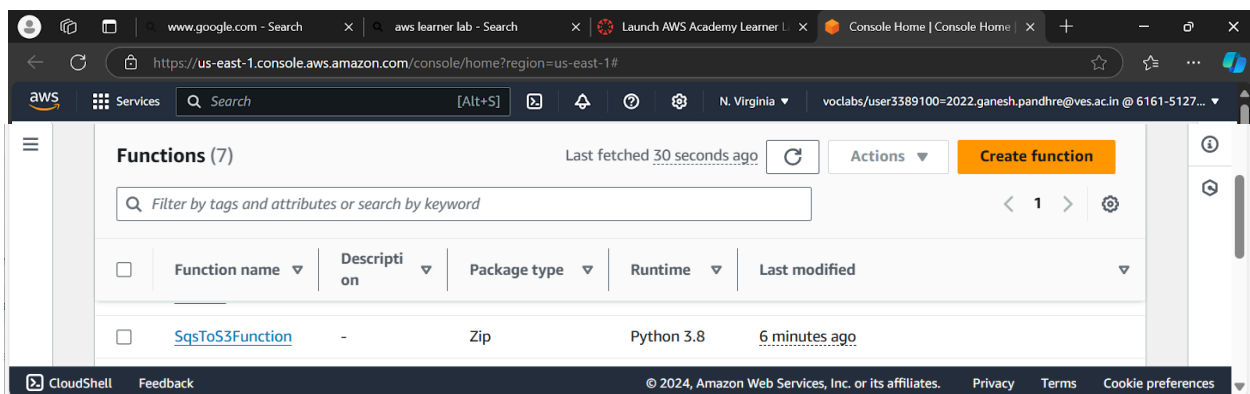
Outputs:

s3_arn = "arn:aws:s3:::fdp-accurately-suited-grouper"
s3_ss = "us-east-1"
sqs_url = "https://sqs.us-east-1.amazonaws.com/916385512917/mySQSqueue"
PS C:\Users\Admin\Documents\Labs\advance_devops\FDP-tf>
```

Once the resources are created, you can log into your AWS console and verify that:

- An S3 bucket is created.
- An SQS queue is created.
- A Lambda function is created.

Lambda Function



The screenshot shows the AWS Lambda console for the function 'SqsToS3Function'. The 'Function overview' tab is active, displaying a diagram of the function architecture. The diagram shows an 'SqsToS3Function' box connected to an 'SQS' box. Below the diagram is a '+ Add trigger' button. To the right of the diagram, there is a 'Layers' section showing '(0)' layers and an '+ Add destination' button. On the far right, a 'Description' section contains the following information:

- Description: -
- Last modified: 10 minutes ago
- Function ARN: `arn:aws:lambda:us-east-1:9163855129:17:function:SqsToS3Function`
- Function URL: [Info](#)

At the top of the console, there are buttons for 'Throttle', 'Copy ARN', and 'Actions'. Below the 'Function overview' tab, there are buttons for 'Export to Application Composer' and 'Download'. The bottom of the console shows the 'CloudShell' and 'Feedback' buttons, along with the copyright notice '© 2024, Amazon Web Services, Inc. or its affiliates.' and links for 'Privacy', 'Terms', and 'Cookie preferences'.

The screenshot shows the 'Code source' tab for the 'SqsToS3Function' in the AWS Lambda console. The 'Code source' tab is active, displaying a code editor with the following Python code:

```
1 import boto3
2 import json
3 import os
4
5 s3 = boto3.client('s3')
6
7 def handler(event, context):
8     bucket_name = os.environ['S3_BUCKET']
9     for record in event['Records']:
10         file_content = record['body']
11         filename = f"{record['messageId']}.txt"
12         s3.put_object(Bucket=bucket_name, Key=filename, Body=file_content)
13     return {
14         'statusCode': 200,
15         'body': json.dumps('Success')
16     }
17
```

The code editor has a 'Test' button and a 'Deploy' button. On the left side of the code editor, there is a file explorer showing the 'SqsToS3Function' folder and the 'lambda_function.py' file. The bottom of the console shows the 'CloudShell' and 'Feedback' buttons, along with the copyright notice '© 2024, Amazon Web Services, Inc. or its affiliates.' and links for 'Privacy', 'Terms', and 'Cookie preferences'.

The screenshot shows the AWS Lambda console for the function 'SqsToS3Function' in the 'us-east-1' region. The function is configured with an SQS trigger named 'mySQSQueue'. The function's ARN is 'arn:aws:lambda:us-east-1:9163855129:17:function:SqsToS3Function'.

