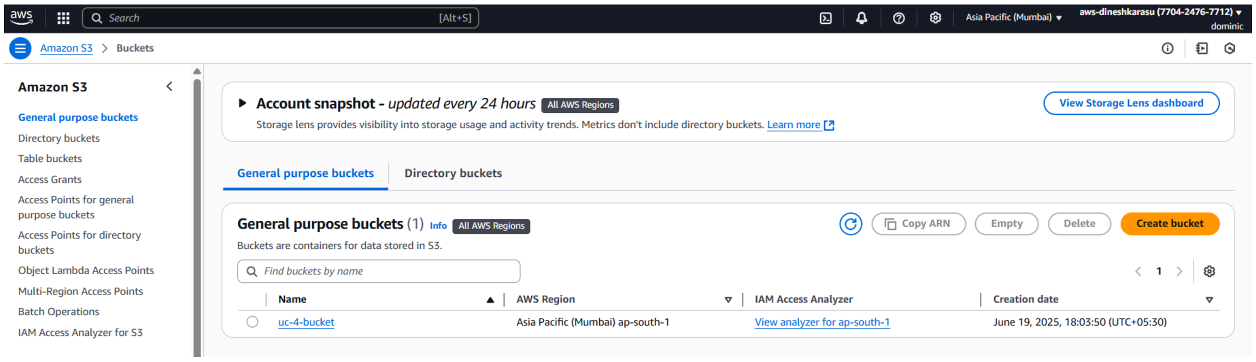
Use case

Deploying a simple serverless app

Use case discription

using Lambda, API Gateway, and S3 for a basic CRUD operation

1. Navigate to S3 service from AWS console and create a simple bucket, here we created the bucket **uc-4-bucket** as shown below



Amazon S3 > Buckets

Account snapshot - *updated every 24 hours* All AWS Regions

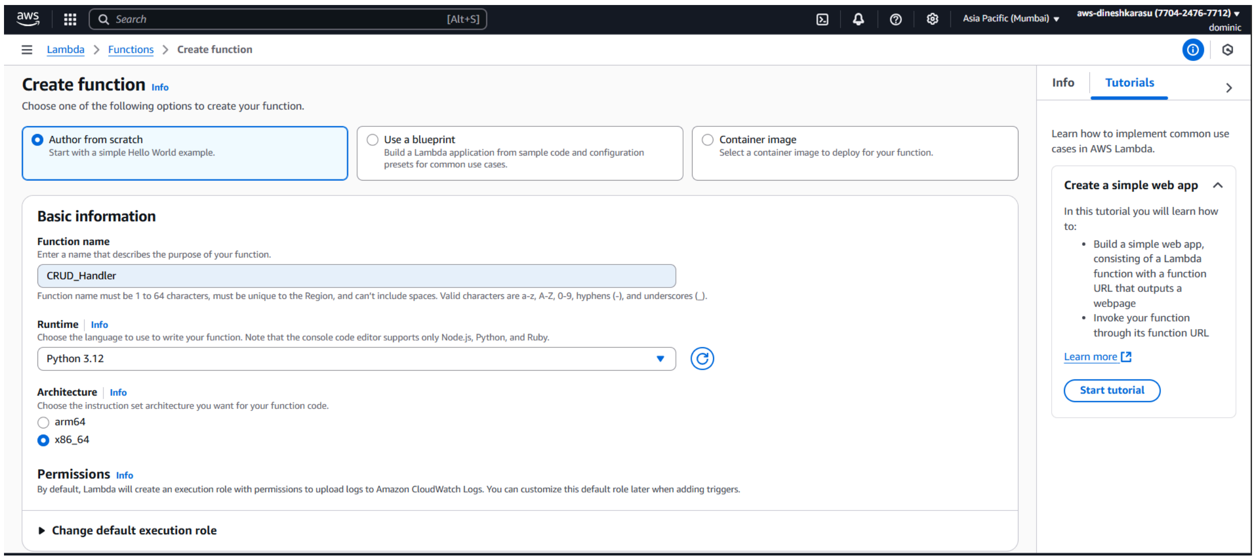
Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. Learn more

General purpose buckets (1) Info All AWS Regions

Buckets are containers for data stored in S3.

