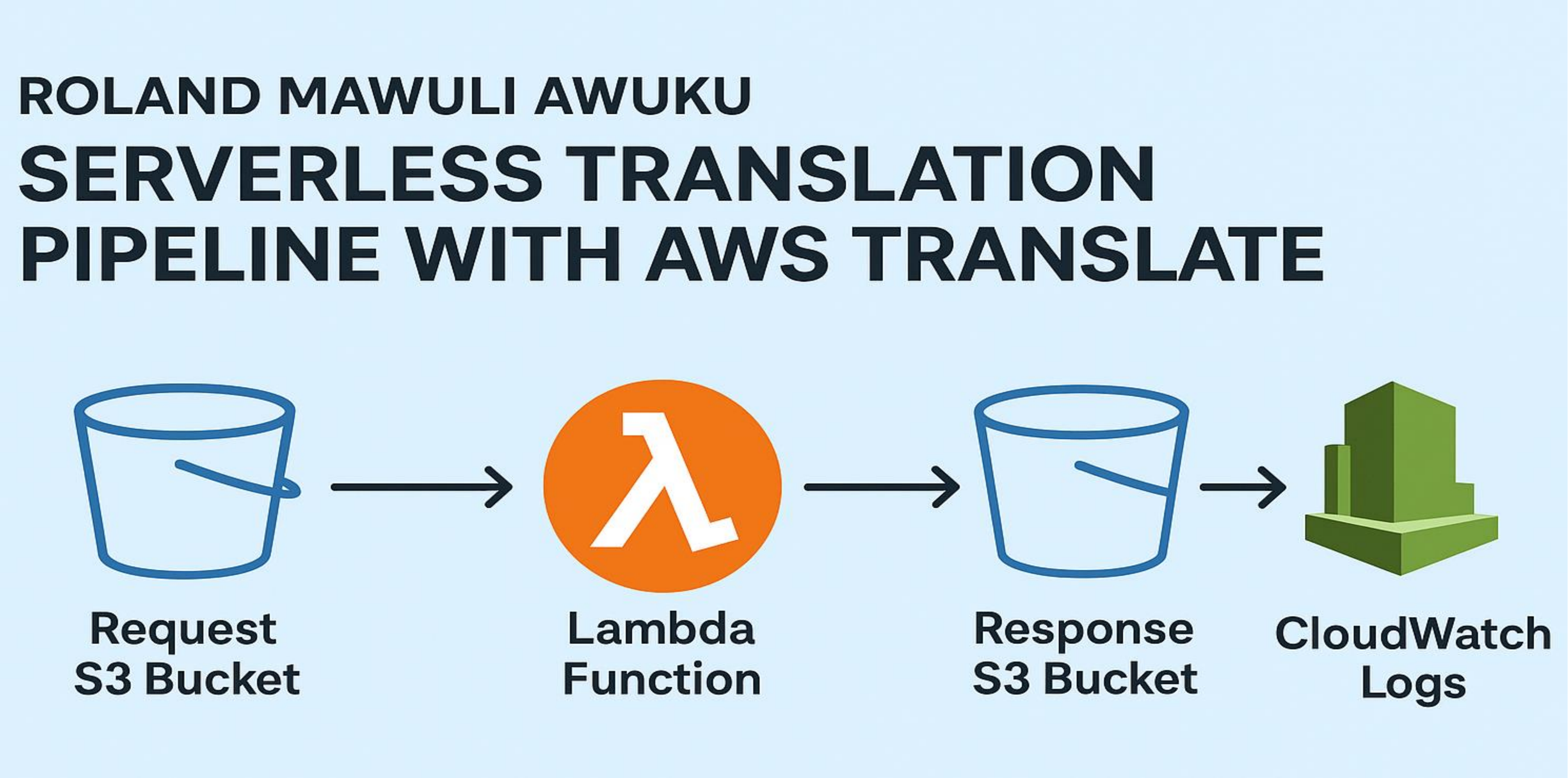
CAPSTONE PROJECT REPORT



INTRODUCTION

- The goal is to architect, build, and deploy an Infrastructure-as-Code (IaC) solution on AWS that enables automated, real-time language translation.

- This implementation leverages Amazon S3, AWS Lambda, and Amazon Translate, with all resources provisioned and managed through Terraform.

- The solution is entirely serverless and optimized to stay within AWS Free Tier limits when handling small-scale test datasets.

- Automate the translation of text contained in uploaded JSON files.
- Save both the original requests and their translated outputs securely in Amazon S3.
- Leverage a serverless architecture to ensure scalability while minimizing operational overhead.
- Provision and deploy infrastructure consistently and reproducibly using Terraform (IaC).
- Apply best practices for IAM, CloudWatch logging, and cost optimization throughout the solution.

The solution is built as an event-driven pipeline:

- ❖ Request S3 Bucket. Holds uploaded JSON files containing translation requests.
- ❖ S3 Event Notification. Automatically triggers the Lambda function upon file upload.
- ❖ Lambda Function (Python, Boto3). Processes the JSON file, invokes Amazon Translate, and generates translated output.
- ❖ Response S3 Bucket. Stores the resulting translated JSON files.
- ❖ CloudWatch Logs. Records execution details, errors, and debugging information.
- ❖ Terraform. Manages infrastructure as code, provisioning S3 buckets, IAM roles, and the Lambda function

PHASE 1: ENVIRONMENT PREPARATION & AWS SETUP

The first stage established the groundwork for the translation pipeline by configuring core AWS services, access controls, and the development environment.

Researched Core AWS Services

Amazon Translate (real-time neural machine translation): Reviewed supported languages, character limits (5,000 bytes per request), and Free Tier allowance (2M characters/month for 12 months).

Amazon S3 (scalable object storage): Studied key features such as versioning, encryption, bucket policies, lifecycle management, and event notifications to support later automation steps.

Installed and Configured Essential Tools

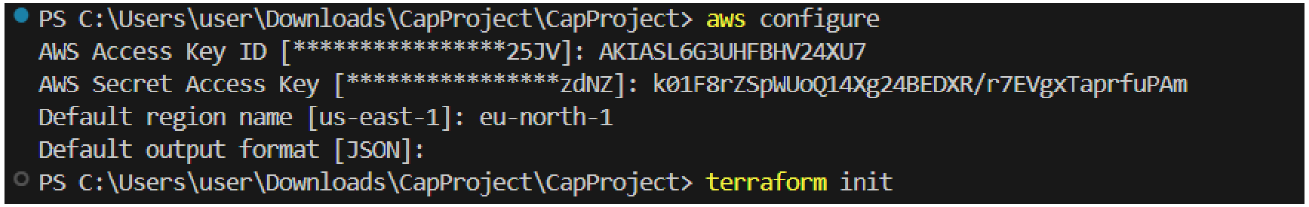
AWS CLI - enabled local interaction with AWS resources.

Terraform - for Infrastructure-as-Code resource provisioning.

Python - selected for Lambda function development.