Triggers (1) Info

Trigger
<input type="checkbox"/> SQS: mySQSQueue arn:aws:sqs:us-east-1:916385512917:mySQSQueue state: Enabled Details

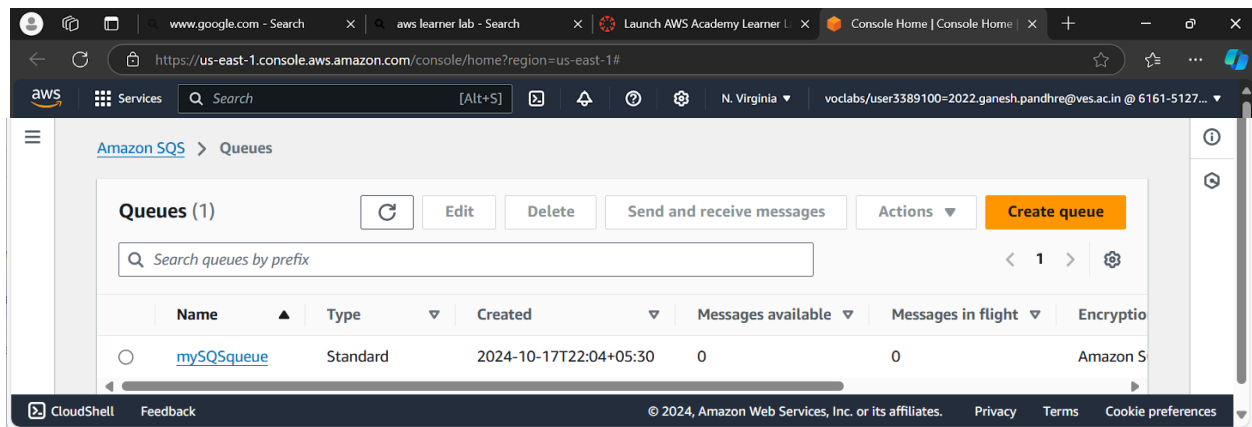
SQS queue

The screenshot shows the AWS Lambda console for the function 'SqsToS3Function' in the 'us-east-1' region. The function is configured with an SQS trigger named 'mySQSQueue'. The function's ARN is 'arn:aws:lambda:us-east-1:9163855129:17:function:SqsToS3Function'.

Environment variables (1)

The environment variables below are encrypted at rest with the default Lambda service key.

Key	Value
S3_BUCKET	fdp-accurately-suited-grouper



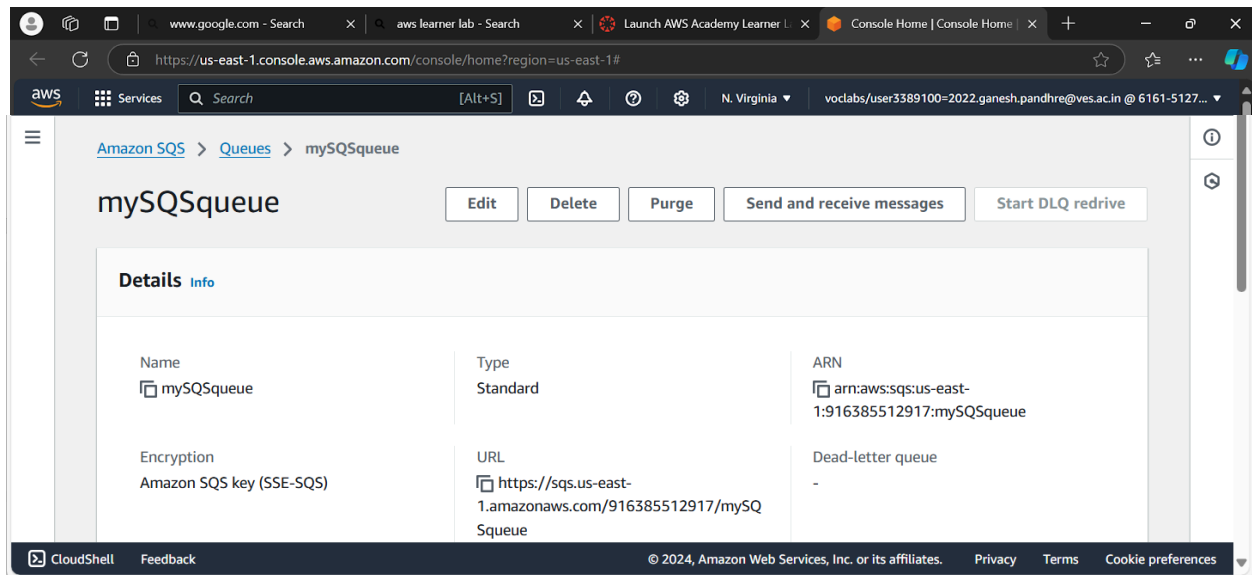
Amazon SQS > Queues

Queues (1) Refresh Edit Delete Send and receive messages Actions Create queue

Search queues by prefix

Name	Type	Created	Messages available	Messages in flight	Encryption