Name	AWS Region	IAM Access Analyzer	Creation date
uc-4-bucket	Asia Pacific (Mumbai) ap-south-1	View analyzer for ap-south-1	June 19, 2025, 18:03:50 (UTC+05:30)

2. Create a lambda function CRUD_Handler



Lambda > Functions > Create function

Create function Info

Choose one of the following options to create your function.

[✓ Author from scratch] Start with a simple Hello World example.

Use a blueprint — Build a Lambda application from sample code and configuration presets for common use cases.

Container image — Select a container image to deploy for your function.

Basic information

Function name: CRUD_Handler

Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).

Runtime: Python 3.12

Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

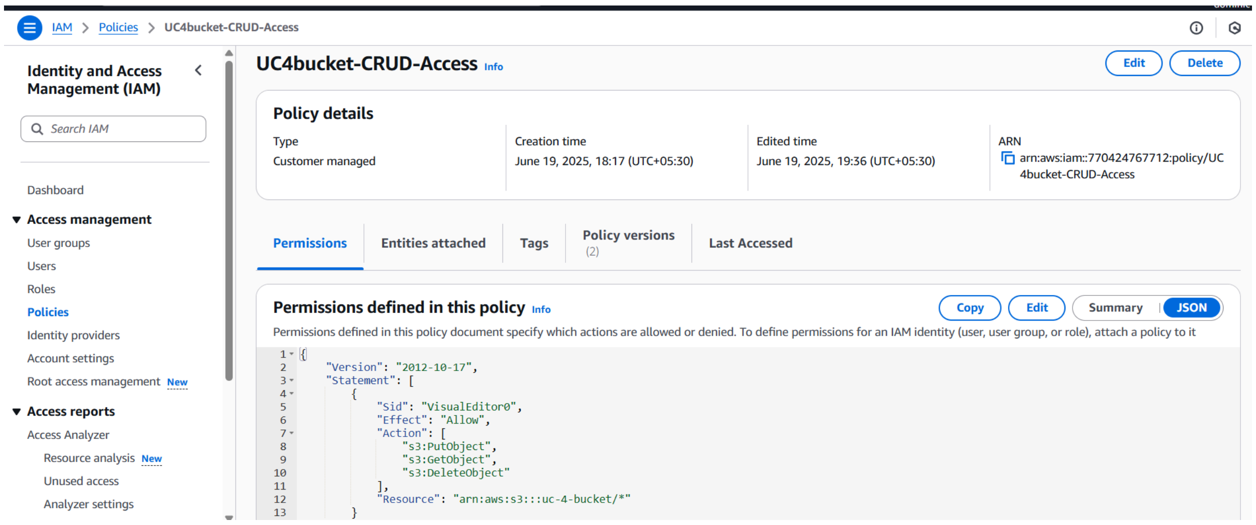
Architecture: arm64 [✓ x86_64]

Permissions Info

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

Change default execution role

3. Navigate to IAM service and assign proper permissions to the Lambda that we just created (GetObject, DeleteObject, PutObject) by creating and attaching new policies