AWS Credentials Setup

Configured local authentication using `aws configure`



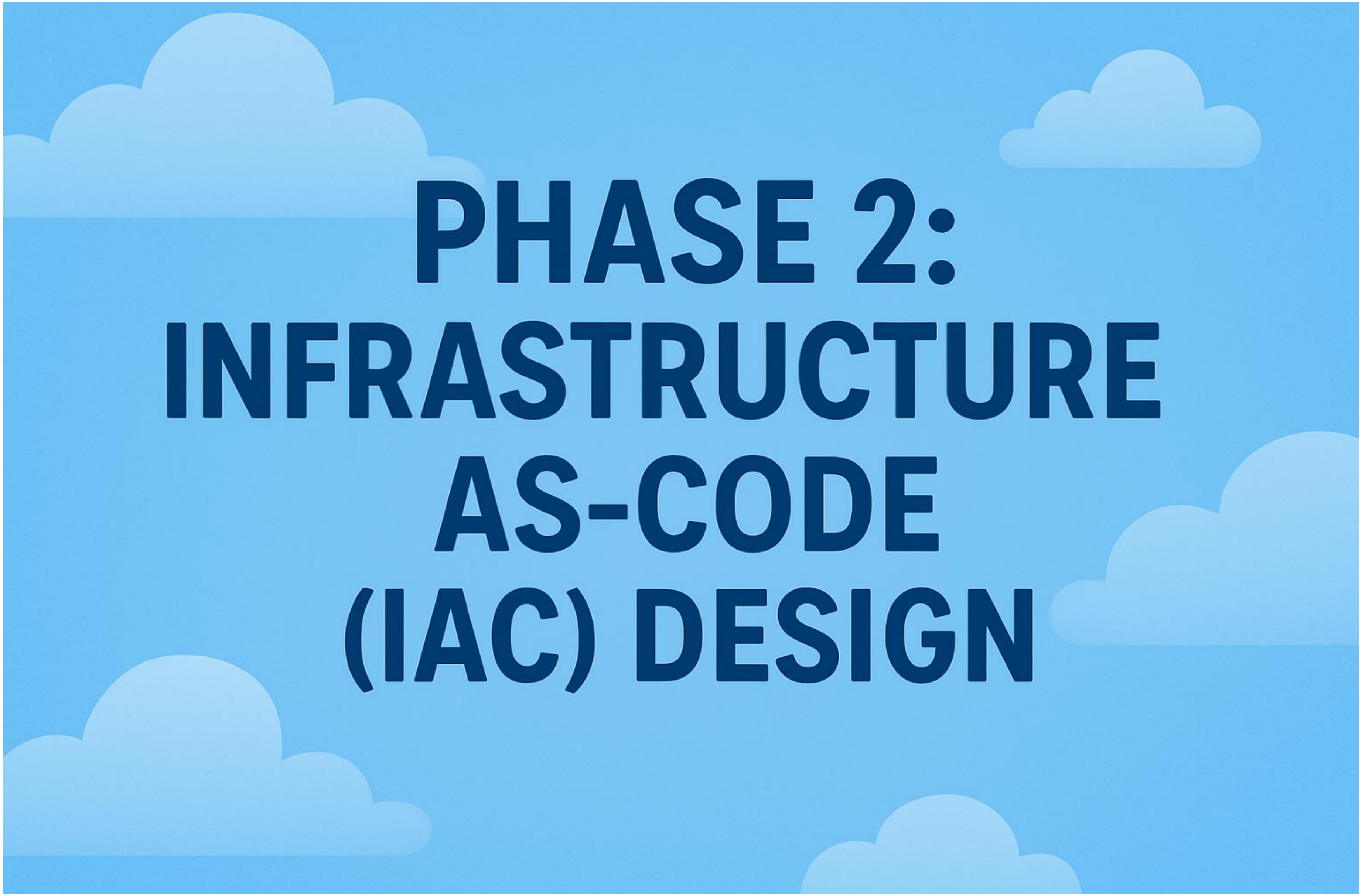
```
● PS C:\Users\user\Downloads\CapProject\CapProject> aws configure
  AWS Access Key ID [****************25JV]: AKIASL6G3UHFBHV24XU7
  AWS Secret Access Key [****************zdNZ]: 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
  Default region name [us-east-1]: eu-north-1
  Default output format [JSON]:
○ PS C:\Users\user\Downloads\CapProject\CapProject> terraform init
```

Drafted IAM Policy with Scoped Permissions

`translate:TranslateText` - access to Amazon Translate.

`s3:GetObject, s3:PutObject, s3:ListBucket` - permissions for Amazon S3.

CloudWatch log permissions - to capture Lambda execution details.

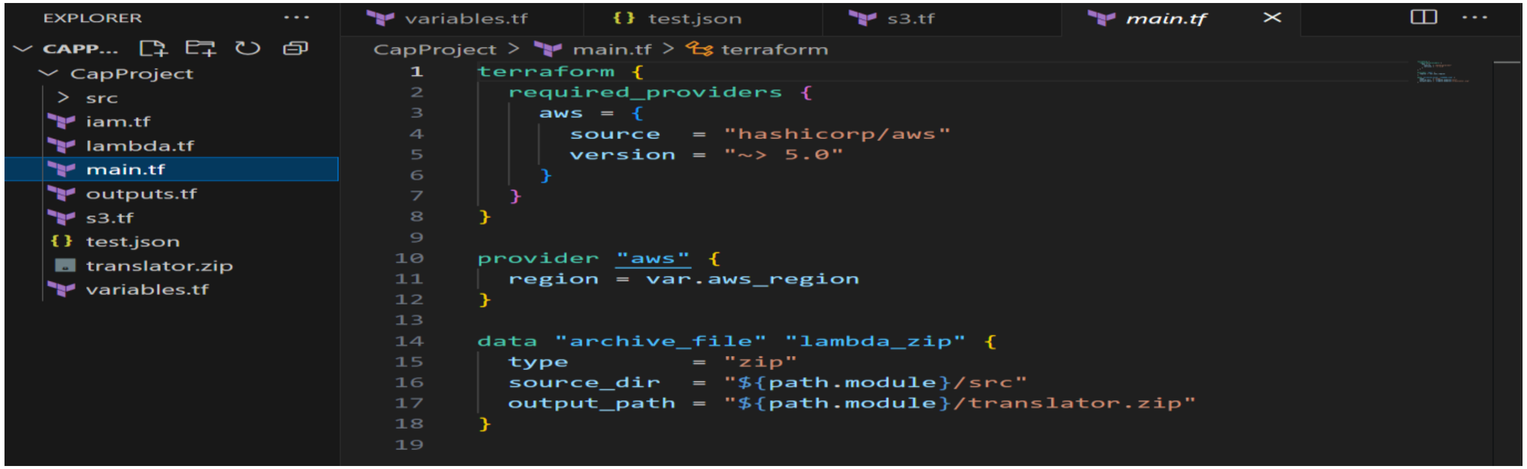


PHASE 2: INFRASTRUCTURE AS-CODE (IAC) DESIGN

In this phase, Terraform was used to fully automate the provisioning of AWS resources, including S3 buckets, IAM roles, and the Lambda deployment. This ensured a reproducible and error-free setup compared to manual configuration.

