Advance Devops-12

Aim:To create a Lambda function which will log "An Image has been added" once you add an object to a specific bucket in S3

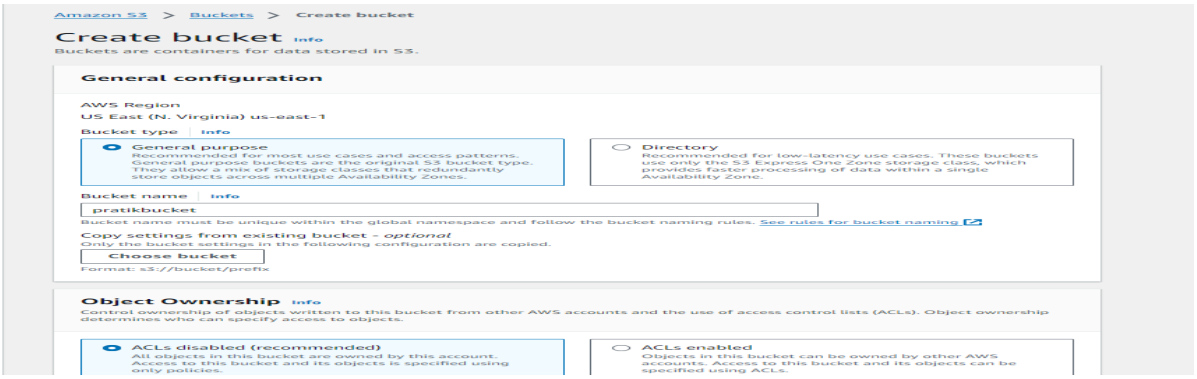
Theory:

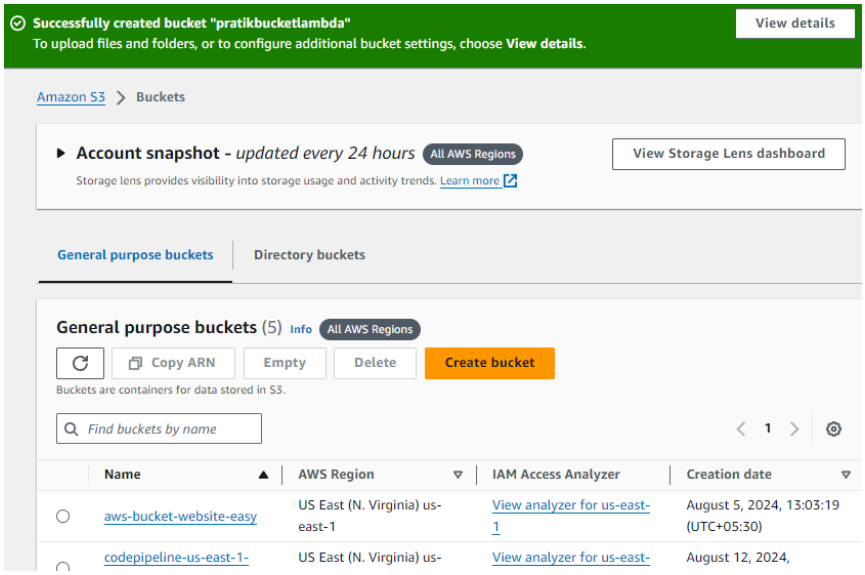
AWS Lambda and S3 Integration: AWS Lambda allows you to execute code in response to various events, including those triggered by Amazon S3. When an object is added to an S3 bucket, it can trigger a Lambda function to execute, allowing for event-driven processing without managing servers.

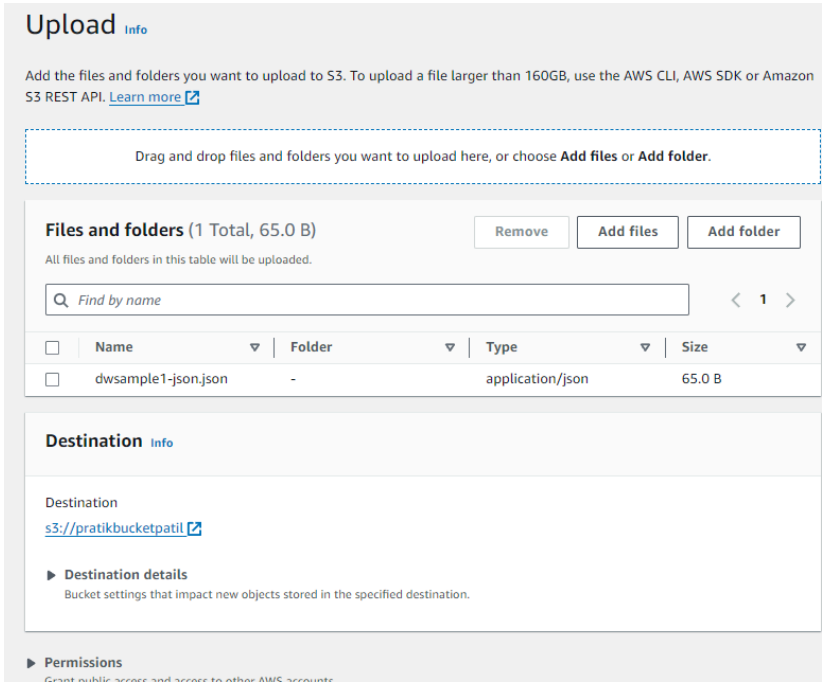
Workflow:

1. **Create an S3 Bucket:**
   - First, create an S3 bucket that will store the objects. This bucket will act as the trigger source for the Lambda function.
2. **Create the Lambda Function:**
   - Set up a new Lambda function using AWS Lambda's console. You can choose a runtime environment like Python, Node.js, or Java.
   - Write code that logs a message like "An Image has been added" when triggered.
3. **Set Up Permissions:**
   - Ensure that the Lambda function has the necessary permissions to access S3. You can do this by attaching an IAM role with policies that allow reading from the bucket and writing logs to CloudWatch.
4. **Configure S3 Trigger:**
   - Link the S3 bucket to the Lambda function by setting up a trigger. Specify that the function should be triggered when an object is created in the bucket (e.g., when an image is uploaded).
5. **Test the Setup:**
   - Upload an object (e.g., an image) to the S3 bucket to test the trigger. The Lambda function should execute and log the message "An Image has been added" in AWS CloudWatch Logs.

Outcomes:




Successfully created bucket "pratikbucketlambda"
To upload files and folders, or to configure additional bucket settings, choose View details.
View details
Amazon S3 > Buckets
Account snapshot - updated every 24 hours
All AWS Regions
Storage lens provides visibility into storage usage and activity trends. Learn more
View Storage Lens dashboard
General purpose buckets
Directory buckets
General purpose buckets (5) Info
All AWS Regions
Copy ARN
Empty
Delete
Create bucket
Buckets are containers for data stored in S3.
Find buckets by name
1
Name
AWS Region
IAM Access Analyzer
Creation date
aws-bucket-website-easy
US East (N. Virginia) us-east-1
View analyzer for us-east-1
August 5, 2024, 13:03:19 (UTC+05:30)
codepipeline-us-east-1-
US East (N. Virginia) us-
View analyzer for us-east-
August 12, 2024,


Upload Info
Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more
Drag and drop files and folders you want to upload here, or choose Add files or Add folder.
Files and folders (1 Total, 65.0 B)
Remove
Add files
Add folder
All files and folders in this table will be uploaded.
Find by name
1
Name
Folder
Type
Size
dwsample1-json.json
-
application/json
65.0 B
Destination Info
Destination
s3://pratikbucketpatil
Destination details
Bucket settings that impact new objects stored in the specified destination.
Permissions

Upload Info

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more

Drag and drop files and folders you want to upload here, or choose **Add files** or **Add folder**.

Files and folders (1 Total, 65.0 B)

Remove | Add files | Add folder

All files and folders in this table will be uploaded.

Find by name

1

Name	Folder	Type	Size
dwsample1-json.json	-	application/json	65.0 B

Destination Info

Destination

s3://pratikbucketpatil

Destination details

Bucket settings that impact new objects stored in the specified destination.