UC4bucket-CRUD-Access Info

Policy details

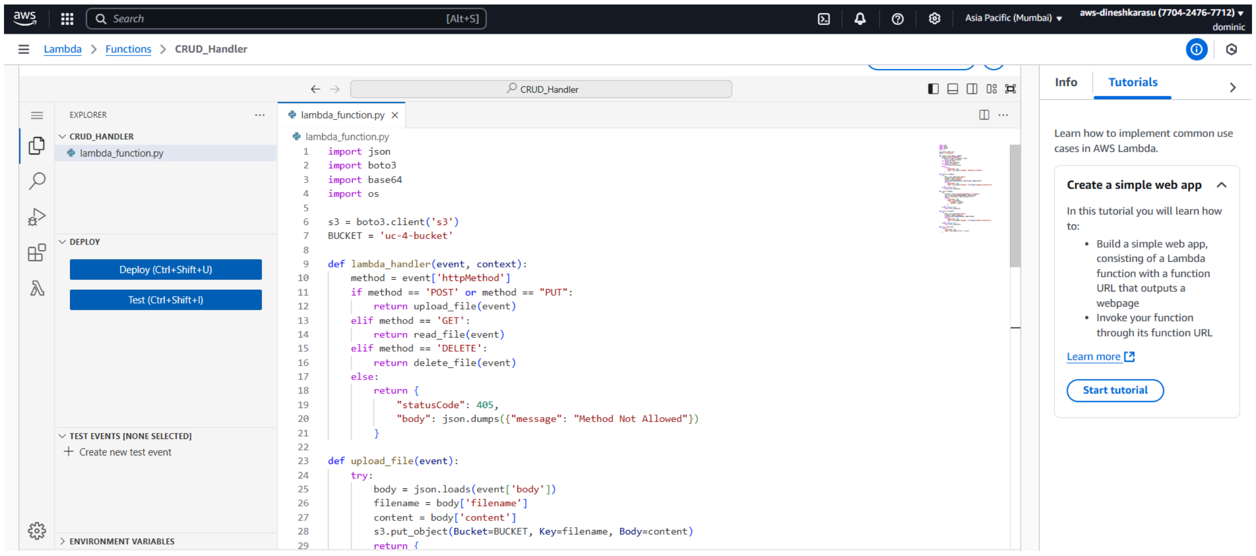
Type	Creation time	Edited time	ARN
Customer managed	June 19, 2025, 18:17 (UTC+05:30)	June 19, 2025, 19:36 (UTC+05:30)	arn:aws:iam::770424767712:policy/UC4bucket-CRUD-Access

Permissions defined in this policy Info

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::uc-4-bucket/*"
        }
```

4. Now, write the logic for the lambda to execute whenever the function gets triggered by the API Gateway
5. Analyse the JSON request that API Gateway sends to the Lambda, based on the **"httpmethod"** parameter value, check whether it is a Get/Post/Delete method and write corresponding logic to execute the action by input request body



Learn how to implement common use cases in AWS Lambda.

In this tutorial you will learn how to:
- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

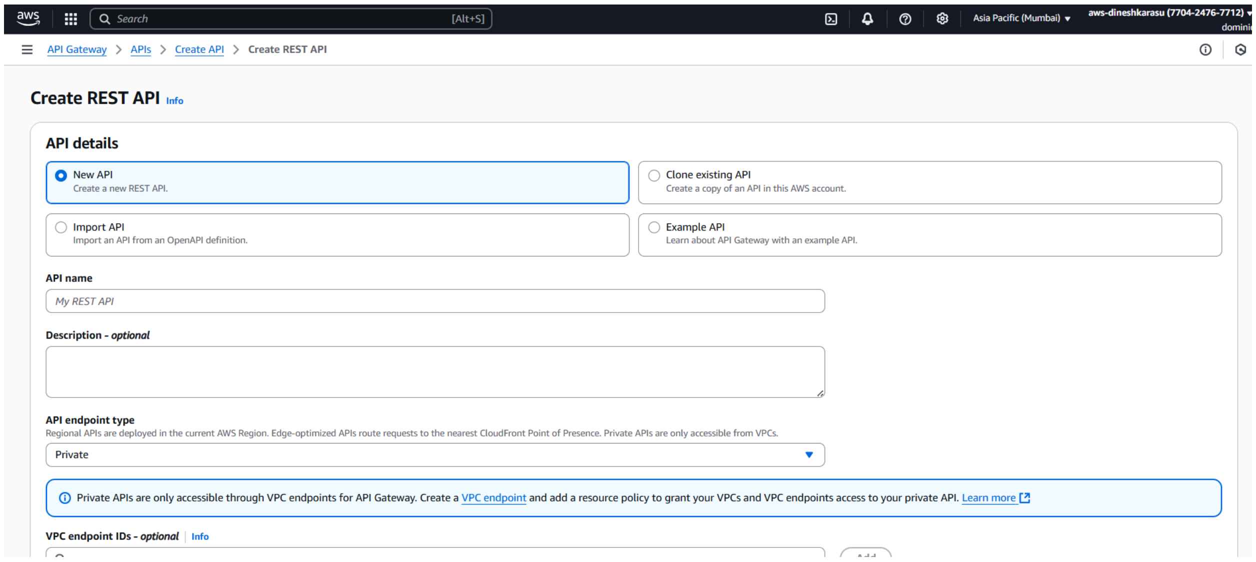
```python
import json
import boto3
import base64
import os

