



# 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.	42
Name	NAIKWADI YASH SHIVDAS
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="ASTIA5KXGMHKK2M5VBZ75"
eee_N_3428291@runweb148728:~$ export AWS_SECRET_ACCESS_KEY="Q+wyIVDy0SsG%myrAFzRHdH1annHIjBwTUMwM"
eee_N_3428291@runweb148728:~$ export AWS_SESSION_TOKEN="IQoJb3JpZ2luX2VjEj//////////wEaCXVzLXdlc3QtMiJHMEUCIDE+rHA
2+MrEv72wQUHw56L8+031TD1L1t+0J57VcYMAIEApAqRbKJNaGic31YuDn0bJezFhC84KjwQJrRVstI0qtGITMRABGwSMYz0DUIMTISMTciDK
10eZ1lneyFH4CzSxqTat8nQmN8rxHkxQFsjY02Ncma6JaC1VxD+LAsrr5usRTK4j4+1bwj22jd4J/YX41kwiCXUeKEJNRkpZjropv4Hfbue7
TN21Ryq2p0S5eiekDQ1j8th511C7r9F5x0mcBP0R1FcY3jzj6k60pRyM+vJ4AobtdgV08PU+CGgC6UV/K3/yzHeB+ZYvY
gwgNu46BdwvTAK9P5PCFUSnzaorITj0d03b1DapFRk9AMQFQ0kHx+R1HemJ1HwkE9Ij8a9J4ckVhIdR3EmdCFQV1833so31k0ktsCQCDPp
a312y1A9MTSb01vY1K19p+Q8tM1X2mUB1mSM3e4caRR0YmTvUkM0LnxLgG0p0B+00S7HmIV9JRq1asc5j6
ofZpq7q484wZn1mMp403Hzfndp2W8U001cw7XDA64
8h0P6NbYYx93JhzKKEN14D8Epm1Rf3gD8o/Ne5g
Djj9bV2YG0rRDeONFco31UYSF9pnXWb71okZWJZj
t/E+ITIXJibpumVHs4aAAWIpj6ZFwkbCbX/BWVZW
Efa2j+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 x aws learner lab - Search x Launch AWS Academy Learner l x Home | EC2 | us-east-1 x + -

https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Home:

aws Services Search [Alt+S] N. Virginia voclabs/user3389086=2022.yash.naikwadi@ves.ac.in @ 0805-4726-...

Roles (21) Info

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

Role name	Trusted entities	Last activity
LabRole	Account: 916385512917	12 hours ago

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

www.google.com - Search x aws learner lab - Search x Launch AWS Academy Learner l x Home | EC2 | us-east-1 x + -

https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Home:

aws Services Search [Alt+S] N. Virginia voclabs/user3389086=2022.yash.naikwadi@ves.ac.in @ 0805-4726-...

IAM > Roles > LabRole

LabRole Info

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

Delete Edit

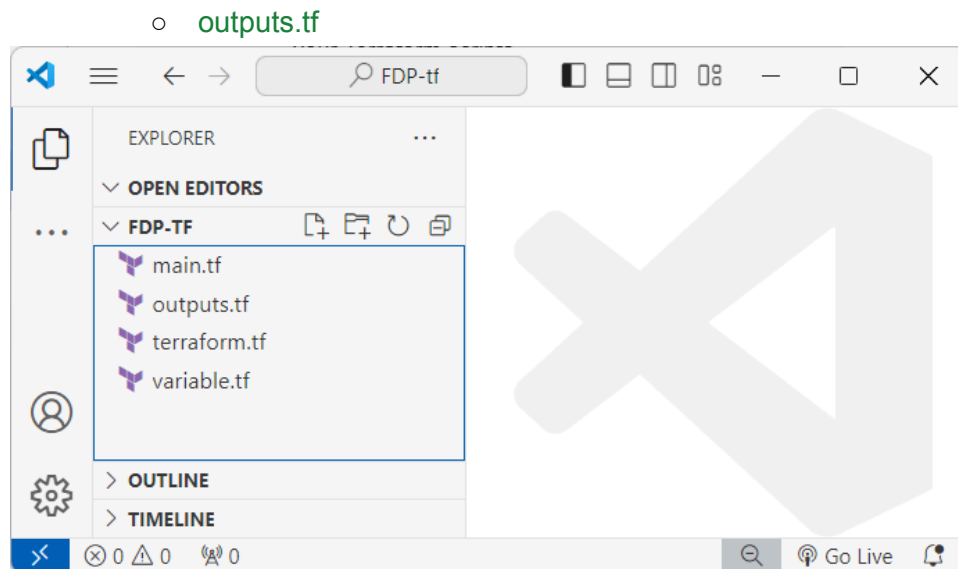
CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

### 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.

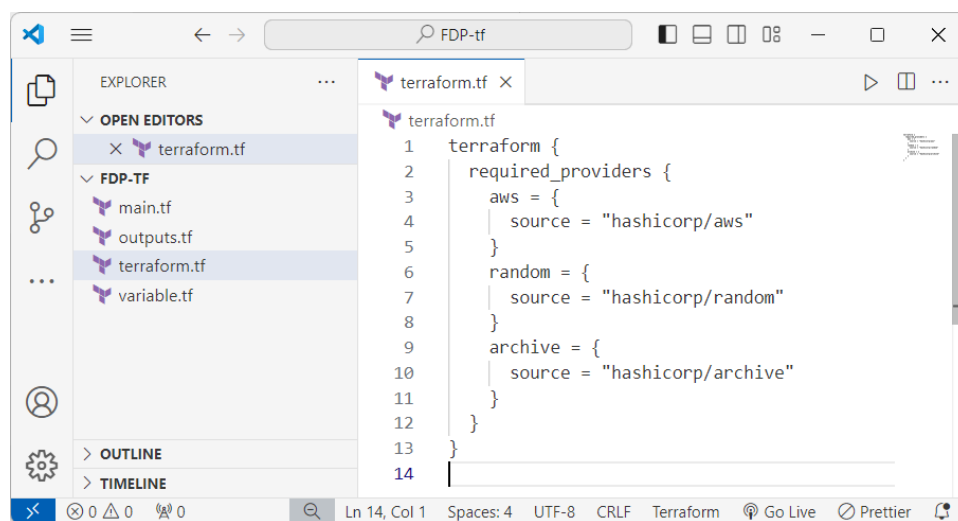
### Set Up Terraform Configuration:

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



### terraform.tf

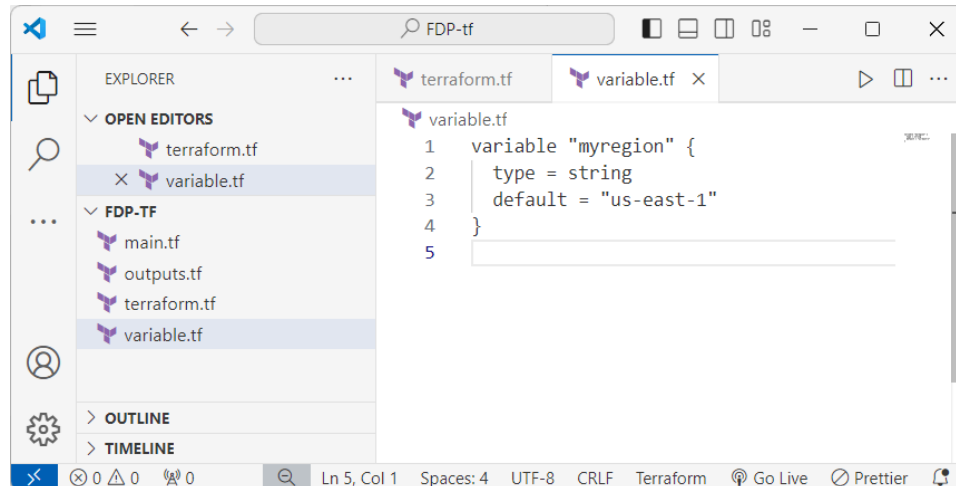
```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
    }
    random = {
      source = "hashicorp/random"
    }
    archive = {
      source = "hashicorp/archive"
    }
  }
}
```