Permissions

Grant public access and access to other AWS accounts

Data

Data source: AwsDataCatalog

Database: default

Tables and views | Create

Filter tables and views

Tables (0) 1

Views (0) 1

Query 1

```
1  CREATE DATABASE IF NOT EXISTS mydatabase;
2
```

SQL Ln 2, Col 1

Run again | Explain | Cancel | Reuse query results up to 60 minutes ago

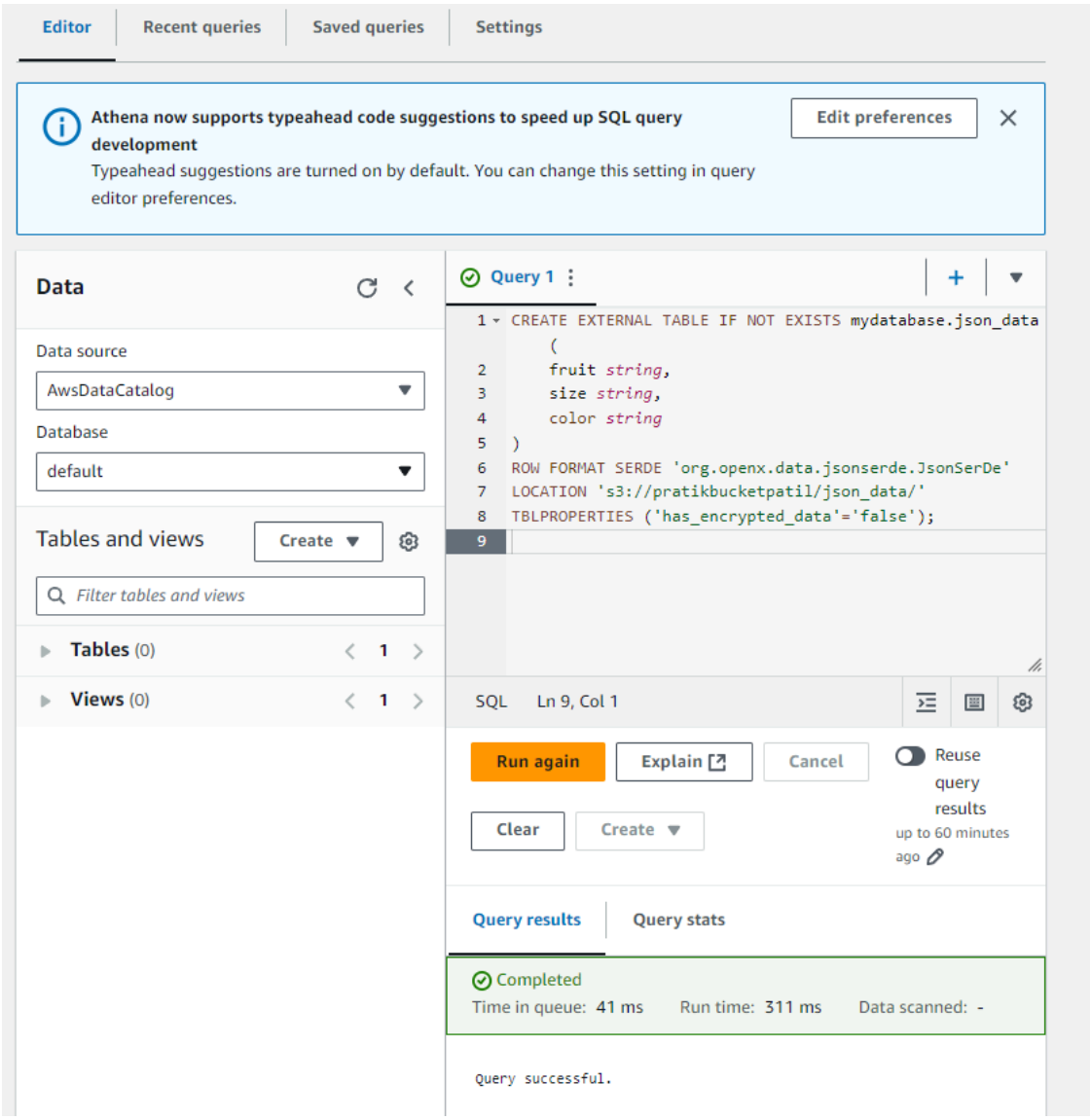
Clear | Create