s3 = boto3.client('s3')
BUCKET = 'uc-4-bucket'

def lambda_handler(event, context):
    method = event['httpMethod']
    if method == 'POST' or method == "PUT":
        return upload_file(event)
    elif method == 'GET':
        return read_file(event)
    elif method == 'DELETE':
        return delete_file(event)
    else:
        return {
            "statusCode": 405,
            "body": json.dumps({"message": "Method Not Allowed"})
        }

def upload_file(event):
    try:
        body = json.loads(event['body'])
        filename = body['filename']
        content = body['content']
        s3.put_object(Bucket=BUCKET, Key=filename, Body=content)
        return {
```

6. Navigate to API Gateway and create a REST API



Search [Alt+S] Asia Pacific (Mumbai) aws-dineshkarasu (7704-2476-7712)

API Gateway > APIs > Create API > Create REST API

Create REST API Info

API details

(●) New API — Create a new REST API.
( ) Clone existing API — Create a copy of an API in this AWS account.
( ) Import API — Import an API from an OpenAPI definition.
( ) Example API — Learn about API Gateway with an example API.

API name: My REST API

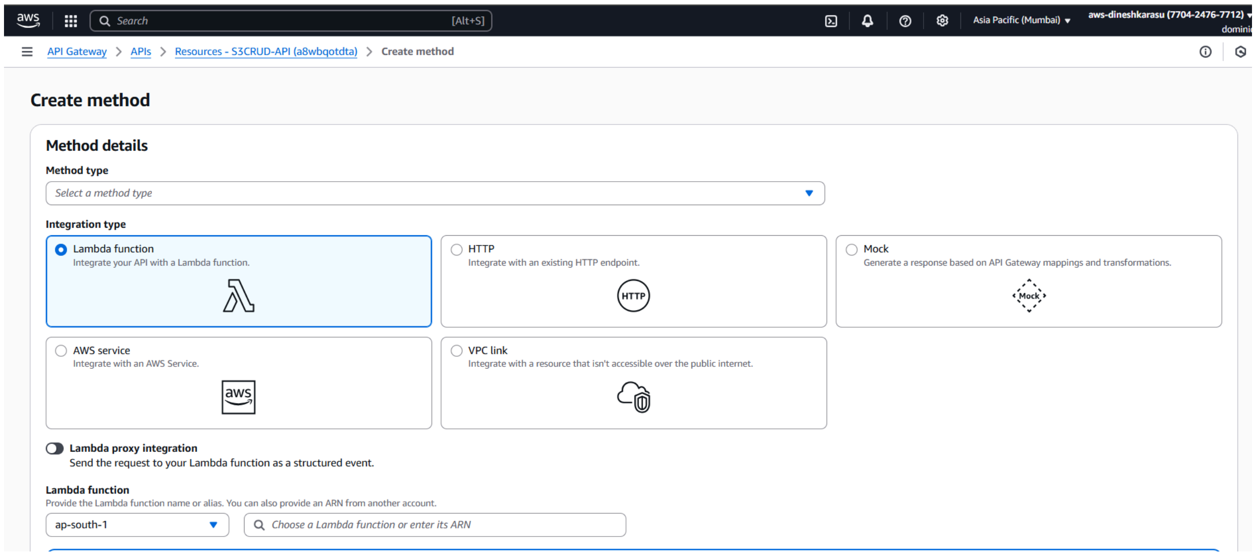
Description - *optional*