### 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 = "ASIA5KXGVHXXK2M5YBZ75"
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

```

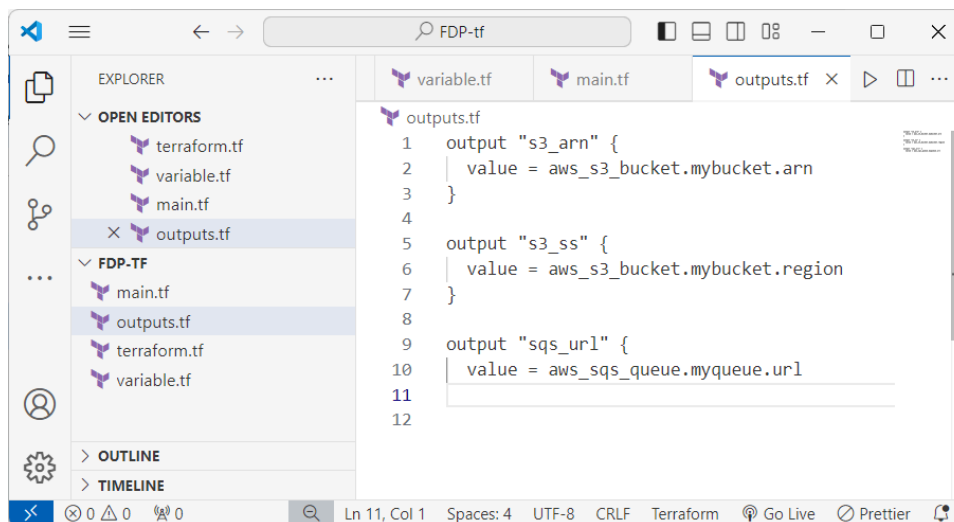
output "s3_arn" {
  value = aws_s3_bucket.mybucket.arn
}

```

```
}

output "s3_ss" {
  value = aws_s3_bucket.mybucket.region
}

output "sqs_url" {
  value = aws_sqs_queue.myqueue.url
}
```



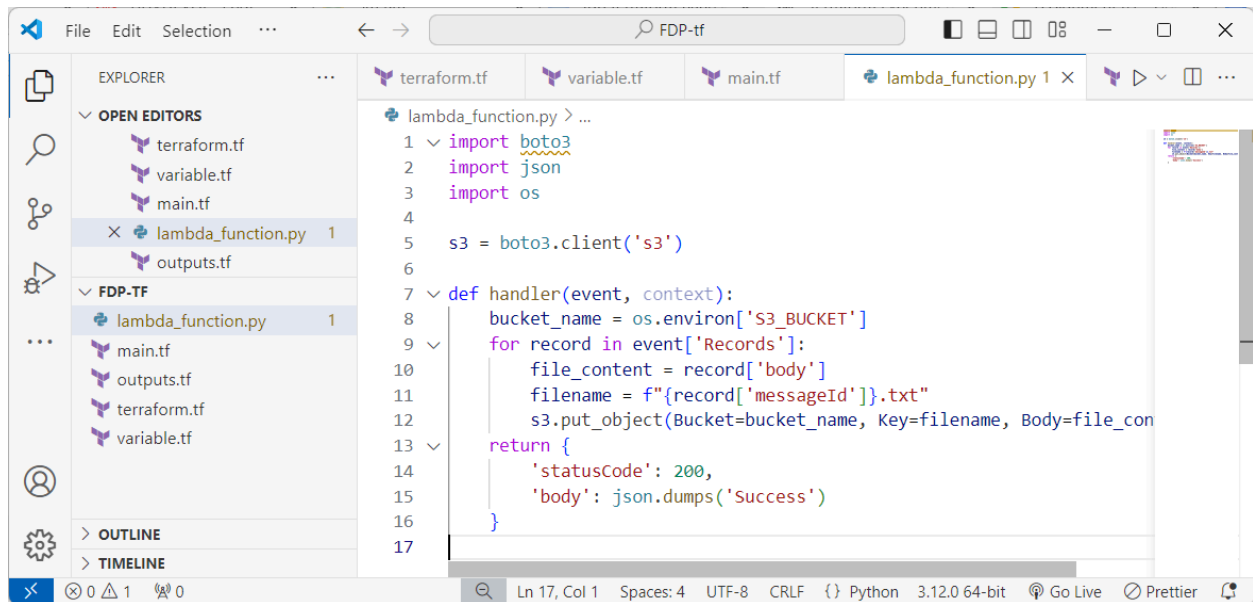
### Create Lambda Python File:

- 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')
    }
```

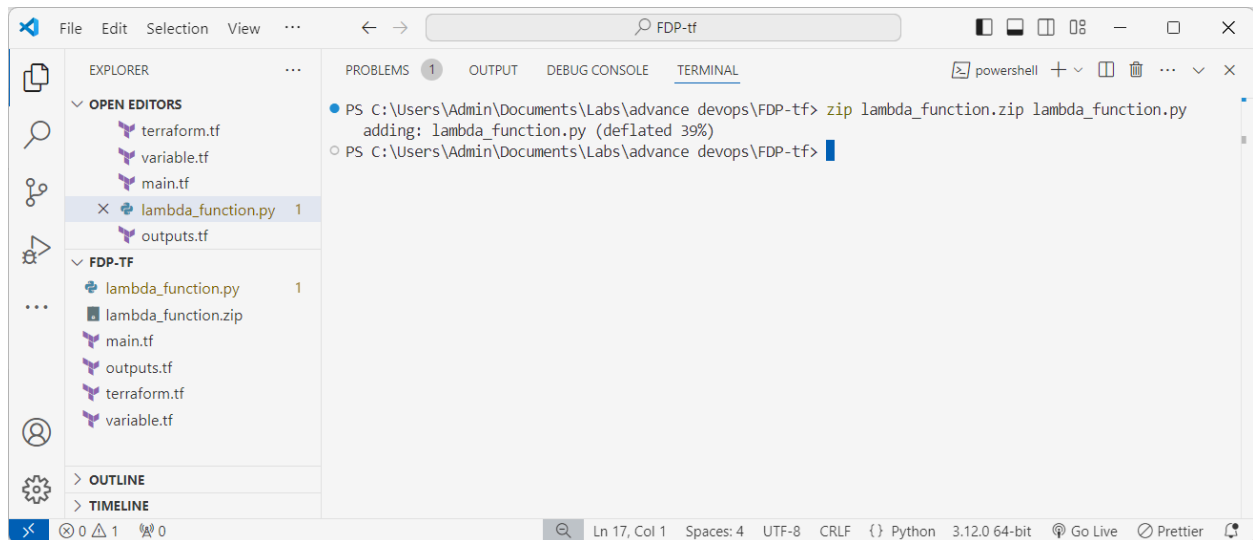


The screenshot shows the Visual Studio Code editor with the file `lambda_function.py` open. The Explorer sidebar on the left shows the project structure with files `terraform.tf`, `variable.tf`, `main.tf`, `lambda_function.py`, and `outputs.tf`. The main editor area displays the following Python code:

```
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
```

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

`zip lambda_function.zip lambda_function.py`



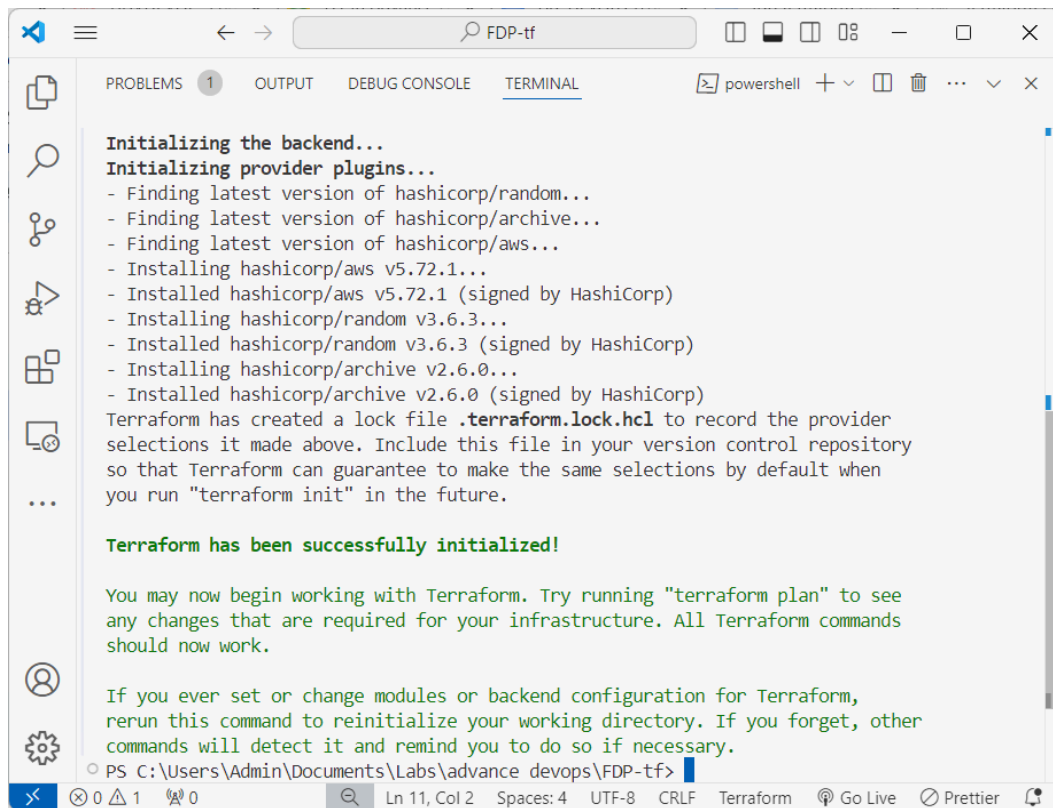
The screenshot shows the Visual Studio Code editor with the terminal window open. The terminal displays the output of the command `zip lambda_function.zip lambda_function.py`:

```
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.

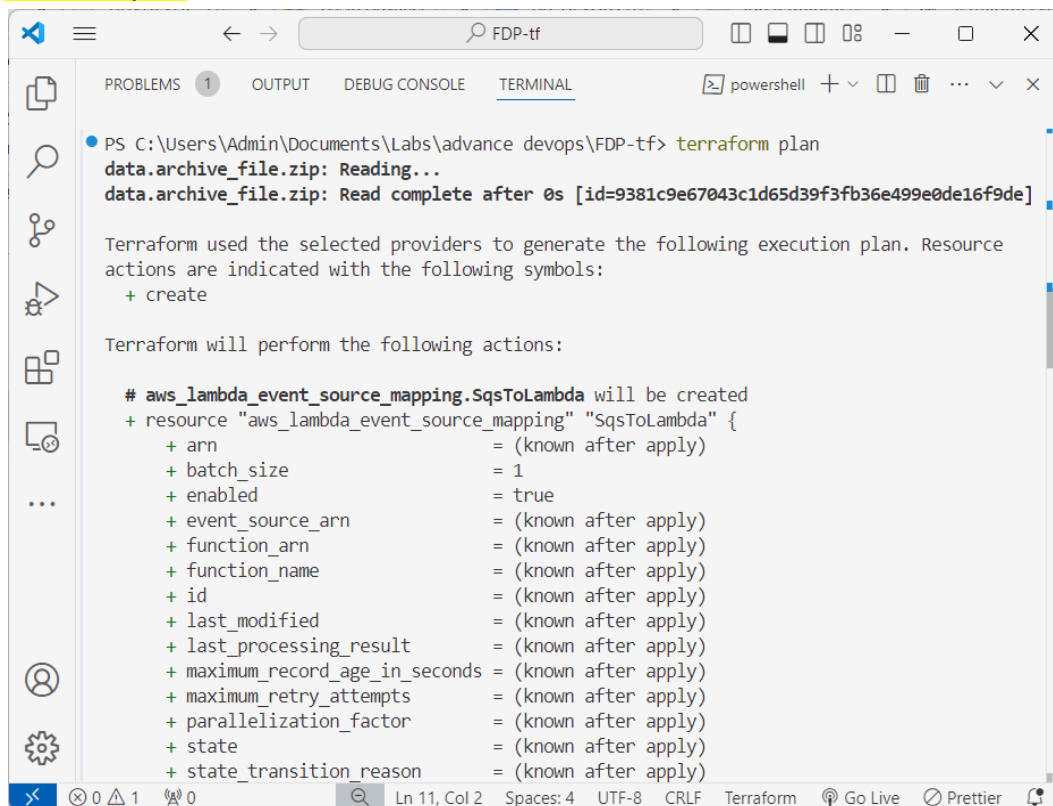
Terraform has been successfully initialized!