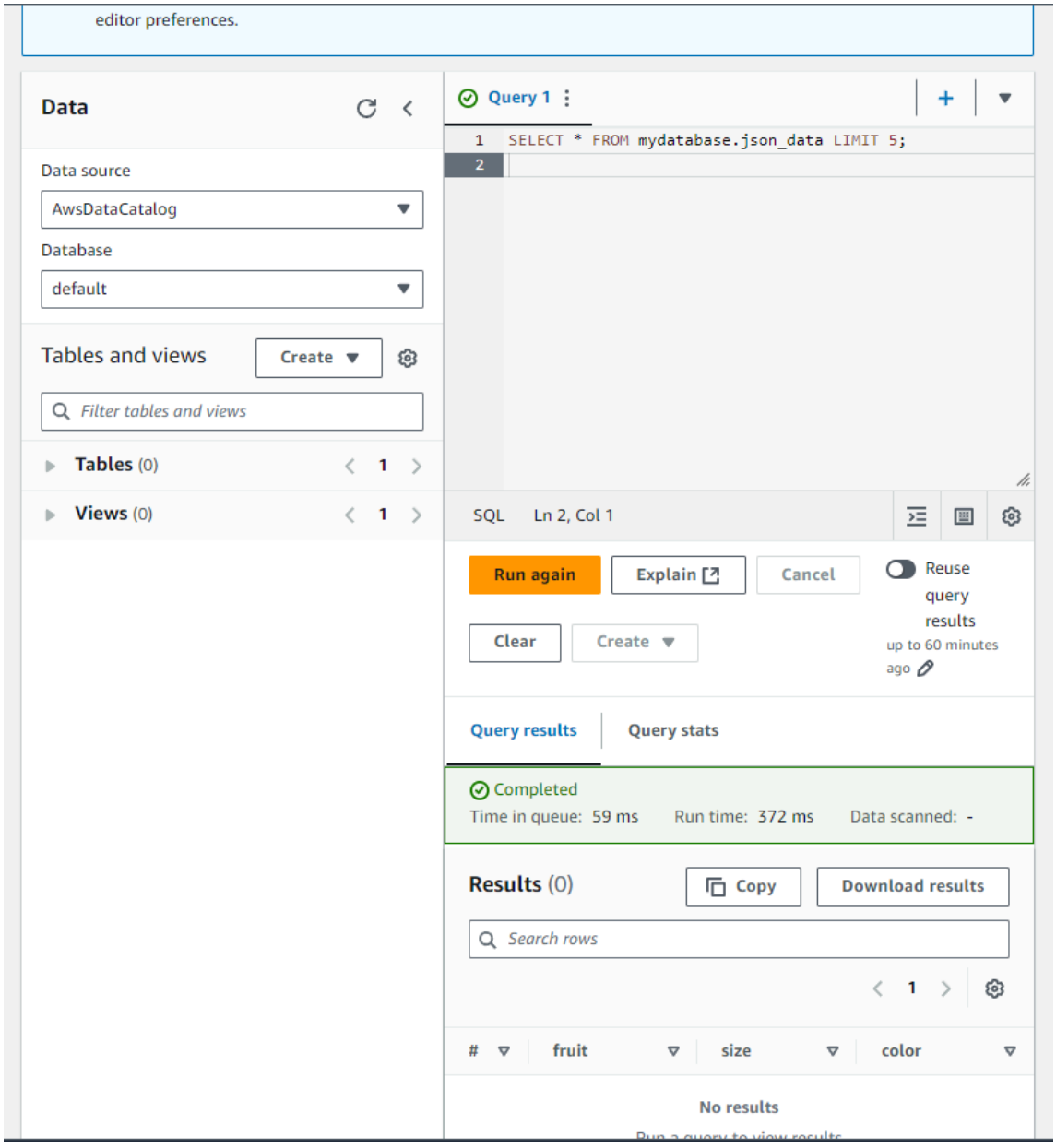
Query results | Query stats

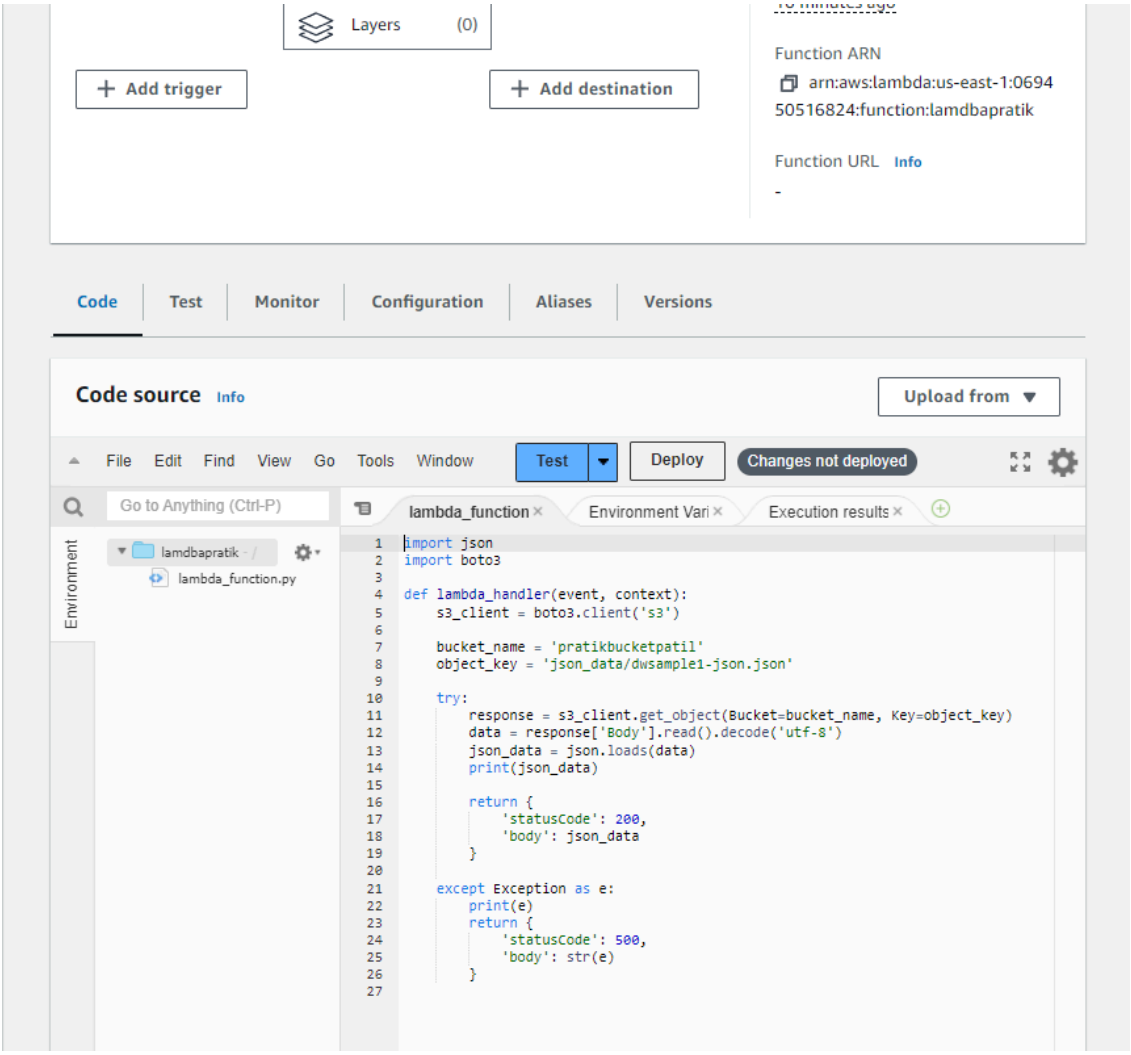
Completed

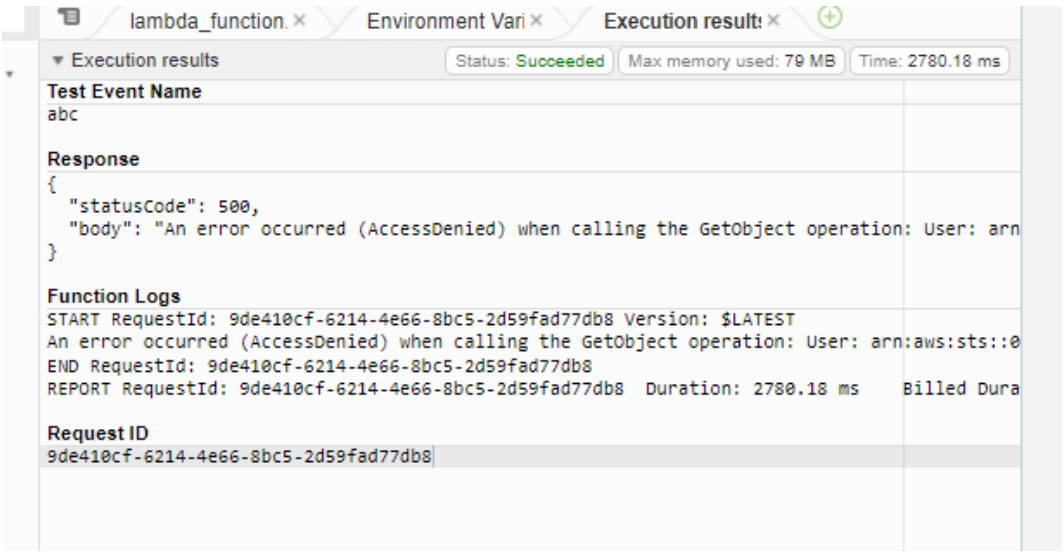
Time in queue: 51 ms Run time: 358 ms Data scanned: -