API endpoint type
Regional APIs are deployed in the current AWS Region. Edge-optimized APIs route requests to the nearest CloudFront Point of Presence. Private APIs are only accessible from VPCs.
Private

Private APIs are only accessible through VPC endpoints for API Gateway. Create a VPC endpoint and add a resource policy to grant your VPCs and VPC endpoints access to your private API. Learn more

VPC endpoint IDs - *optional* Info

7. Create a resource and add methods (Get, Post, Put, Delete) to that resource
8. Select the method type, ambda proxy and lambda function while creating the method



Search [Alt+S] Asia Pacific (Mumbai) aws-dineshkarasu (7704-2476-7712)

API Gateway > APIs > Resources - S3CRUD-API (a8wbqotdta) > Create method

Create method

Method details

Method type: Select a method type

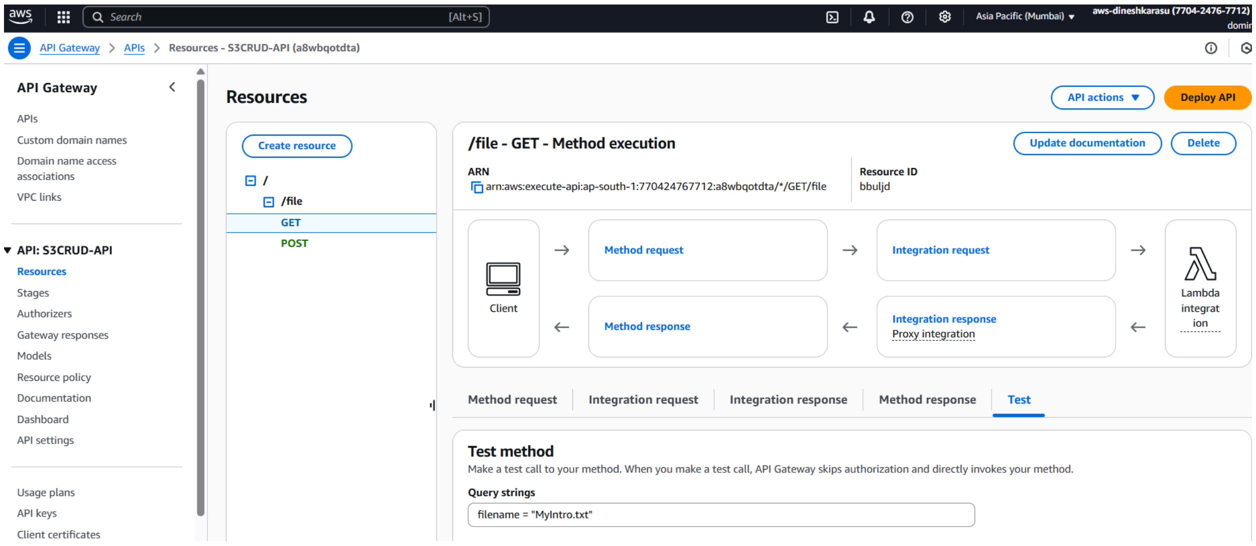
Integration type

(●) Lambda function — Integrate your API with a Lambda function.
( ) HTTP — Integrate with an existing HTTP endpoint.
( ) Mock — Generate a response based on API Gateway mappings and transformations.
( ) AWS service — Integrate with an AWS Service.
( ) VPC link — Integrate with a resource that isn't accessible over the public internet.