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**

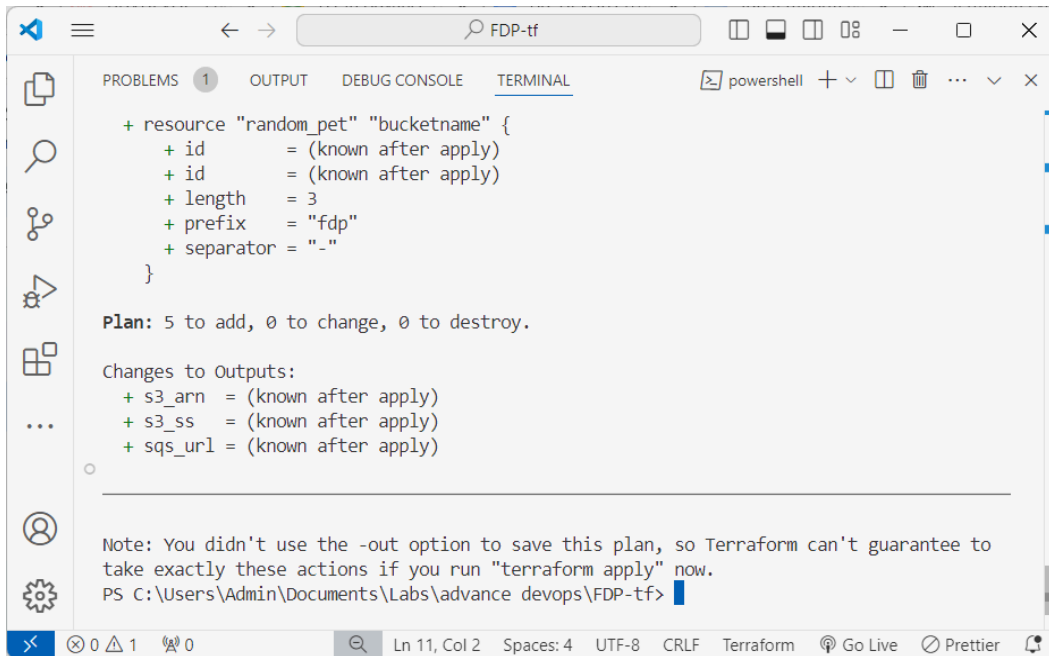


```
PS C:\Users\Admin\Documents\Labs\advance devops\FDP-tf> terraform plan
data.archive_file.zip: Reading...
data.archive_file.zip: Read complete after 0s [id=9381c9e67043c1d65d39f3fb36e499e0de16f9de]

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.

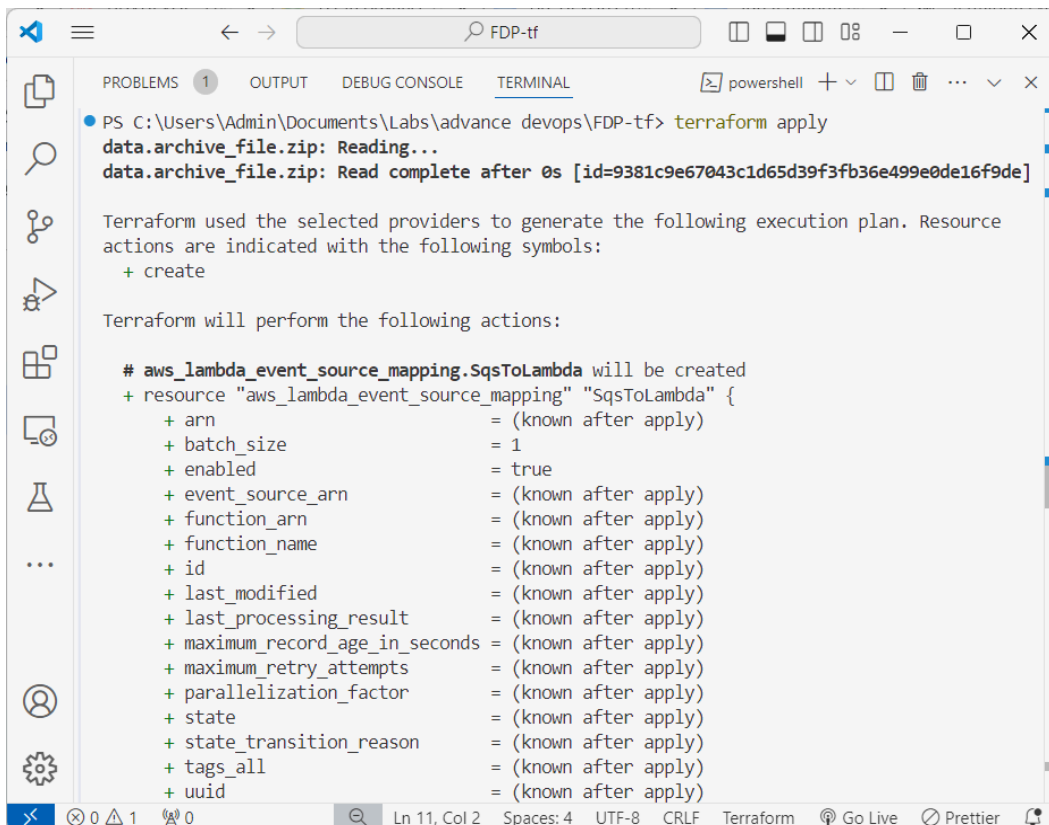
Changes to Outputs:
+ s3_arn = (known after apply)
+ s3_ss  = (known after apply)
+ sqs_url = (known after apply)

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.)

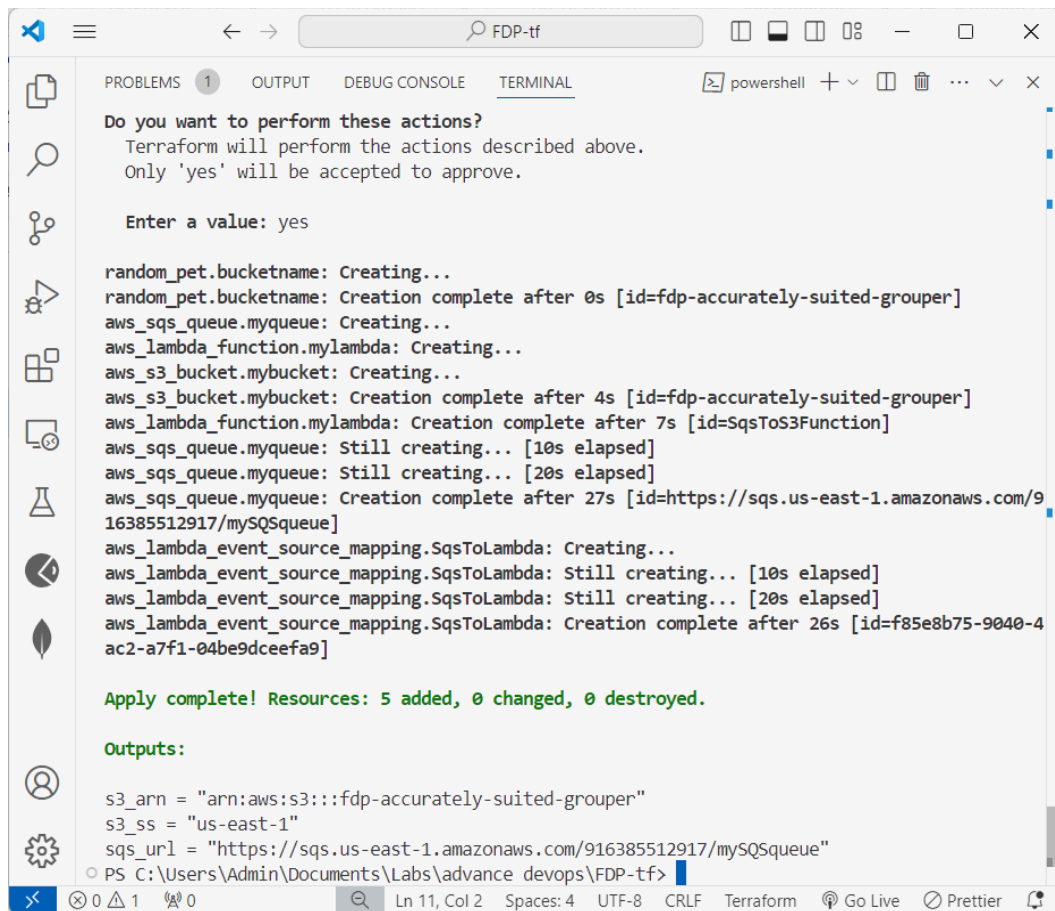


```
PS C:\Users\Admin\Documents\Labs\advance devops\FDP-tf> terraform apply
data.archive_file.zip: Reading...
data.archive_file.zip: Read complete after 0s [id=9381c9e67043c1d65d39f3fb36e499e0de16f9de]

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.