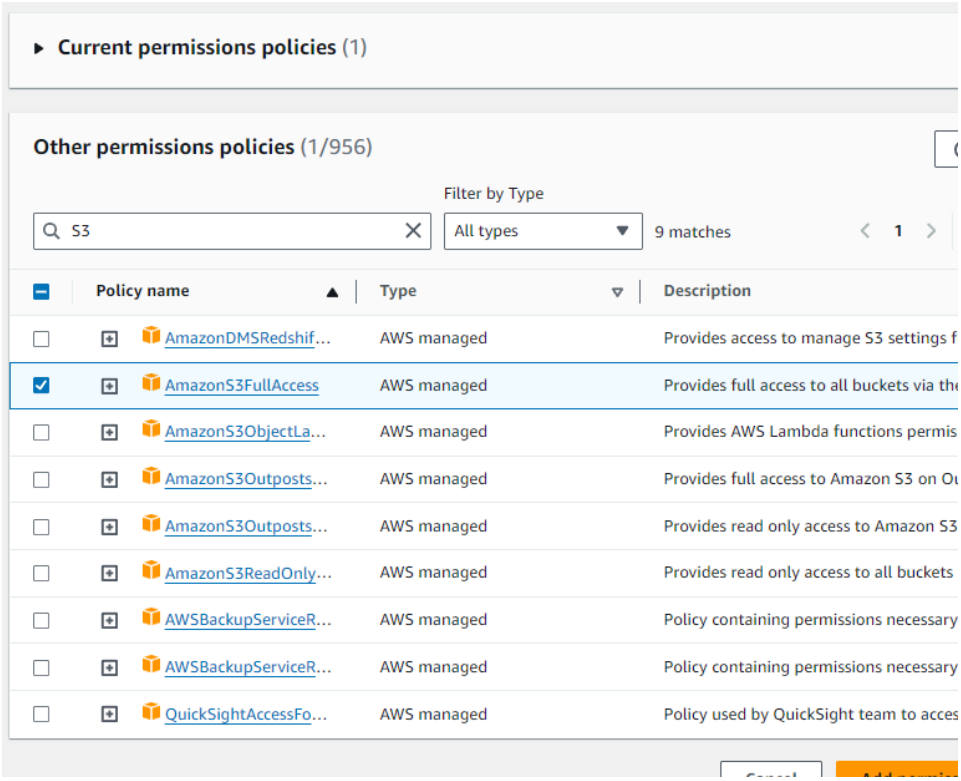
```
Query successful.
```

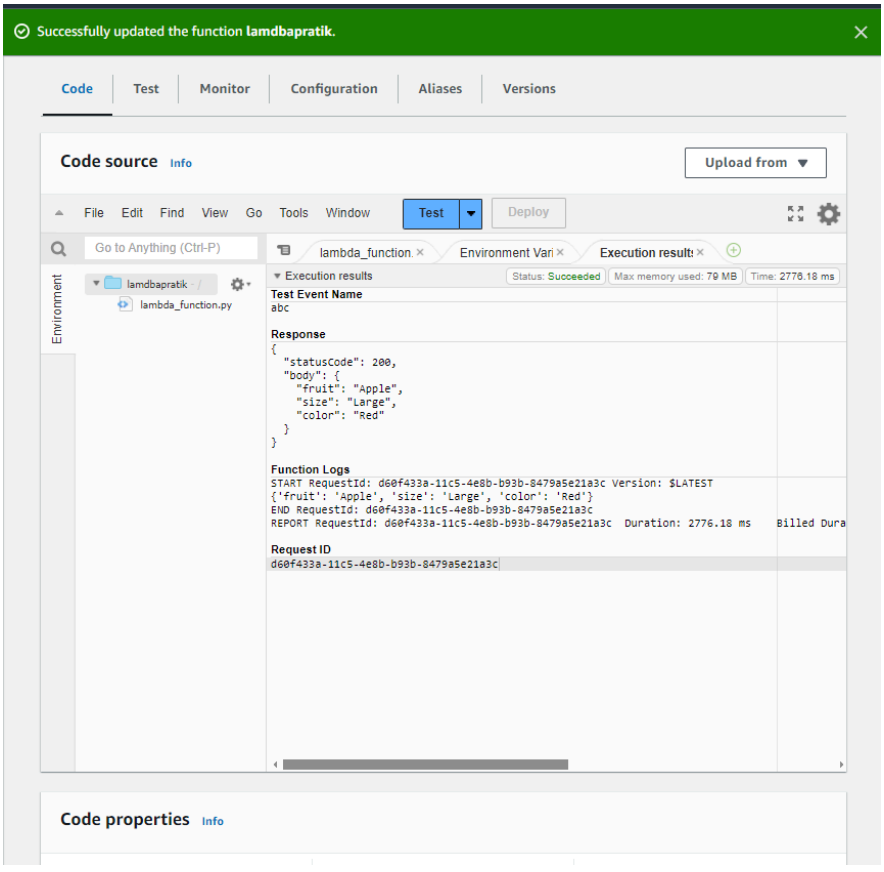

Editor
Recent queries
Saved queries
Settings
Athena now supports typeahead code suggestions to speed up SQL query development
Typeahead suggestions are turned on by default. You can change this setting in query editor preferences.
Edit preferences
Data
Data source
AwsDataCatalog
Database
default
Tables and views
Create
Filter tables and views
Tables (0)
Views (0)
Query 1
1 CREATE EXTERNAL TABLE IF NOT EXISTS mydatabase.json_data
(2 fruit string, 3 size string, 4 color string 5) 6 ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe' 7 LOCATION 's3://pratikbucketpatil/json_data/' 8 TBLPROPERTIES ('has_encrypted_data'='false'); 9
SQL Ln 9, Col 1
Run again
Explain
Cancel
Reuse query results
up to 60 minutes ago
Clear
Create
Query results
Query stats
Completed
Time in queue: 41 ms
Run time: 311 ms
Data scanned: -
Query successful.