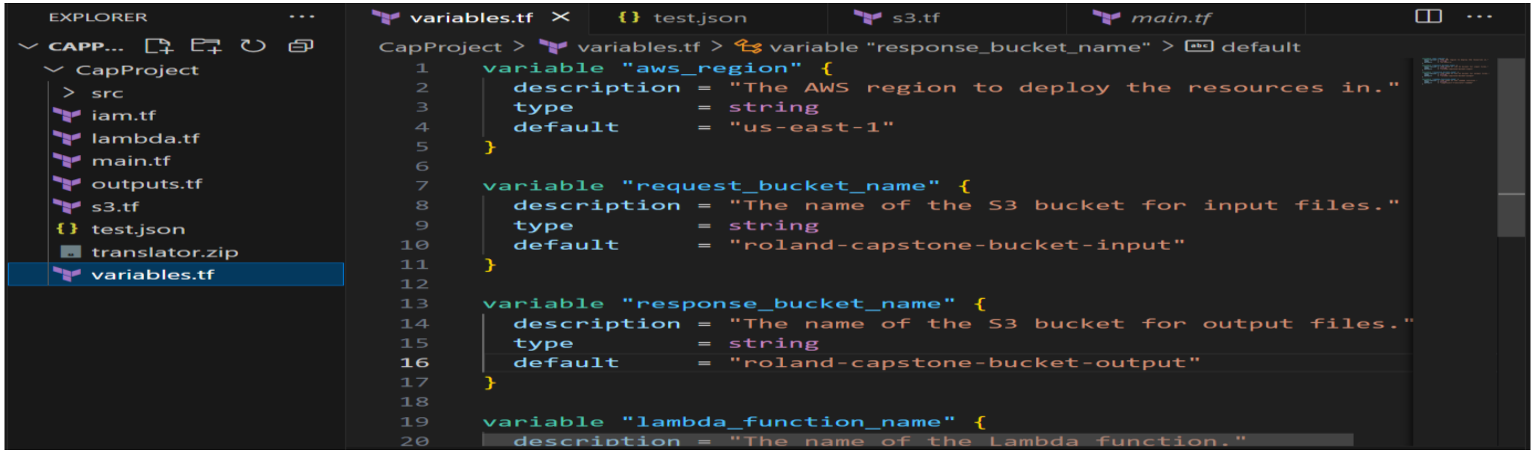
ACTIVITIES:

Created Terraform provider configuration



```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}

provider "aws" {
  region = var.aws_region
}

data "archive_file" "lambda_zip" {
  type        = "zip"
  source_dir  = "${path.module}/src"
  output_path = "${path.module}/translator.zip"
}
```

Defined variables for region, input/output buckets, and Lambda function:



```
variable "aws_region" {
  description = "The AWS region to deploy the resources in."
  type        = string
  default     = "us-east-1"
}

variable "request_bucket_name" {
  description = "The name of the S3 bucket for input files."
  type        = string
  default     = "roland-capstone-bucket-input"
}

variable "response_bucket_name" {
  description = "The name of the S3 bucket for output files."
  type        = string
  default     = "roland-capstone-bucket-output"
}

variable "lambda_function_name" {
  description = "The name of the Lambda function."
```

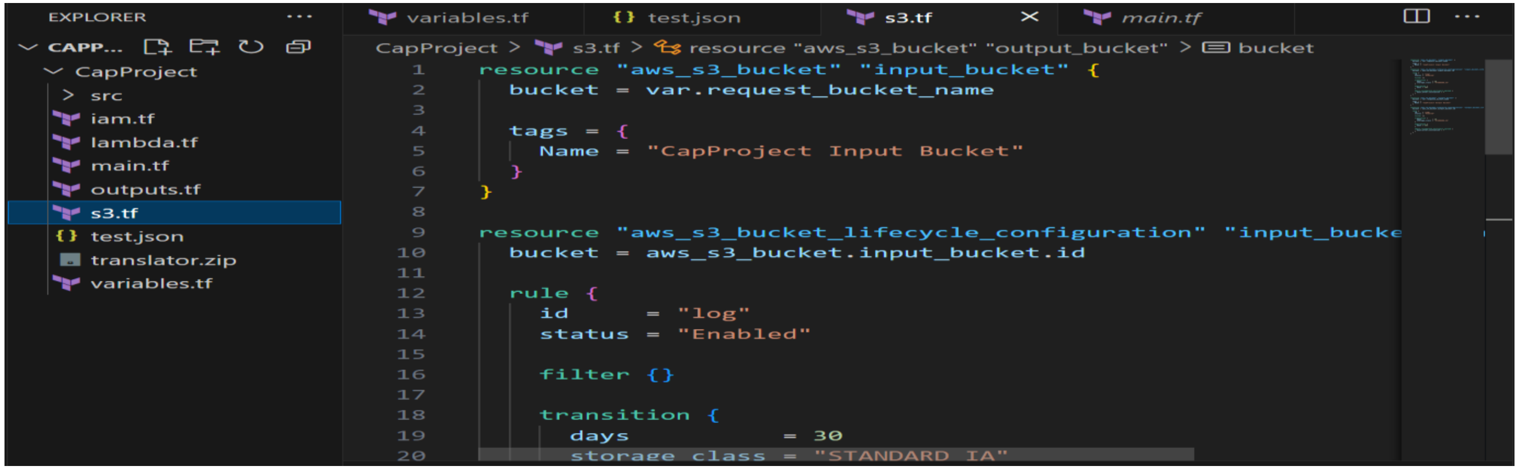
S3 Buckets with Lifecycle Policies

Two Amazon S3 buckets were provisioned using Terraform:

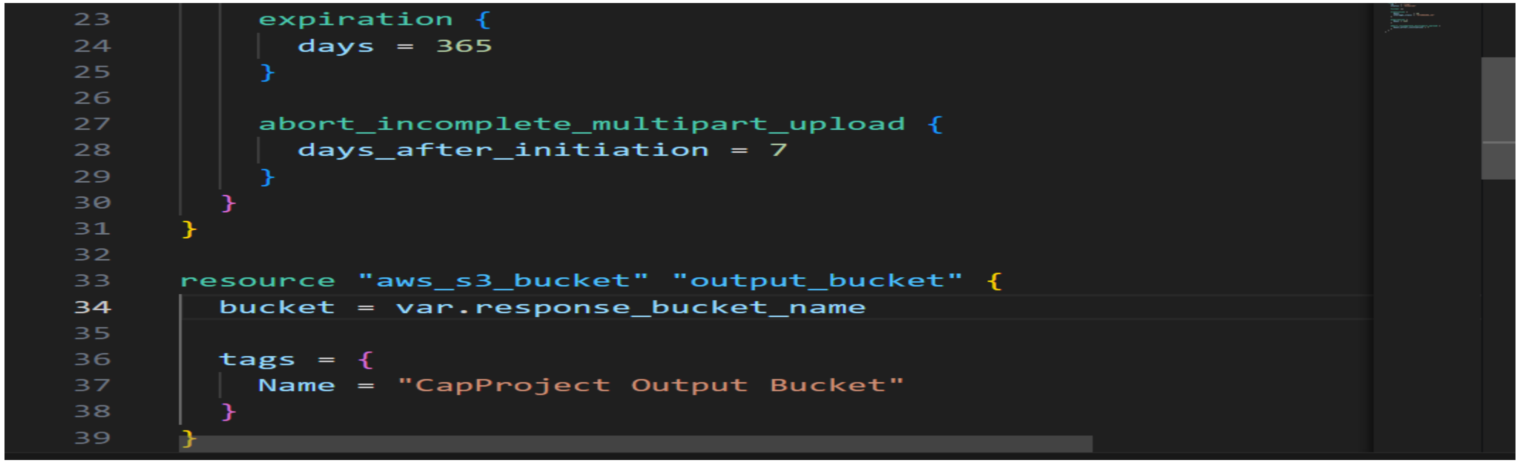
request-bucket. For storing the incoming JSON files requiring translation.

response-bucket. For storing the translated JSON files after processing.