mySQSqueue	Standard	2024-10-17T22:04:05:30	0	0	Amazon S

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



Amazon SQS > Queues > mySQSqueue

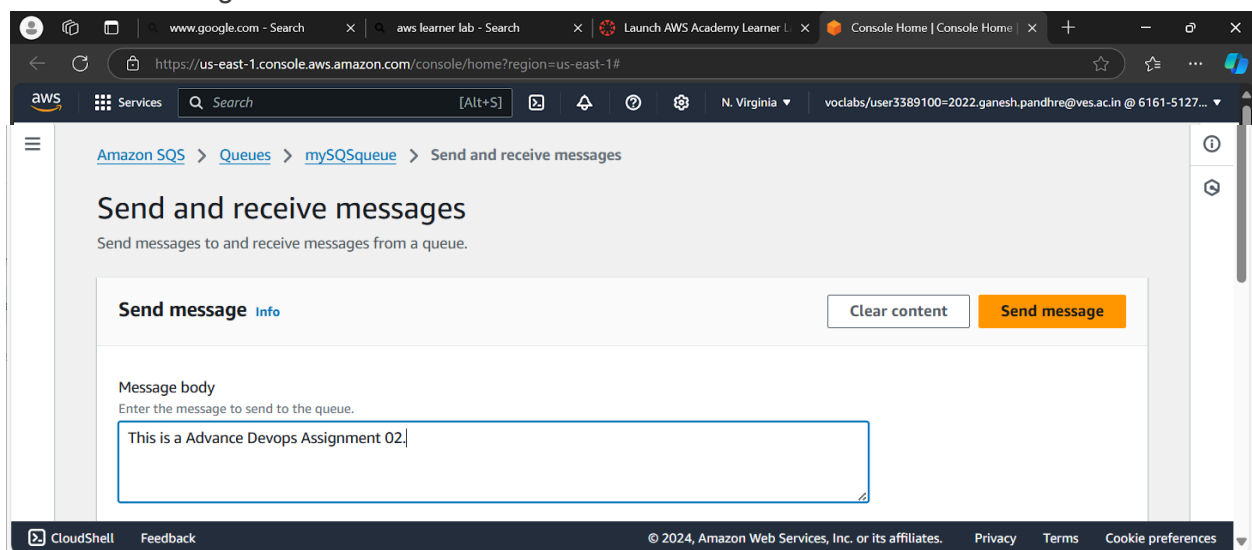
mySQSqueue Edit Delete Purge Send and receive messages Start DLQ redrive

Details [Info](#)

Name	Type	ARN
mySQSqueue	Standard	arn:aws:sqs:us-east-1:916385512917:mySQSqueue
Encryption	URL	Dead-letter queue
Amazon SQS key (SSE-SQS)	https://sqs.us-east-1.amazonaws.com/916385512917/mySQSqueue	-

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

Send the message from the SQS.



Amazon SQS > Queues > mySQSqueue > Send and receive messages

Send and receive messages

Send messages to and receive messages from a queue.

Send message [Info](#) Clear content Send message

Message body

Enter the message to send to the queue.

This is a Advance Devops Assignment 02.