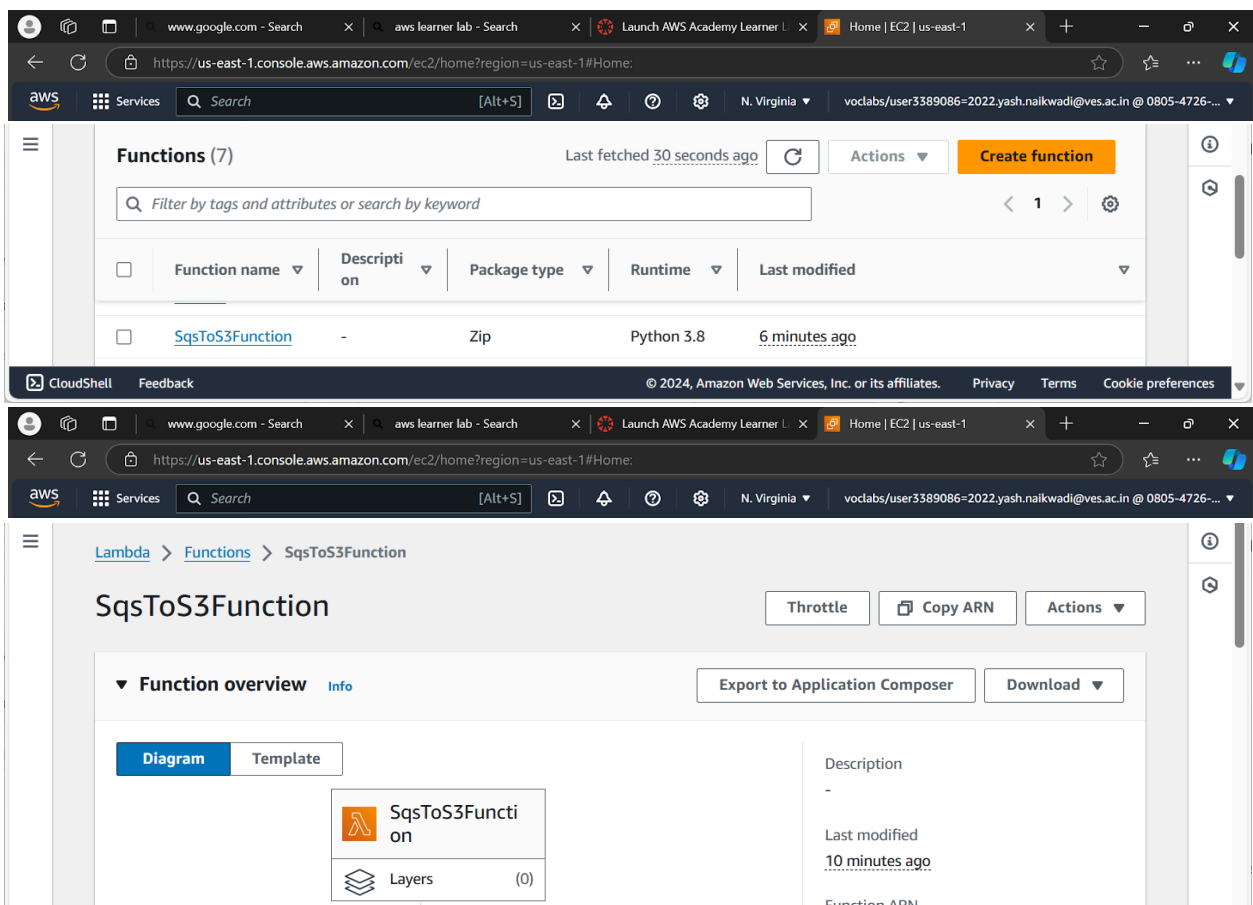
Outputs:

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 Management Console interface. The top navigation bar includes the AWS logo, a search bar, and the user's profile. The main content area displays the 'Functions (7)' page, which lists the created Lambda function 'SqsToS3Function' with a package type of 'Zip' and a runtime of 'Python 3.8'. Below this, the 'Function overview' section provides details about the function, including its description, last modified time, and function ARN.

Function name	Description	Package type	Runtime	Last modified
SqsToS3Function	-	Zip	Python 3.8	6 minutes ago

**SqsToS3Function**

Throttle Copy ARN Actions

Function overview Info

Export to Application Composer Download

Diagram Template

SqsToS3Function

Layers (0)

Description

Last modified: 10 minutes ago

Function ARN

The screenshot shows the AWS Lambda console for the function 'SqsToS3Function'. The 'Code source' tab is selected, displaying the Python code for the lambda function. The code imports boto3, json, and os, and uses boto3 to interact with S3. It defines a handler function that processes records from an S3 event and uploads them to a bucket.

```
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
```

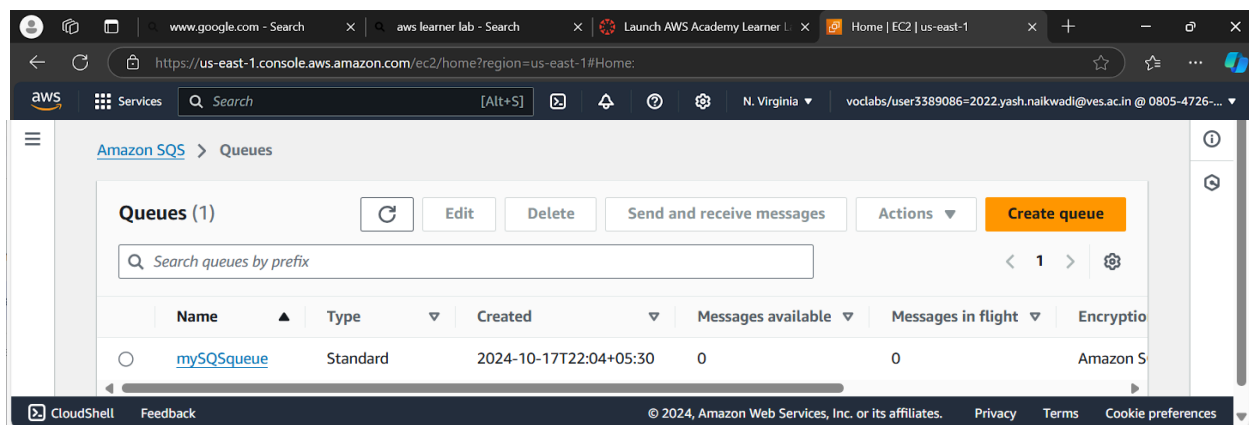
The screenshot shows the AWS Lambda console for the function 'SqsToS3Function'. The 'Configuration' tab is selected, displaying the function's configuration. The 'Triggers' section shows a single trigger named 'mySQSqueue' of type 'SQS'.