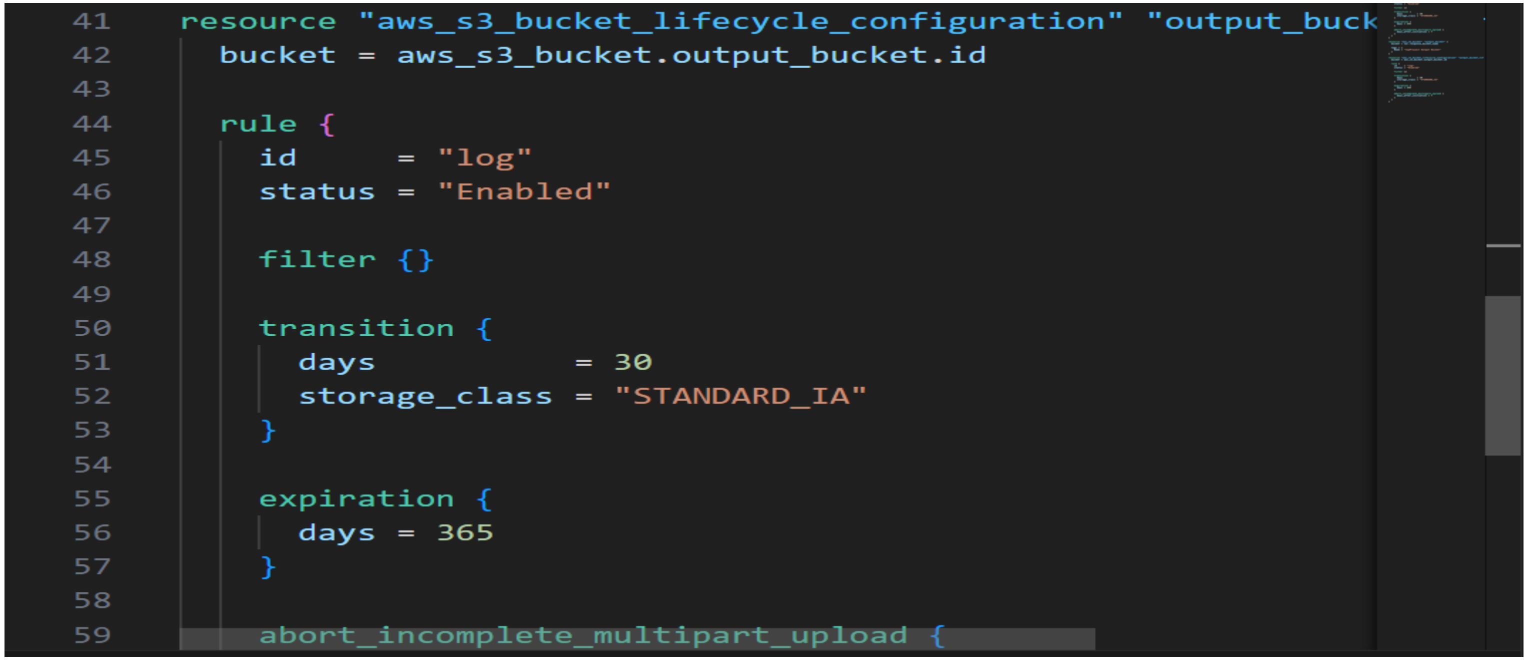
Lifecycle rules were added to automatically transition or expire files after a set period, ensuring cost optimization within Free Tier limits.



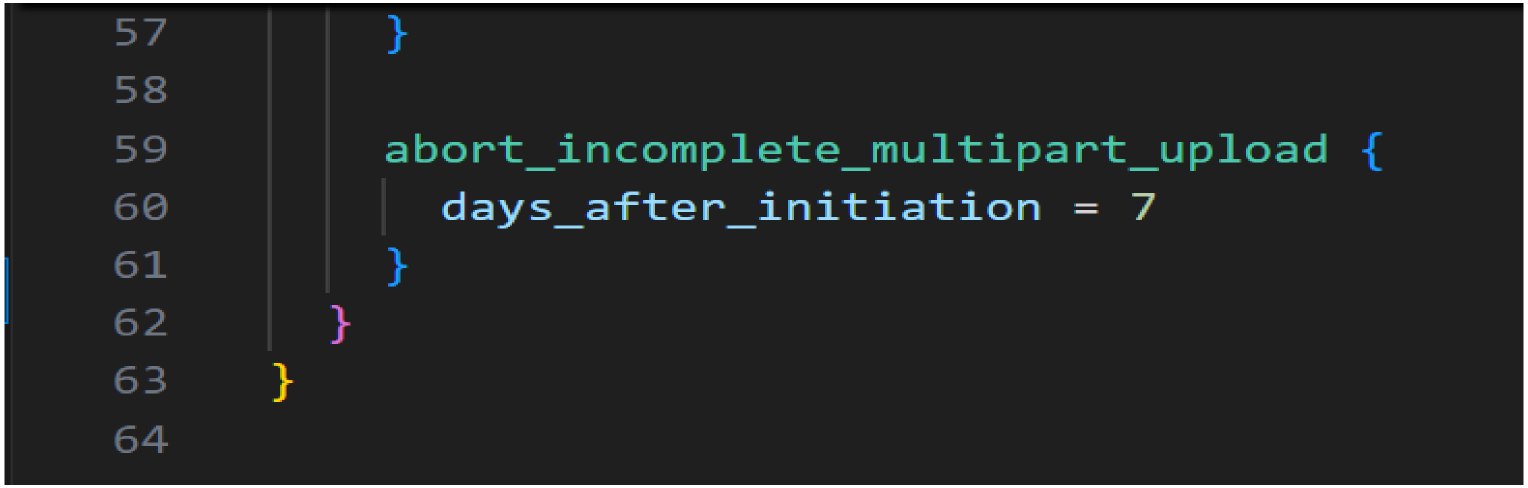
```
resource "aws_s3_bucket" "input_bucket" {
  bucket = var.request_bucket_name

  tags = {
    Name = "CapProject Input Bucket"
  }
}

resource "aws_s3_bucket_lifecycle_configuration" "input_bucke
  bucket = aws_s3_bucket.input_bucket.id

  rule {
    id     = "log"
    status = "Enabled"

    filter {}

    transition {
      days          = 30
      storage_class = "STANDARD_IA"
```



```
    expiration {
      days = 365
    }

    abort_incomplete_multipart_upload {
      days_after_initiation = 7
    }
  }
}

resource "aws_s3_bucket" "output_bucket" {
  bucket = var.response_bucket_name

  tags = {
    Name = "CapProject Output Bucket"
  }
}
```