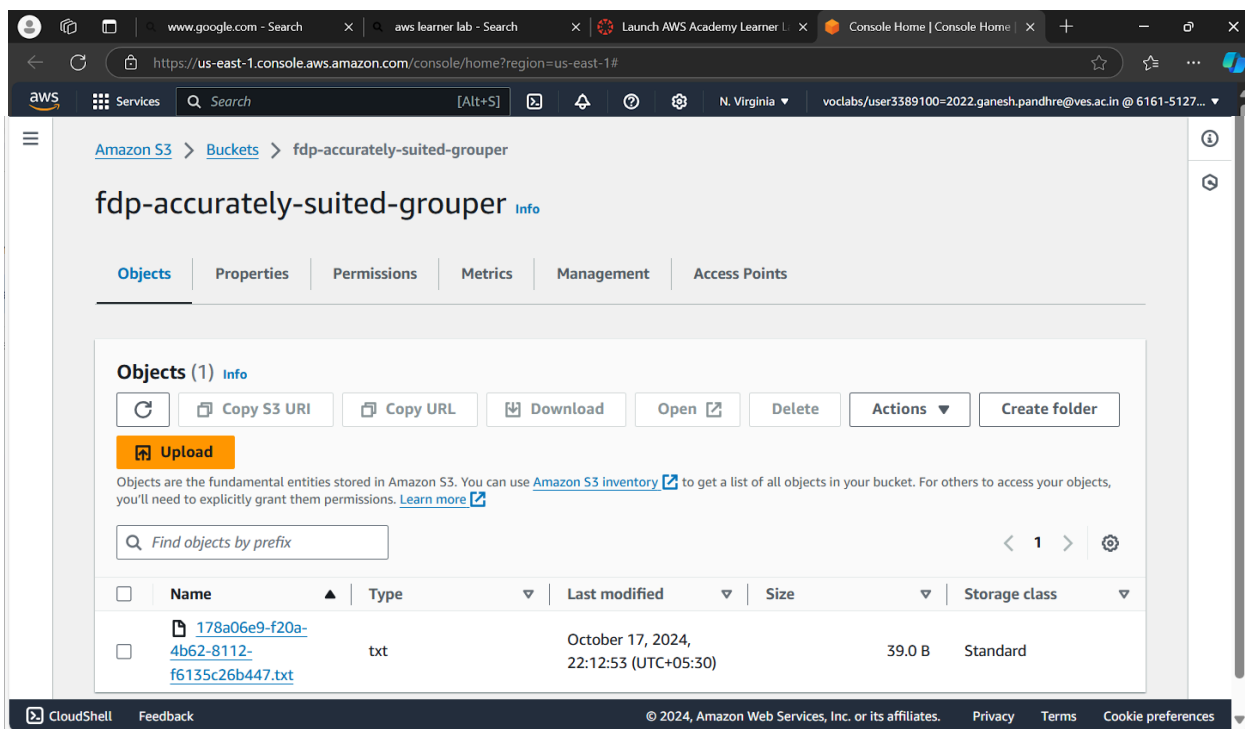
CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

The screenshot shows the AWS Management Console interface for the 'Send and receive messages' page of an Amazon SQS queue. The breadcrumb navigation is 'Amazon SQS > Queues > mySQSqueue > Send and receive messages'. The page title is 'Send and receive messages' with the subtitle 'Send messages to and receive messages from a queue.' There are two buttons at the top right: 'Clear content' and 'Send message'. A green success message box states: 'Your message has been sent and is ready to be received.' with a 'View details' link. Below this, the 'Message body' section has a text input field containing 'This is a Advance Devops Assignment 02.' The footer of the console shows '© 2024, Amazon Web Services, Inc. or its affiliates.' and links for 'Privacy', 'Terms', and 'Cookie preferences'.

S3 bucket

The screenshot shows the AWS Management Console interface for the 'General purpose buckets' page of Amazon S3. The breadcrumb navigation is 'Amazon S3 > General purpose buckets'. The page title is 'General purpose buckets (2)' with the subtitle 'Buckets are containers for data stored in S3.' There are several buttons at the top right: 'Copy ARN', 'Empty', 'Delete', and 'Create bucket'. A search bar is labeled 'Find buckets by name'. Below the search bar, there is a table with the following columns: 'Name', 'AWS Region', 'IAM Access Analyzer', and 'Creation date'. The table contains one entry: 'fdp-accurately-suited-grouper' in the 'US East (N. Virginia) us-east-1' region, with a link to 'View analyzer for us-east-1' and a creation date of 'October 17, 2024, 22:04:14 (UTC+05:30)'. The footer of the console shows '© 2024, Amazon Web Services, Inc. or its affiliates.' and links for 'Privacy', 'Terms', and 'Cookie preferences'.