**Triggers (1)**

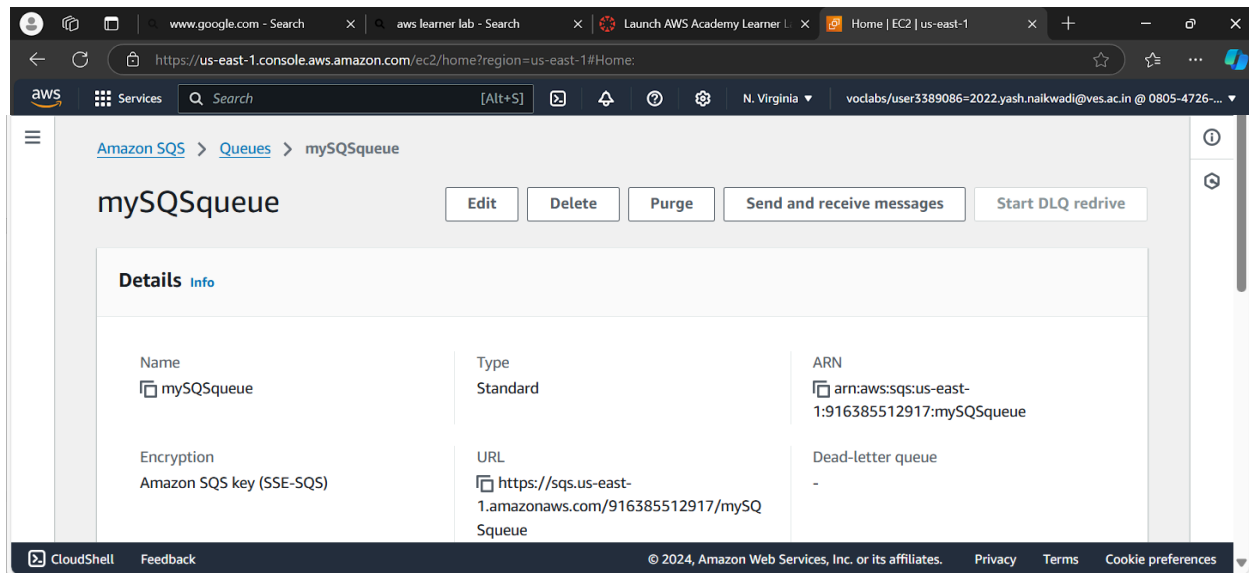
Trigger
<input type="checkbox"/> <b>SQS: mySQSqueue</b> arn:aws:sqs:us-east-1:916385512917:mySQSqueue state: <b>Enabled</b> <a href="#">Details</a>

## SQS queue

This screenshot is identical to the previous one, showing the AWS Lambda console for the function 'SqsToS3Function' with the 'Configuration' tab selected. It displays the function's configuration, including the 'Triggers' section with the 'mySQSqueue' trigger.

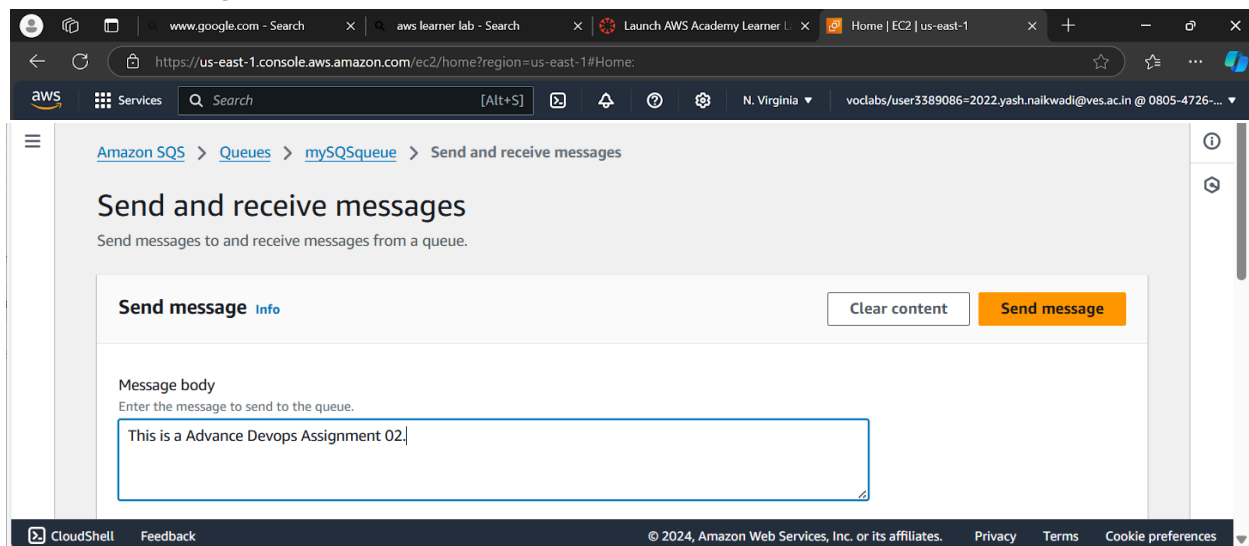


The screenshot shows the AWS Management Console 'Queues' page. At the top, there's a search bar and a 'Create queue' button. Below, a table lists the queues. The 'mySQSQueue' is the only one shown, with a 'Standard' type, created on 2024-10-17T22:04:05:30, and 0 messages available and 0 in flight. The console footer shows '© 2024, Amazon Web Services, Inc. or its affiliates.'

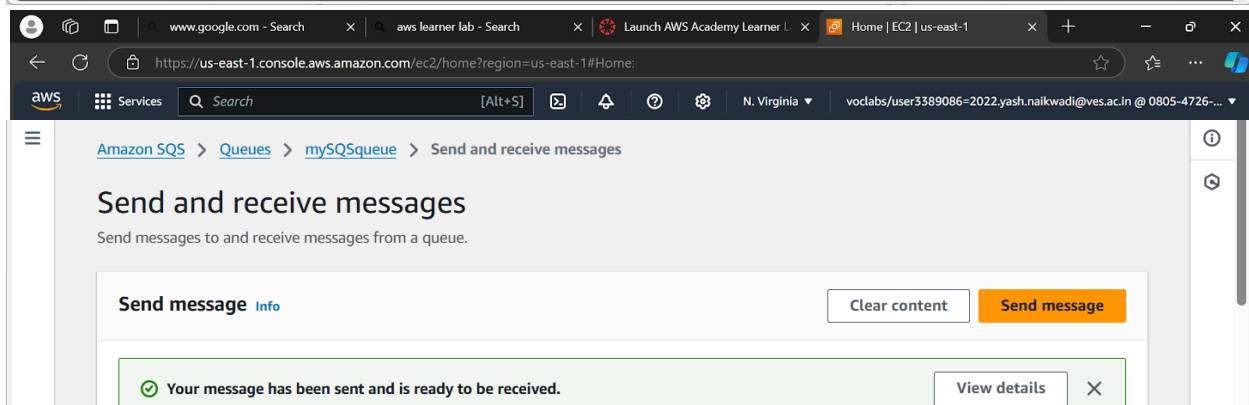


The screenshot shows the 'mySQSQueue' details page. It includes buttons for 'Edit', 'Delete', 'Purge', 'Send and receive messages', and 'Start DLQ redrive'. The 'Details' section shows the queue's name, type, ARN, encryption status, and URL. The console footer shows '© 2024, Amazon Web Services, Inc. or its affiliates.'

Send the message from the SQS.



The screenshot shows the 'Send and receive messages' page for 'mySQSQueue'. It includes a 'Send message' button and a text input field for the message body. The message body is 'This is a Advance Devops Assignment 02.'. The console footer shows '© 2024, Amazon Web Services, Inc. or its affiliates.'