Lambda proxy integration
Send the request to your Lambda function as a structured event.

Lambda function
Provide the Lambda function name or alias. You can also provide an ARN from another account.
ap-south-1 | Choose a Lambda function or enter its ARN

9. Properly format URL QueryParameters/Request Body JSON and create methods
10. Test by the endpoint either in Postman or in the test tab provided down there itself



/file - GET - Method execution

ARN: arn:aws:execute-api:ap-south-1:770424767712:a8wbqotdta/*/GET/file

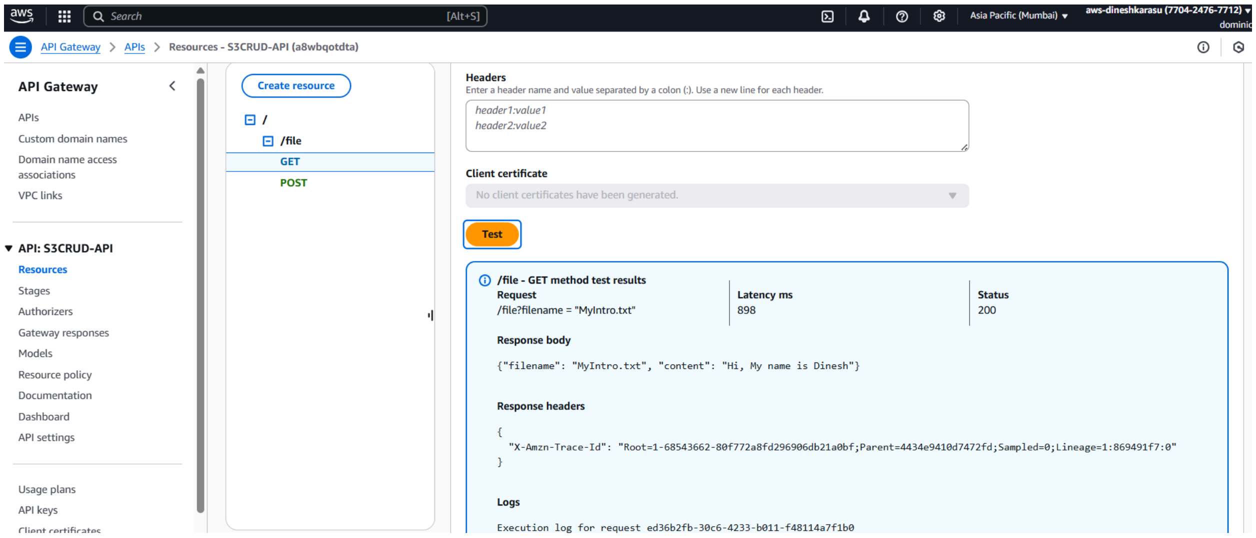
Resource ID: bbuljd

Test method

Make a test call to your method. When you make a test call, API Gateway skips authorization and directly invokes your method.

Query strings

filename = "MyIntro.txt"



Headers

Enter a header name and value separated by a colon (:). Use a new line for each header.

header1:value1
header2:value2

Client certificate

No client certificates have been generated.

/file - GET method test results

Request	Latency ms	Status
/file?filename = "MyIntro.txt"	898	200

Response body

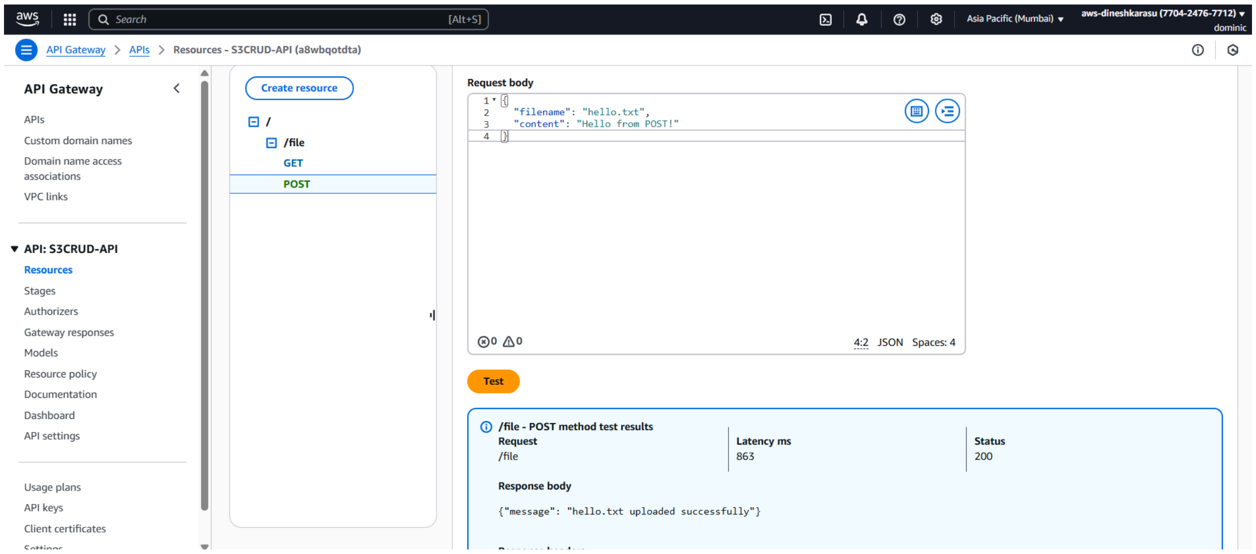
```
{"filename": "MyIntro.txt", "content": "Hi, My name is Dinesh"}
```

Response headers