Name	AWS Region	IAM Access Analyzer	Creation date
fdp-accurately-suited-grouper	US East (N. Virginia) us-east-1	View analyzer for us-east-1	October 17, 2024, 22:04:14 (UTC+05:30)



Amazon S3 > Buckets > fdp-accurately-suited-grouper

fdp-accurately-suited-grouper

Objects (1) Info

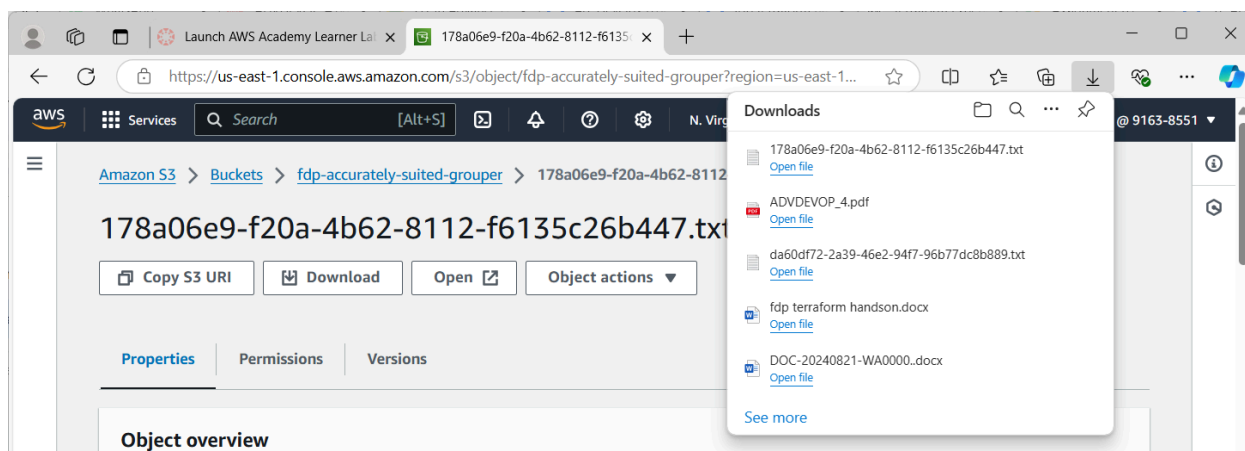
Copy S3 URI Copy URL Download Open Delete Actions Create folder

Upload

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

Name	Type	Last modified	Size	Storage class
178a06e9-f20a-4b62-8112-f6135c26b447.txt	txt	October 17, 2024, 22:12:53 (UTC+05:30)	39.0 B	Standard



Amazon S3 > Buckets > fdp-accurately-suited-grouper > 178a06e9-f20a-4b62-8112-f6135c26b447.txt

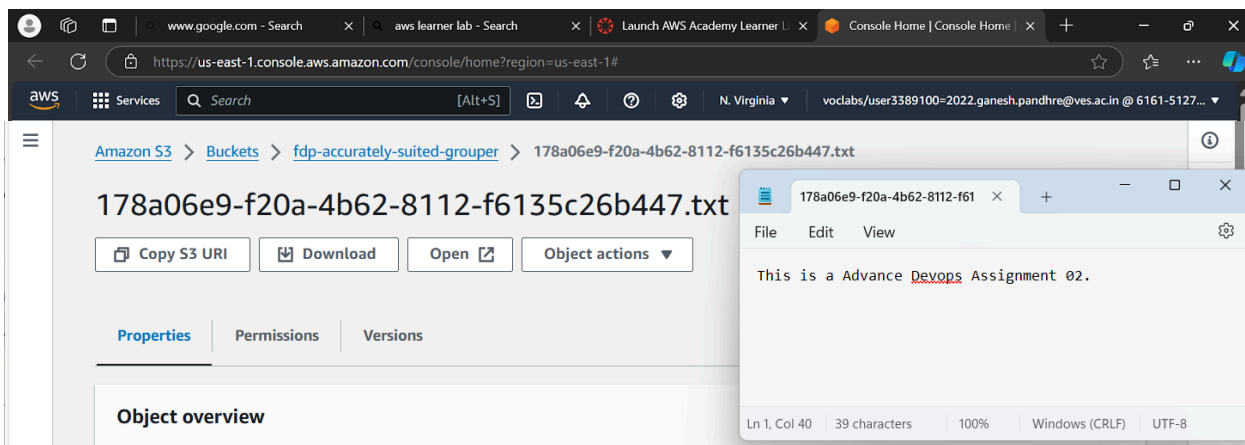
Copy S3 URI Download Open Object actions

Properties Permissions Versions

Object overview

Downloads

- 178a06e9-f20a-4b62-8112-f6135c26b447.txt
- ADVDEVOP_4.pdf
- da60df72-2a39-46e2-94f7-96b77dc8b889.txt
- fdp terraform handson.docx