The screenshot shows the 'Send and receive messages' page for 'mySQSQueue'. A green notification bar at the bottom states: 'Your message has been sent and is ready to be received.' The console footer shows '© 2024, Amazon Web Services, Inc. or its affiliates.'

## S3 bucket

Amazon S3

Account snapshot - updated every 24 hours All AWS Regions View Storage Lens dashboard

Storage lens provides visibility into storage usage and activity trends. [Learn more](#)

General purpose buckets Directory buckets

General purpose buckets (2) Info All AWS Regions Copy ARN Empty Delete Create bucket

Buckets are containers for data stored in S3.

Find buckets by name

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

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

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

fdp-accurately-suited-grouper Info

Objects Properties Permissions Metrics Management Access Points

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
<a href="#">178a06e9-f20a-4b62-8112-f6135c26b447.txt</a>	txt	October 17, 2024, 22:12:53 (UTC+05:30)	39.0 B	Standard

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

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

178a06e9-f20a-4b62-8112-f6135c26b447.txt

Copy S3 URI Download Open Object actions

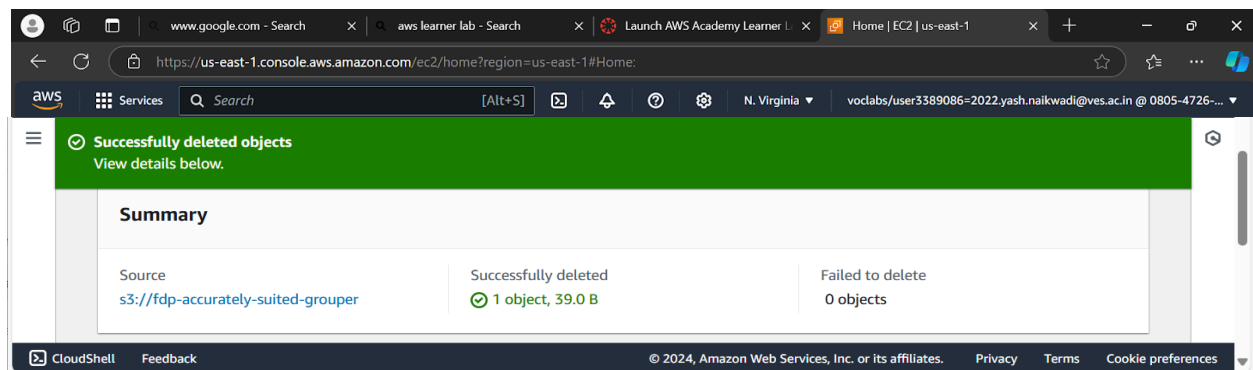
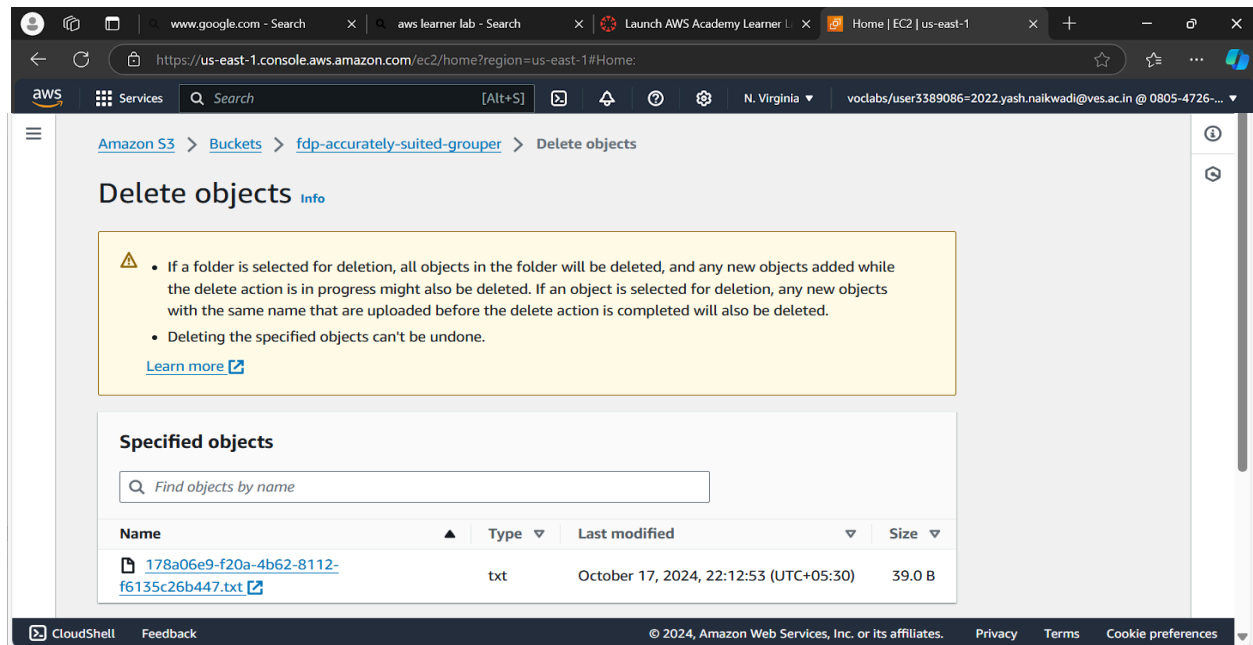
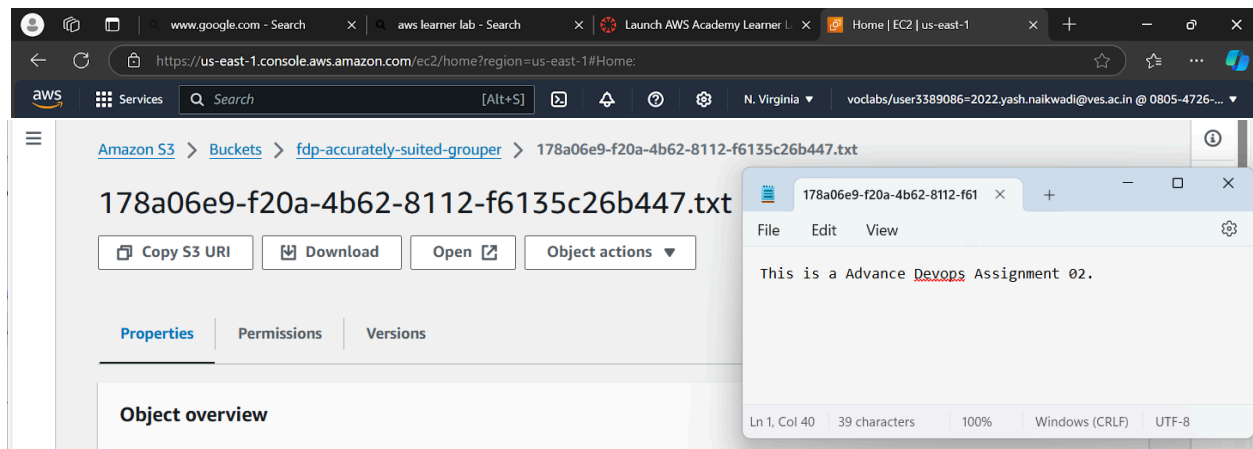
Properties Permissions Versions