```
{
  "X-Amzn-Trace-Id": "Root=1-68543662-80f772a8fd296906db21a0bf;Parent=4434e9410d7472fd;Sampled=0;Lineage=1:869491f7:0"
}
```

Logs

```
Execution log for request ed36b2fb-30c6-4233-b011-f48114a7f1b0
```



Request body

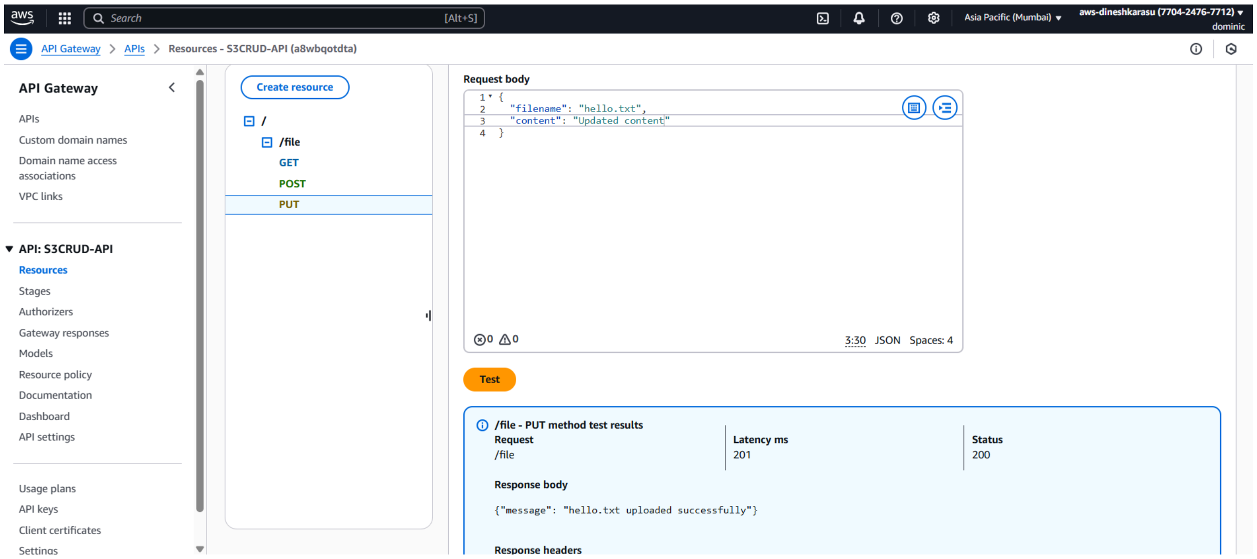
```
{
    "filename": "hello.txt",
    "content": "Hello from POST!"
}
```

/file - POST method test results

Request	Latency ms	Status
/file	863	200

Response body