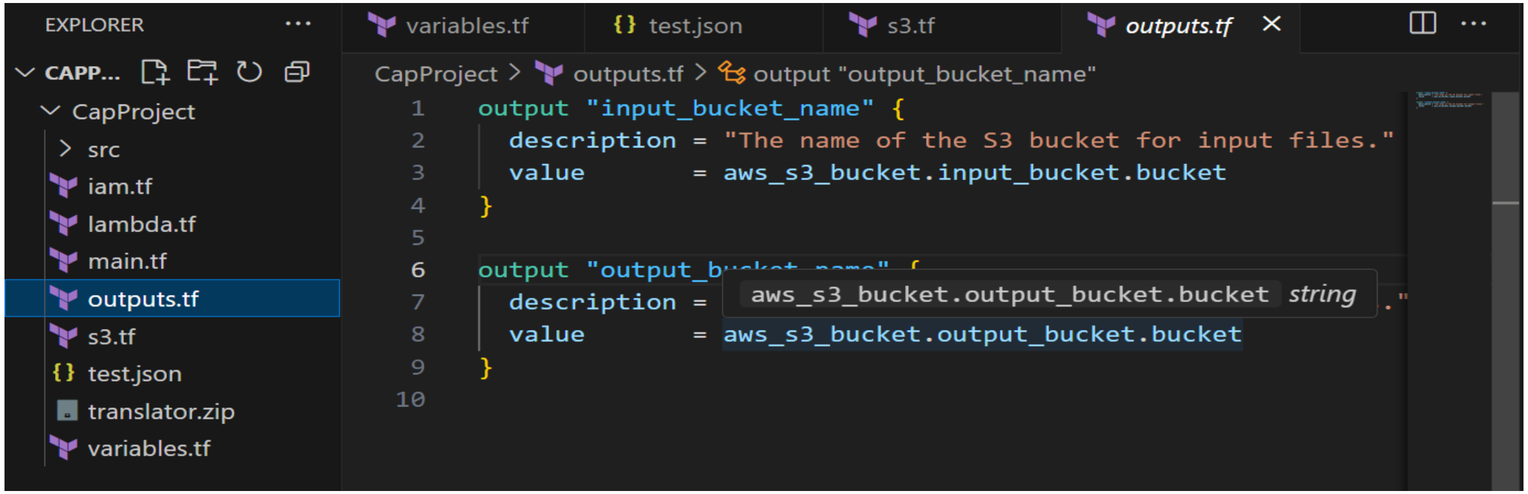


```
resource "aws_s3_bucket_lifecycle_configuration" "output_buck
  bucket = aws_s3_bucket.output_bucket.id

  rule {
    id     = "log"
    status = "Enabled"

    filter {}

    transition {
      days          = 30
      storage_class = "STANDARD_IA"
    }

    expiration {
      days = 365
    }

    abort_incomplete_multipart_upload {
```



```
    }

    abort_incomplete_multipart_upload {
      days_after_initiation = 7
    }
  }
}

```

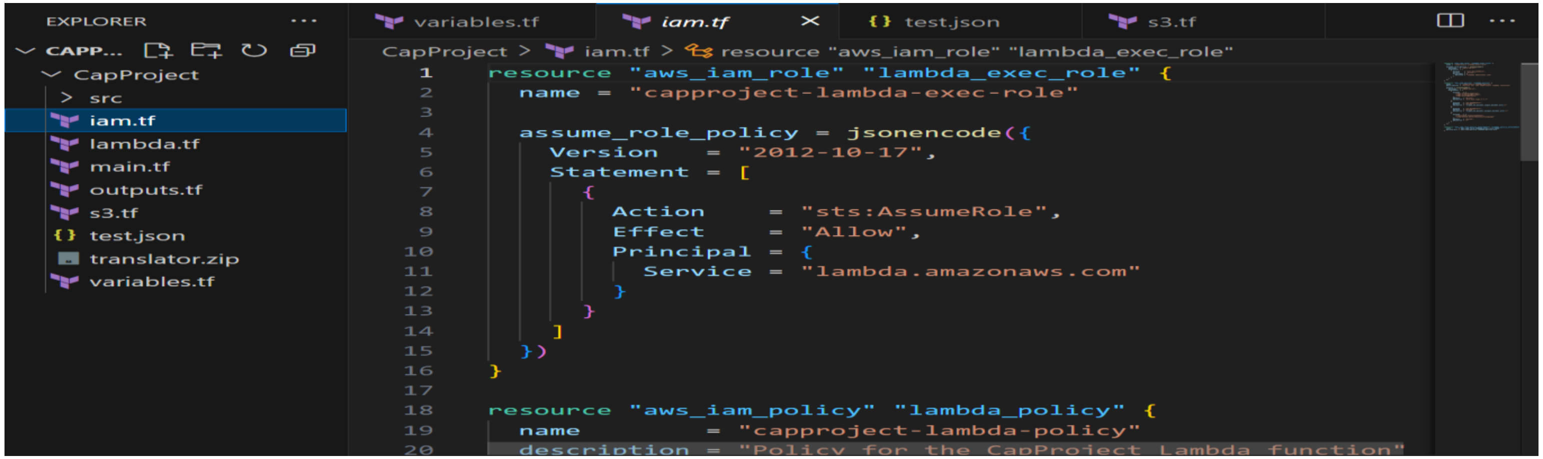
After provisioning the S3 buckets, Terraform outputs the names of the input and output buckets:



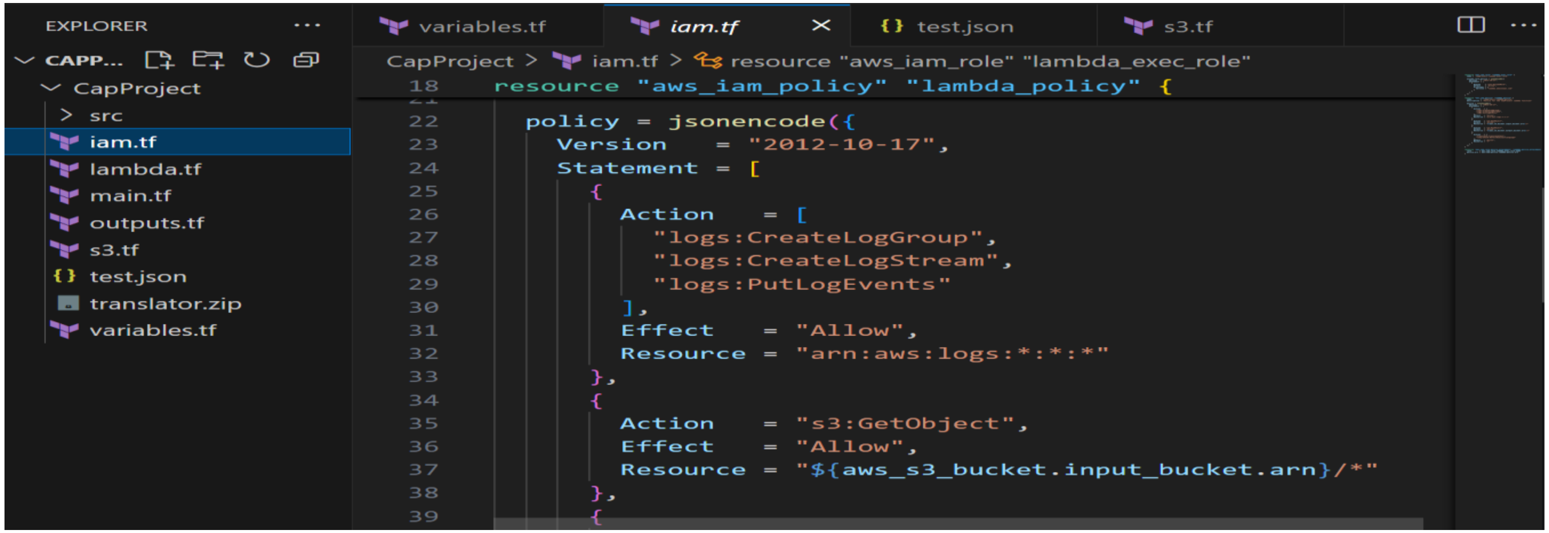
```
output "input_bucket_name" {
  description = "The name of the S3 bucket for input files."
  value       = aws_s3_bucket.input_bucket.bucket
}

output "output_bucket_name" {
  description =                                          ."
  value       = aws_s3_bucket.output_bucket.bucket
}