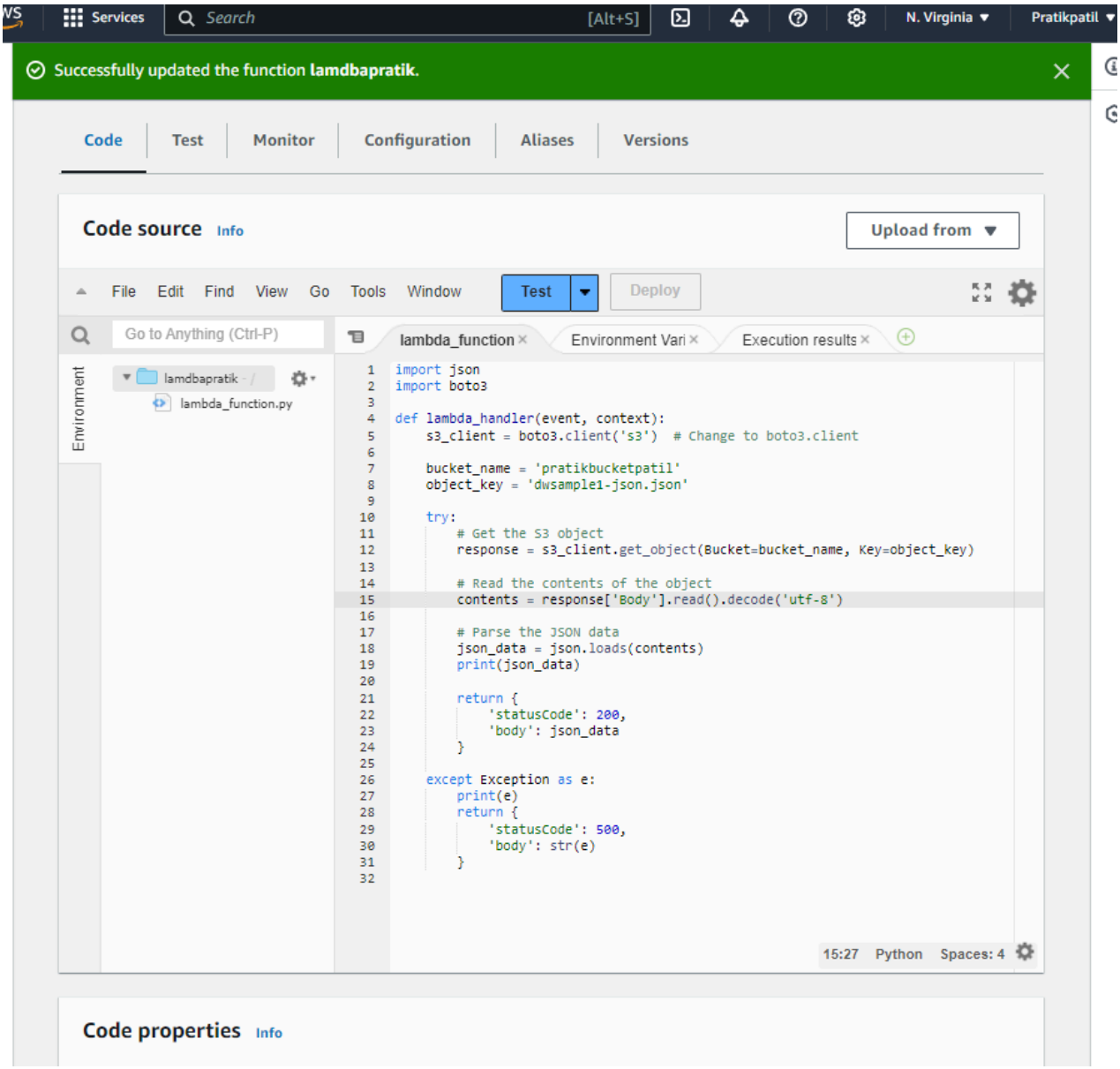

editor preferences.
Data
Data source
AwsDataCatalog
Database
default
Tables and views
Create
Filter tables and views
Tables (0)
1
Views (0)
1
Query 1
1 SELECT * FROM mydatabase.json_data LIMIT 5;
2
SQL Ln 2, Col 1
Run again
Explain
Cancel
Reuse query results
up to 60 minutes ago
Clear
Create
Query results
Query stats
Completed
Time in queue: 59 ms
Run time: 372 ms
Data scanned: -
Results (0)
Copy
Download results
Search rows
1
fruit size color
No results