```
{"message": "hello.txt uploaded successfully"}
```



Request body

```
{
  "filename": "hello.txt",
  "content": "Updated content"
}
```

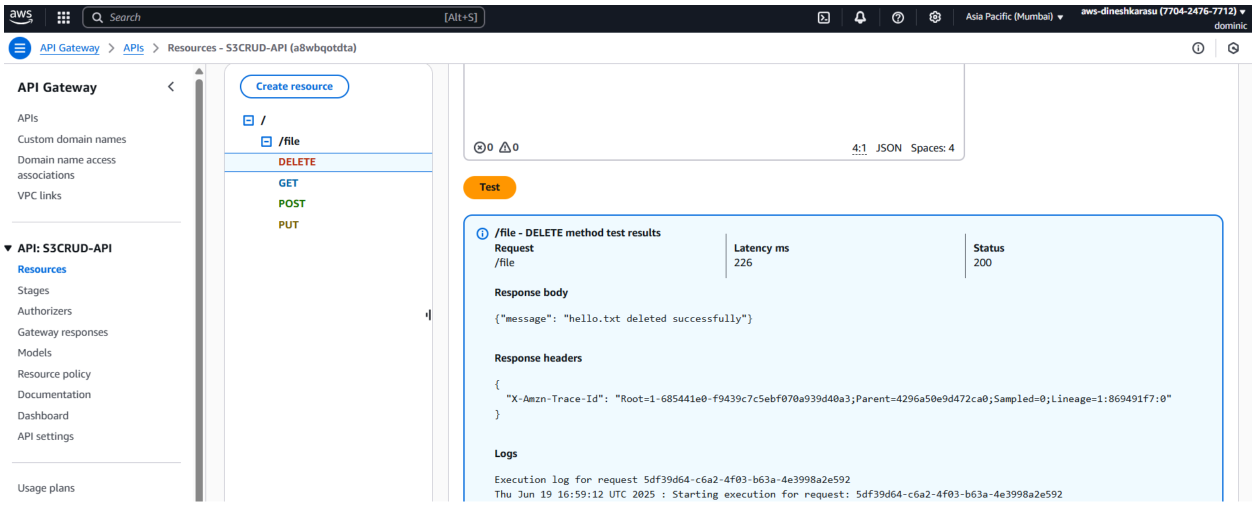
/file - PUT method test results

Request	Latency ms	Status
/file	201	200

Response body

```
{"message": "hello.txt uploaded successfully"}
```

Response headers



/file - DELETE method test results

Request	Latency ms	Status
/file	226	200

Response body

```
{"message": "hello.txt deleted successfully"}
```

Response headers

```
{
  "X-Amzn-Trace-Id": "Root=1-685441e0-f9439c7c5ebf070a939d40a3;Parent=4296a50e9d472ca0;Sampled=0;Lineage=1:869491f7:0"
}
```

Logs

```
Execution log for request 5df39d64-c6a2-4f03-b63a-4e3998a2e592
Thu Jun 19 16:59:12 UTC 2025 : Starting execution for request: 5df39d64-c6a2-4f03-b63a-4e3998a2e592
```

11. After testing, we can deploy the API to the desired stage. And use the endpoint whenever needed