```

IAM Role and Policy for Lambda Execution

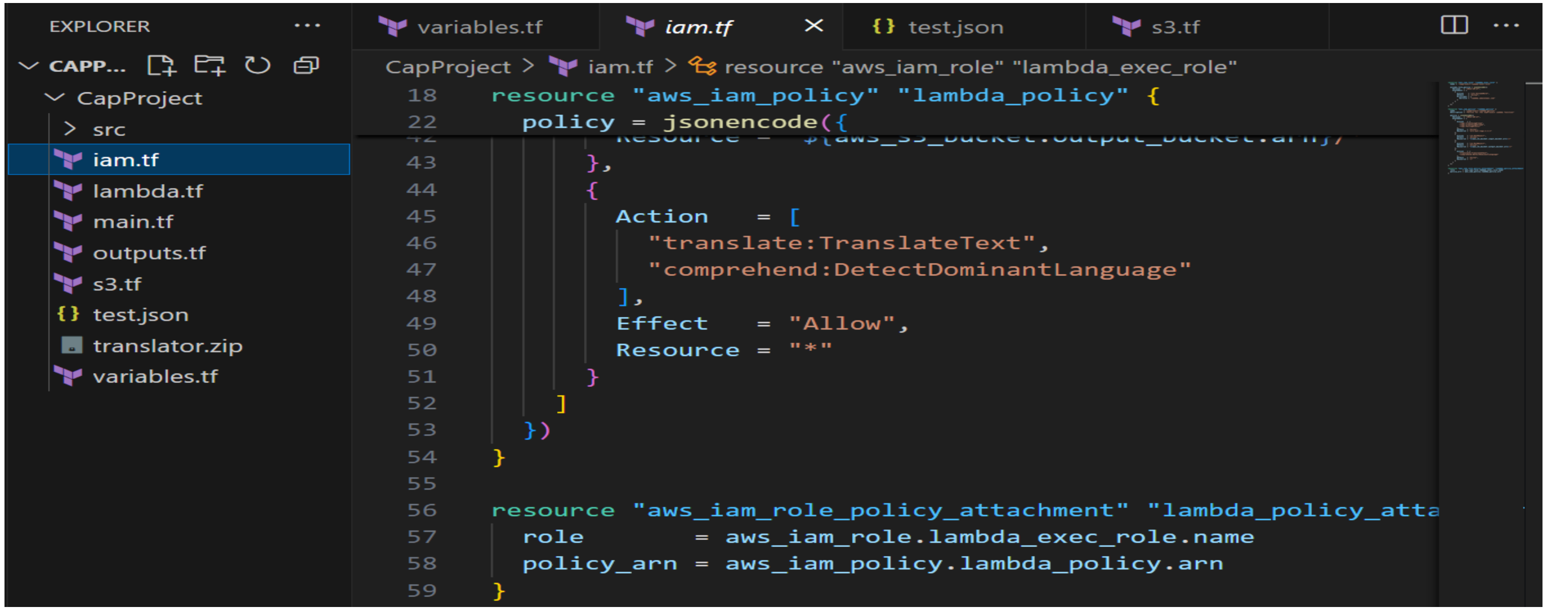
Using Terraform, a least-privilege IAM role and policy were provisioned for the Lambda function. This guaranteed that the function operated with only the permissions it needed.



```
resource "aws_iam_role" "lambda_exec_role" {
  name = "capproject-lambda-exec-role"

  assume_role_policy = jsonencode({
    Version   = "2012-10-17",
    Statement = [
      {
        Action    = "sts:AssumeRole",
        Effect    = "Allow",
        Principal = {
          Service = "lambda.amazonaws.com"
        }
      }
    ]
  })
}

resource "aws_iam_policy" "lambda_policy" {
  name        = "capproject-lambda-policy"
  description = "Policy for the CapProject Lambda function"
```



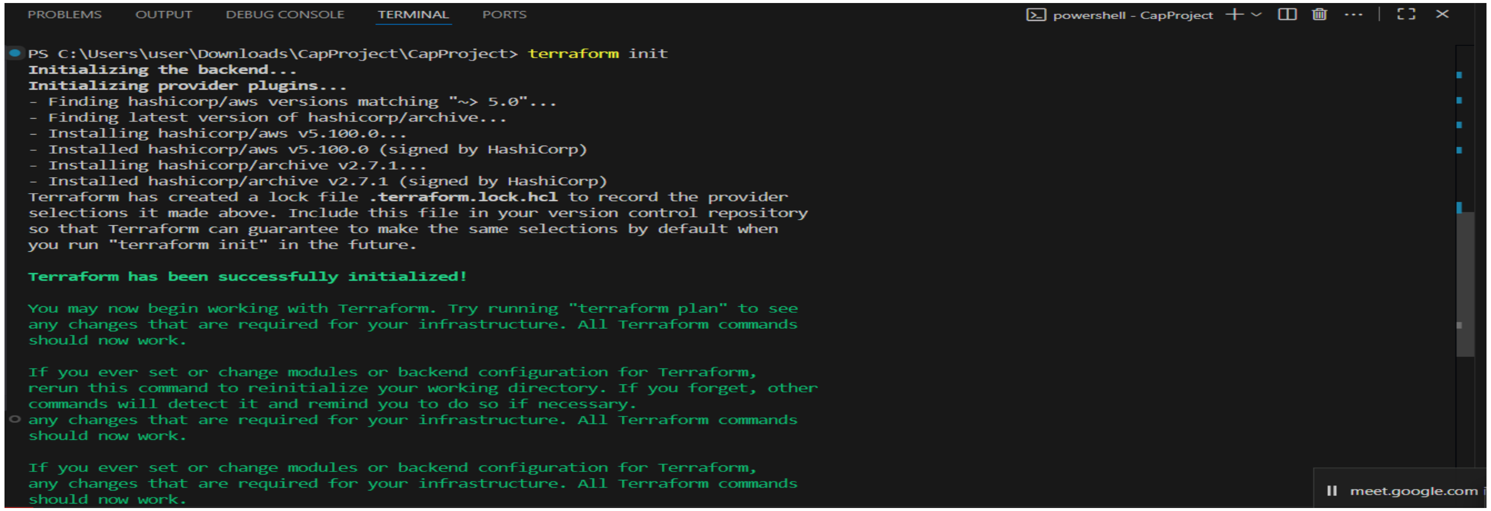
```
resource "aws_iam_policy" "lambda_policy" {
  policy = jsonencode({
    Version   = "2012-10-17",
    Statement = [
      {
        Action   = [
          "logs:CreateLogGroup",
          "logs:CreateLogStream",
          "logs:PutLogEvents"
        ],
        Effect   = "Allow",
        Resource = "arn:aws:logs:*:*:*"
      },
      {
        Action   = "s3:GetObject",
        Effect   = "Allow",
        Resource = "${aws_s3_bucket.input_bucket.arn}/*"
      },
      {
```



```
resource "aws_iam_policy" "lambda_policy" {
  policy = jsonencode({
        Resource = "${aws_s3_bucket.output_bucket.arn}/*"
      },
      {
        Action   = [
          "translate:TranslateText",
          "comprehend:DetectDominantLanguage"
        ],
        Effect   = "Allow",
        Resource = "*"
      }
    ]
  })
}

resource "aws_iam_role_policy_attachment" "lambda_policy_atta
  role       = aws_iam_role.lambda_exec_role.name
  policy_arn = aws_iam_policy.lambda_policy.arn
}
```