- DOC-20240821-WA0000.docx



Amazon S3 > Buckets > fdp-accurately-suited-grouper > 178a06e9-f20a-4b62-8112-f6135c26b447.txt

Copy S3 URI Download Open Object actions

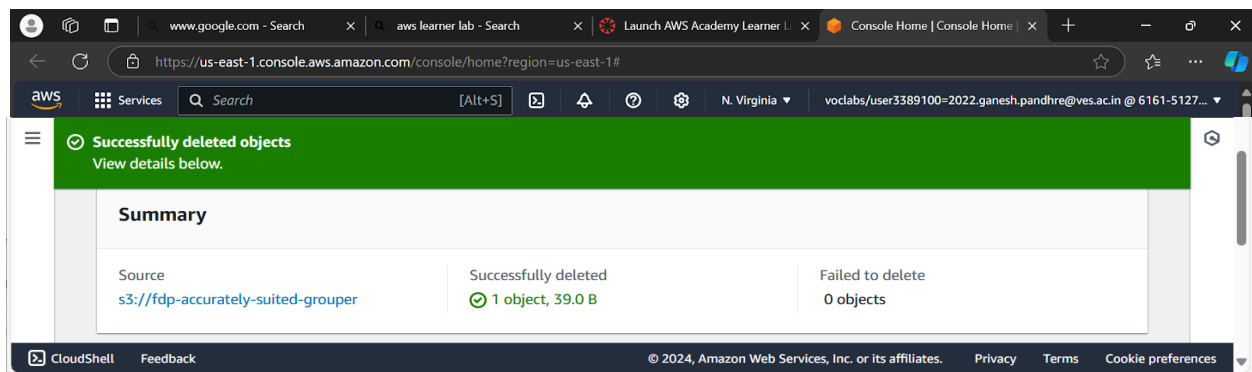
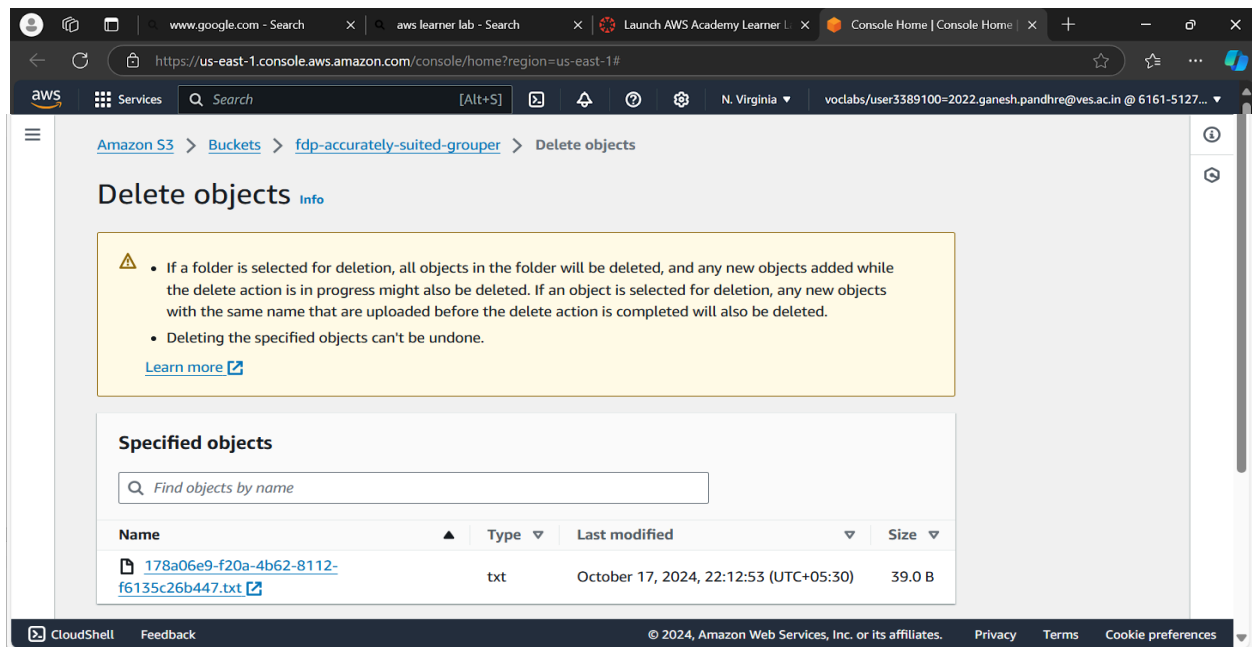
Properties Permissions Versions

Object overview

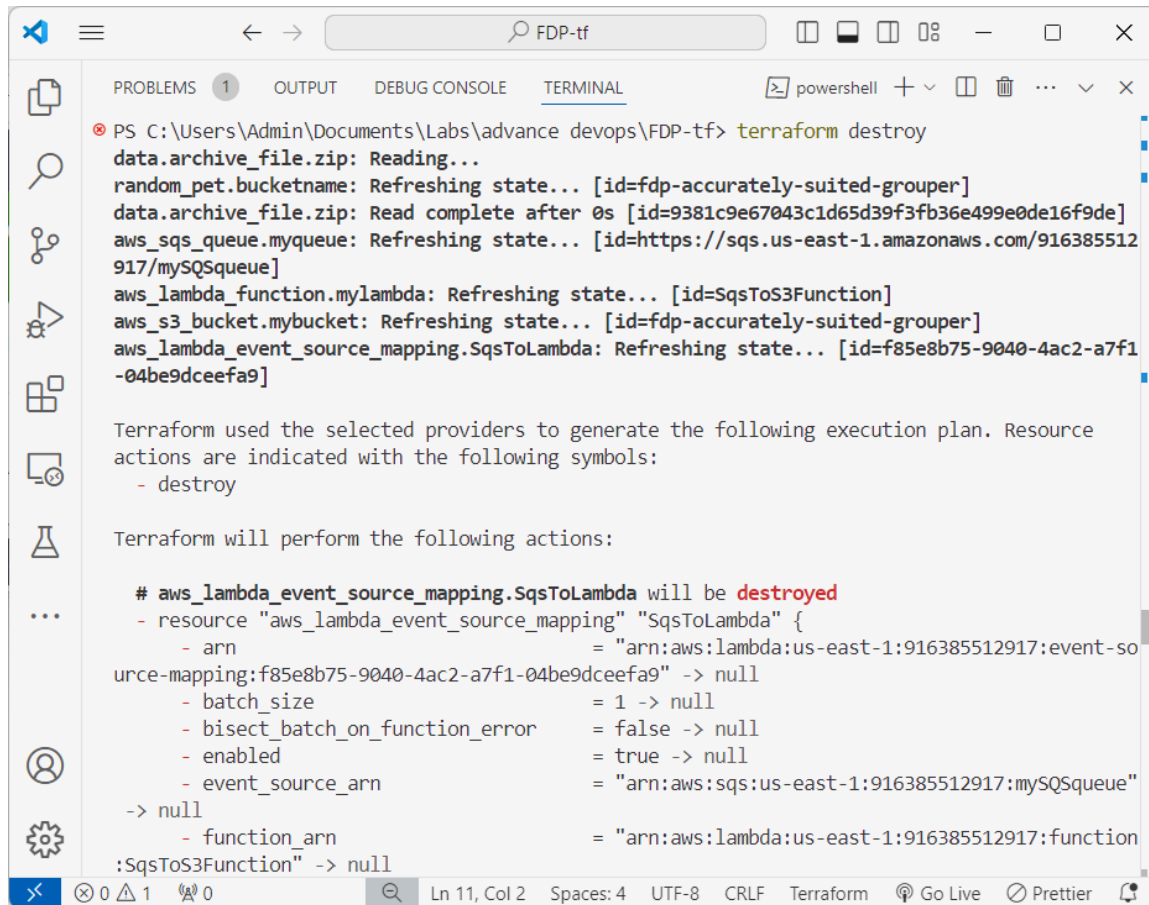
178a06e9-f20a-4b62-8112-f61

This is a Advance Devops Assignment 02.

Ln 1, Col 40 39 characters 100% Windows (CRLF) UTF-8



If you want to clean up the resources after testing, you can destroy them by running:
terraform destroy
(Confirm the destruction by typing **yes**.)



```

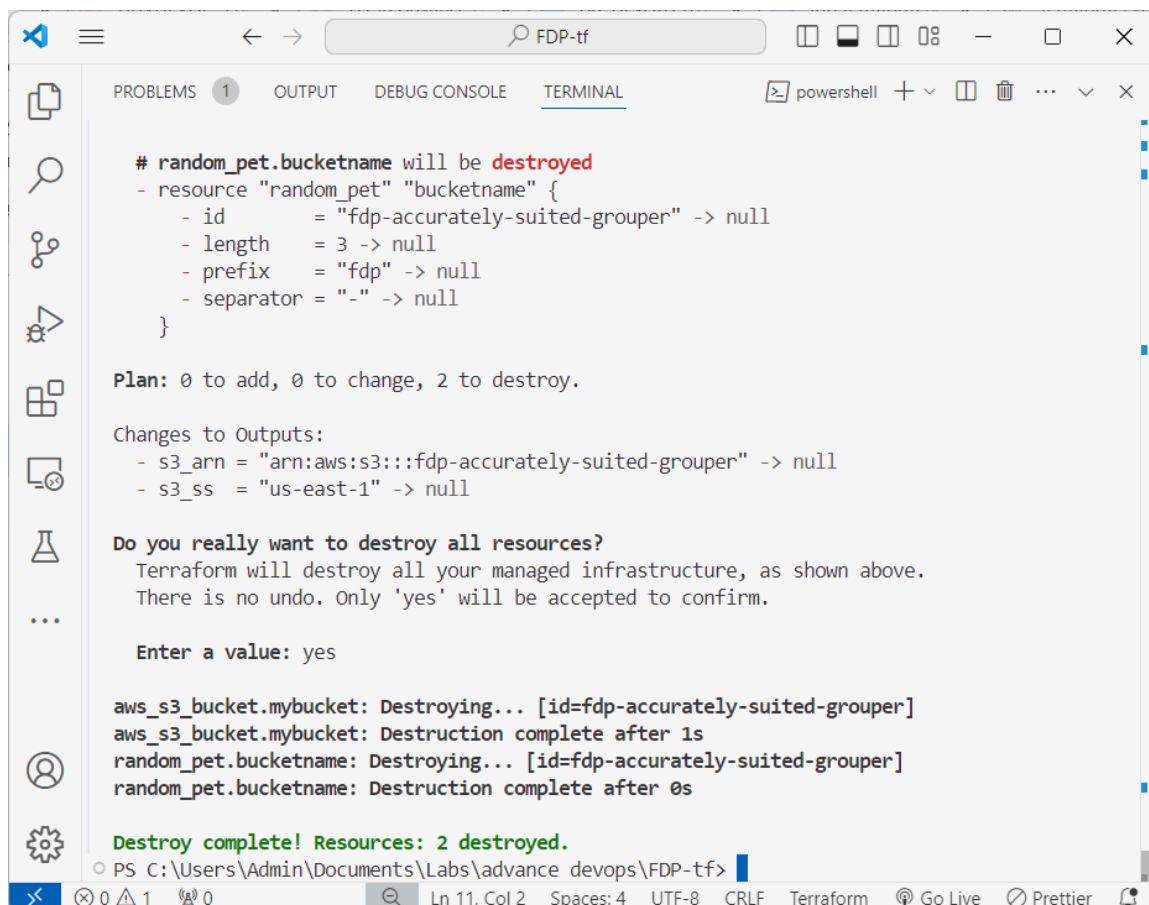
PS C:\Users\Admin\Documents\Labs\advance devops\FDP-tf> terraform destroy
data.archive_file.zip: Reading...
random_pet.bucketname: Refreshing state... [id=fdp-accurately-suited-grouper]
data.archive_file.zip: Read complete after 0s [id=9381c9e67043c1d65d39f3fb36e499e0de16f9de]