Object overview

Downloads

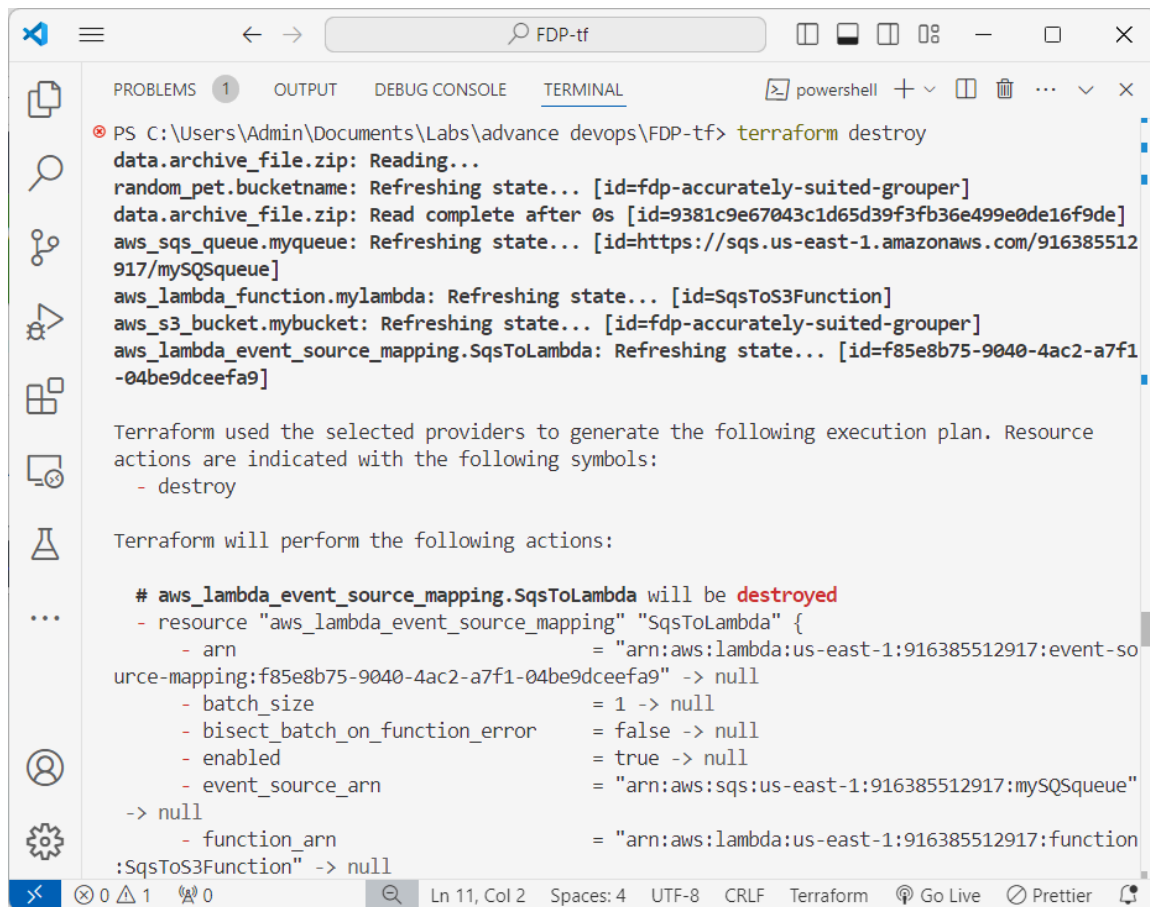
- 178a06e9-f20a-4b62-8112-f6135c26b447.txt [Open file](#)
- ADVDEVOP\_4.pdf [Open file](#)
- da60df72-2a39-46e2-94f7-96b77dc8b889.txt [Open file](#)
- fdp terraform handson.docx [Open file](#)
- DOC-20240821-WA0000.docx [Open file](#)

See more



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





```

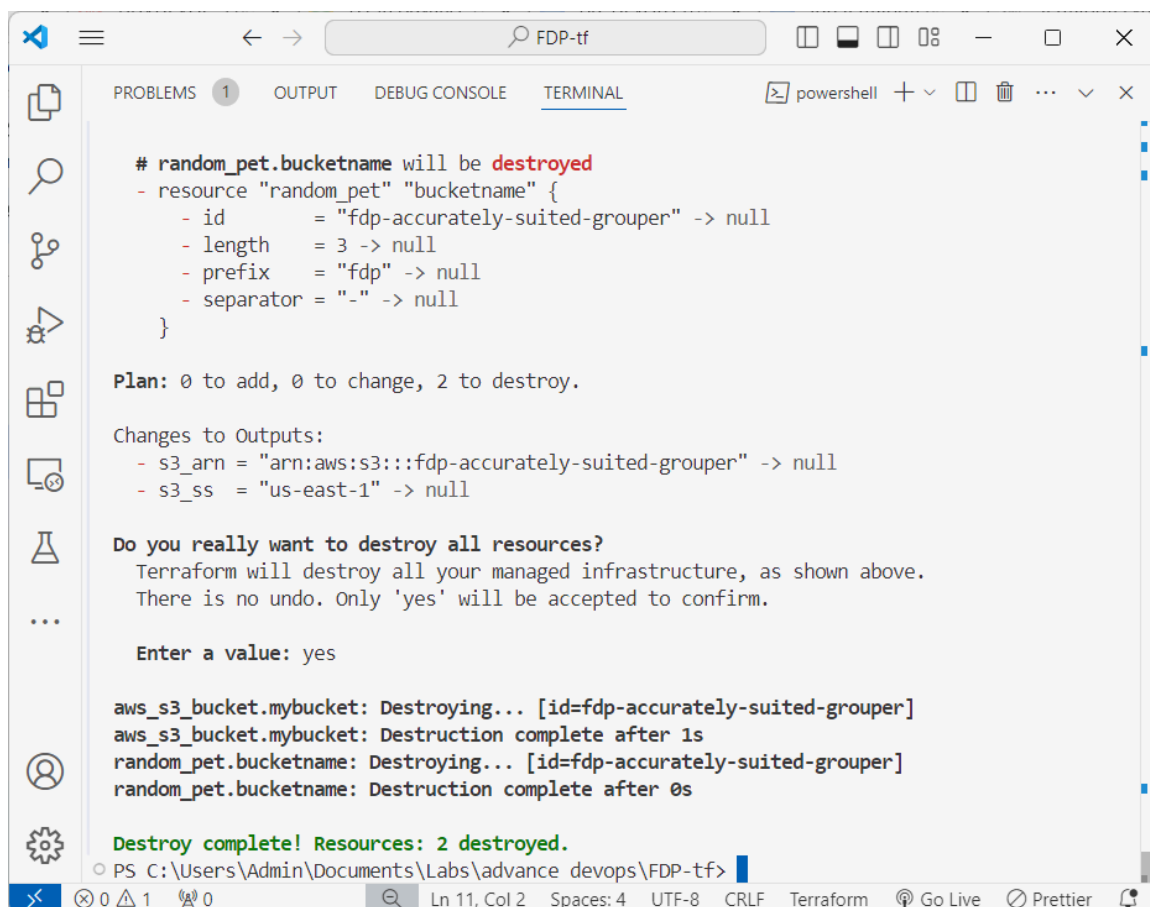
PS C:\Users\Admin\Documents\Labs\advance devops\FDP-tf> terraform destroy
data.archive_file.zip: Reading...
random_pet.bucketname: Refreshing state... [id=fdp-accurately-suited-grouper]
data.archive_file.zip: Read complete after 0s [id=9381c9e67043c1d65d39f3fb36e499e0de16f9de]
aws_sqs_queue.myqueue: 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.