Layers (0)
+ Add trigger
+ Add destination
Function ARN
arn:aws:lambda:us-east-1:069450516824:function:lamdbapratik
Function URL Info
-
Code
Test
Monitor
Configuration
Aliases
Versions
Code source Info
Upload from
File Edit Find View Go Tools Window
Test
Deploy
Changes not deployed
Go to Anything (Ctrl-P)
Environment
lamdbapratik
lambda_function.py
lambda_function
Environment Vari
Execution results
import json
import boto3
def lambda_handler(event, context):
s3_client = boto3.client('s3')
bucket_name = 'pratikbucketpatil'
object_key = 'json_data/dwsample1-json.json'
try:
response = s3_client.get_object(Bucket=bucket_name, Key=object_key)
data = response['Body'].read().decode('utf-8')
json_data = json.loads(data)
print(json_data)
return {
'statusCode': 200,
'body': json_data
}
except Exception as e:
print(e)
return {
'statusCode': 500,
'body': str(e)
}


lambda_function
Environment Vari
Execution results
Execution results
Status: Succeeded
Max memory used: 79 MB
Time: 2780.18 ms
Test Event Name
abc
Response
{
"statusCode": 500,
"body": "An error occurred (AccessDenied) when calling the GetObject operation: User: arn
}
Function Logs
START RequestId: 9de410cf-6214-4e66-8bc5-2d59fad77db8 Version: $LATEST
An error occurred (AccessDenied) when calling the GetObject operation: User: arn:aws:sts::0
END RequestId: 9de410cf-6214-4e66-8bc5-2d59fad77db8
REPORT RequestId: 9de410cf-6214-4e66-8bc5-2d59fad77db8 Duration: 2780.18 ms Billed Dura
Request ID
9de410cf-6214-4e66-8bc5-2d59fad77db8


Current permissions policies (1)
Other permissions policies (1/956)
Filter by Type
S3
All types
9 matches
1
Policy name
Type
Description
AmazonDMSRedshif...
AWS managed
Provides access to manage S3 settings f
AmazonS3FullAccess
AWS managed
Provides full access to all buckets via the
AmazonS3ObjectLa...
AWS managed
Provides AWS Lambda functions permis
AmazonS3Outposts...
AWS managed
Provides full access to Amazon S3 on Ou
AmazonS3Outposts...
AWS managed
Provides read only access to Amazon S3
AmazonS3ReadOnly...
AWS managed
Provides read only access to all buckets
AWSBackupServiceR...
AWS managed
Policy containing permissions necessary
AWSBackupServiceR...
AWS managed
Policy containing permissions necessary
QuickSightAccessFo...
AWS managed
Policy used by QuickSight team to acces


Successfully updated the function lamdbapratik.
Code
Test
Monitor
Configuration
Aliases
Versions
Code source Info
Upload from
File Edit Find View Go Tools Window
Test
Deploy
Go to Anything (Ctrl-P)
lambda_function
Environment Vari
Execution result
Environment
lamdbapratik
lambda_function.py
Execution results
Status: Succeeded
Max memory used: 79 MB
Time: 2776.18 ms
Test Event Name
abc
Response
{
"statusCode": 200,
"body": {
"fruit": "Apple",
"size": "Large",
"color": "Red"
}
}
Function Logs
START RequestId: d60f433a-11c5-4e8b-b93b-8479a5e21a3c Version: $LATEST
{'fruit': 'Apple', 'size': 'Large', 'color': 'Red'}
END RequestId: d60f433a-11c5-4e8b-b93b-8479a5e21a3c
REPORT RequestId: d60f433a-11c5-4e8b-b93b-8479a5e21a3c Duration: 2776.18 ms Billed Dura
Request ID
d60f433a-11c5-4e8b-b93b-8479a5e21a3c
Code properties Info



Conclusion:

Integrating AWS Lambda with S3 enables real-time, automated processing of events, such as file uploads. In this example, a Lambda function is configured to log a message whenever an image is added to a specific S3 bucket. This setup highlights the advantages of serverless computing by automating tasks without the need for manual intervention or server management. By utilizing AWS Lambda, developers can effectively manage event-driven workflows, minimize operational overhead, and swiftly deploy scalable solutions that react to specific actions within cloud environments. This approach not only enhances efficiency but also allows organizations to focus more on innovation rather than infrastructure maintenance.