aws_sqs_queue.myqueue: Refreshing state... [id=https://sqs.us-east-1.amazonaws.com/916385512917/mySQSqueue]
aws_lambda_function.mylambda: Refreshing state... [id=SqsToS3Function]
aws_s3_bucket.mybucket: Refreshing state... [id=fdp-accurately-suited-grouper]
aws_lambda_event_source_mapping.SqsToLambda: Refreshing state... [id=f85e8b75-9040-4ac2-a7f1-04be9dceefa9]

Terraform used the selected providers to generate the following execution plan. Resource
actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

# aws_lambda_event_source_mapping.SqsToLambda will be destroyed
- resource "aws_lambda_event_source_mapping" "SqsToLambda" {
  - arn = "arn:aws:lambda:us-east-1:916385512917:event-so
    ource-mapping:f85e8b75-9040-4ac2-a7f1-04be9dceefa9" -> null
  - batch_size = 1 -> null
  - bisect_batch_on_function_error = false -> null
  - enabled = true -> null
  - event_source_arn = "arn:aws:sqs:us-east-1:916385512917:mySQSqueue"
    -> null
  - function_arn = "arn:aws:lambda:us-east-1:916385512917:function
    :SqsToS3Function" -> null

```



```

# random_pet.bucketname will be destroyed
- resource "random_pet" "bucketname" {
  - id = "fdp-accurately-suited-grouper" -> null
  - length = 3 -> null
  - prefix = "fdp" -> null
  - separator = "-" -> null
}

Plan: 0 to add, 0 to change, 2 to destroy.

Changes to Outputs:
  - s3_arn = "arn:aws:s3:::fdp-accurately-suited-grouper" -> null
  - s3_ss = "us-east-1" -> null

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

aws_s3_bucket.mybucket: Destroying... [id=fdp-accurately-suited-grouper]
aws_s3_bucket.mybucket: Destruction complete after 1s
random_pet.bucketname: Destroying... [id=fdp-accurately-suited-grouper]
random_pet.bucketname: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
PS C:\Users\Admin\Documents\Labs\advance devops\FDP-tf>

```

Conclusion

Deploying AWS infrastructure with Terraform and integrating S3, SQS, and Lambda creates robust and scalable cloud applications, essential for modern development.