Validation

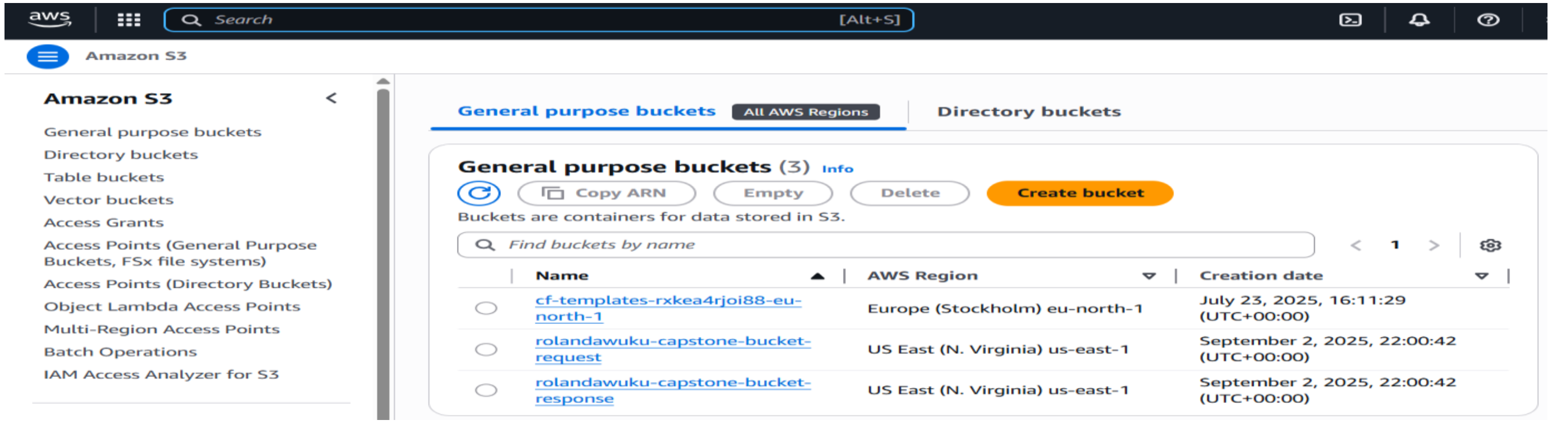
Ran *terraform* to deploy stack.



```
PS C:\Users\user\Downloads\CapProject\CapProject> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Finding latest version of hashicorp/archive...
- Installing hashicorp/aws v5.100.0...
- Installed hashicorp/aws v5.100.0 (signed by HashiCorp)
- Installing hashicorp/archive v2.7.1...
- Installed hashicorp/archive v2.7.1 (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.
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,
any changes that are required for your infrastructure. All Terraform commands
should now work.
```

Verified resources in the AWS Console

S3 buckets



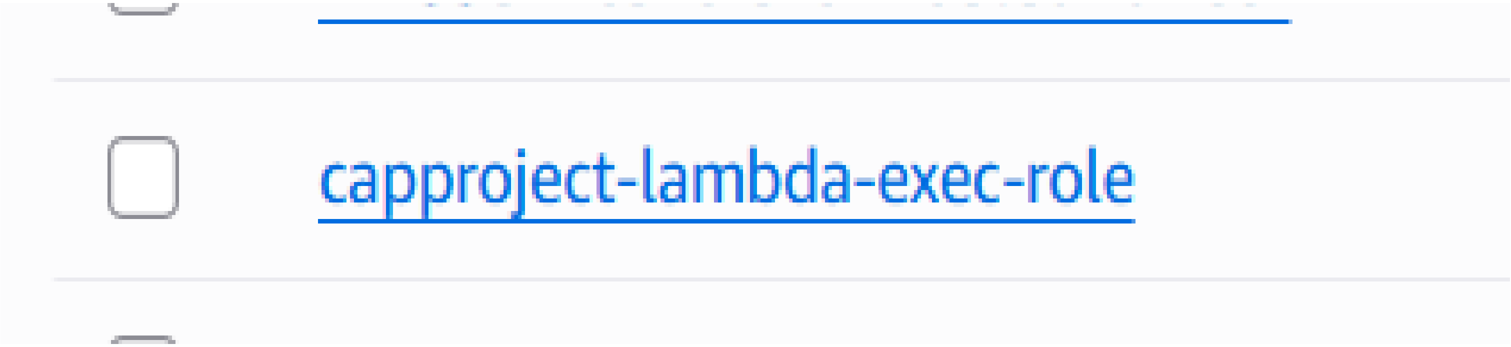
General purpose buckets (3)

Buckets are containers for data stored in S3.

	Name	AWS Region	Creation date
○	cf-templates-rxkea4rjoi88-eu-north-1	Europe (Stockholm) eu-north-1	July 23, 2025, 16:11:29 (UTC+00:00)
○	rolandawuku-capstone-bucket-request	US East (N. Virginia) us-east-1	September 2, 2025, 22:00:42 (UTC+00:00)
○	rolandawuku-capstone-bucket-response	US East (N. Virginia) us-east-1	September 2, 2025, 22:00:42 (UTC+00:00)

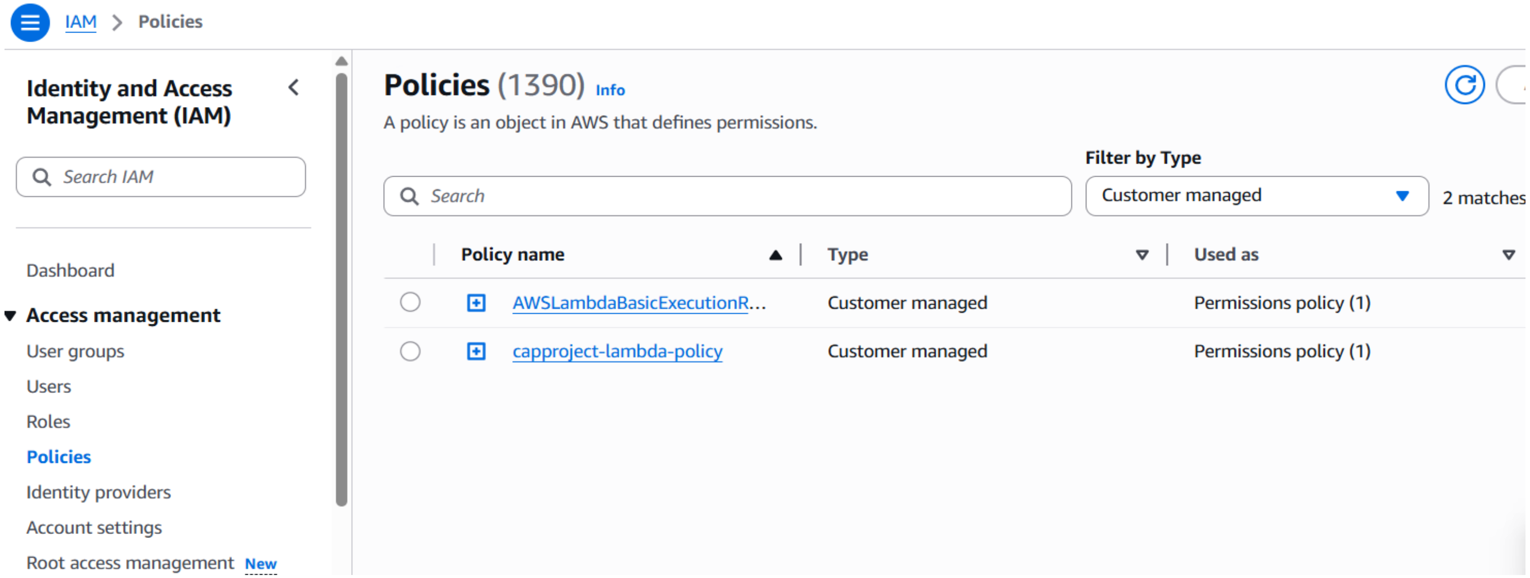
Lambda Function

Lambda execution role



□ capproject-lambda-exec-role

Lambda Policies



Policies (1390) Info

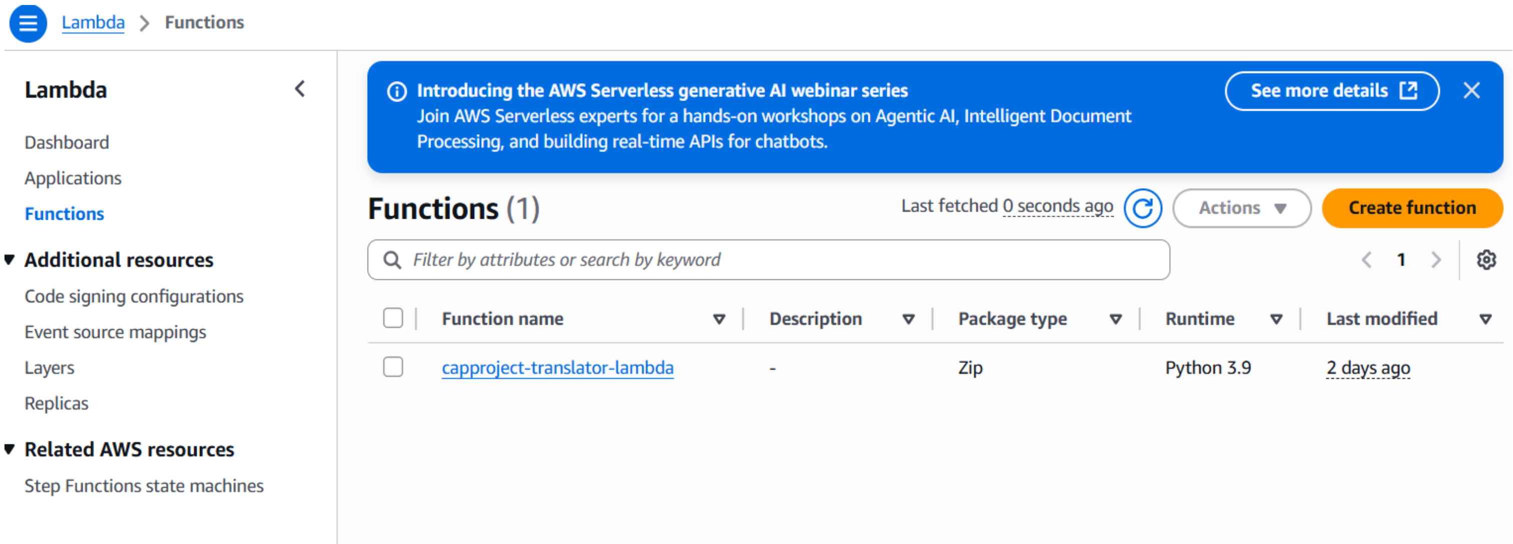
A policy is an object in AWS that defines permissions.

Filter by Type: Customer managed

2 matches

	Policy name	Type	Used as
○	AWSLambdaBasicExecutionR...	Customer managed	Permissions policy (1)
○	capproject-lambda-policy	Customer managed	Permissions policy (1)

Lambda function



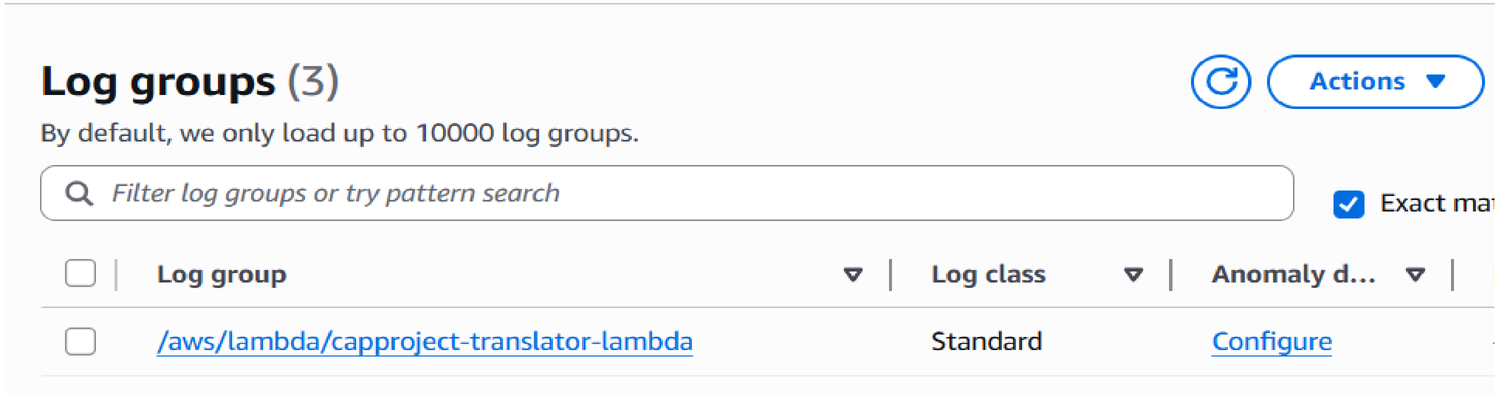
Introducing the AWS Serverless generative AI webinar series
Join AWS Serverless experts for a hands-on workshops on Agentic AI, Intelligent Document Processing, and building real-time APIs for chatbots.

Functions (1)

Last fetched 0 seconds ago

□	Function name	Description	Package type	Runtime	Last modified
□	capproject-translator-lambda	-	Zip	Python 3.9	2 days ago

Lambda Event Logs

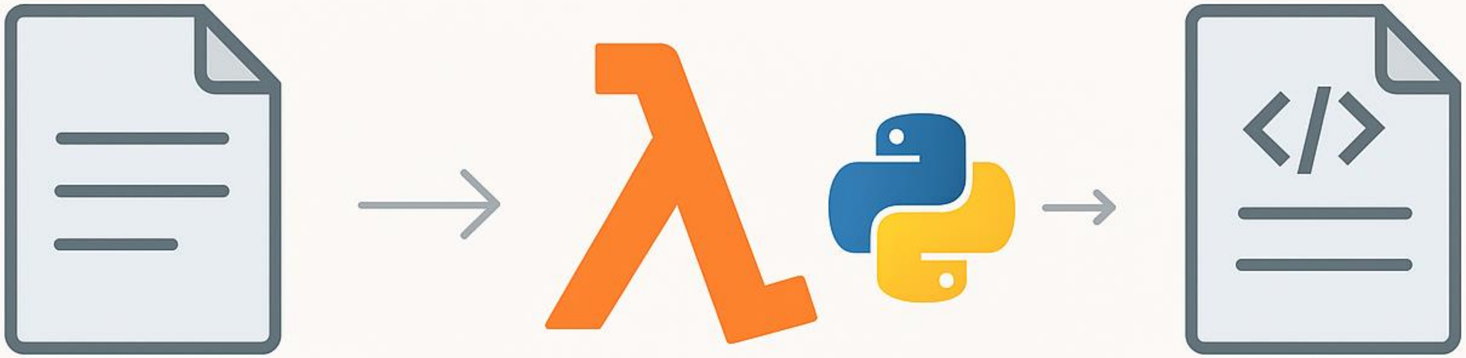


Log groups (3)

By default, we only load up to 10000 log groups.

□	Log group	Log class	Anomaly d...
□	/aws/lambda/capproject-translator-lambda	Standard	Configure

PHASE 3: BACKEND SCRIPT DEVELOPMENT (TRANSLATION LOGIC



Input JSON

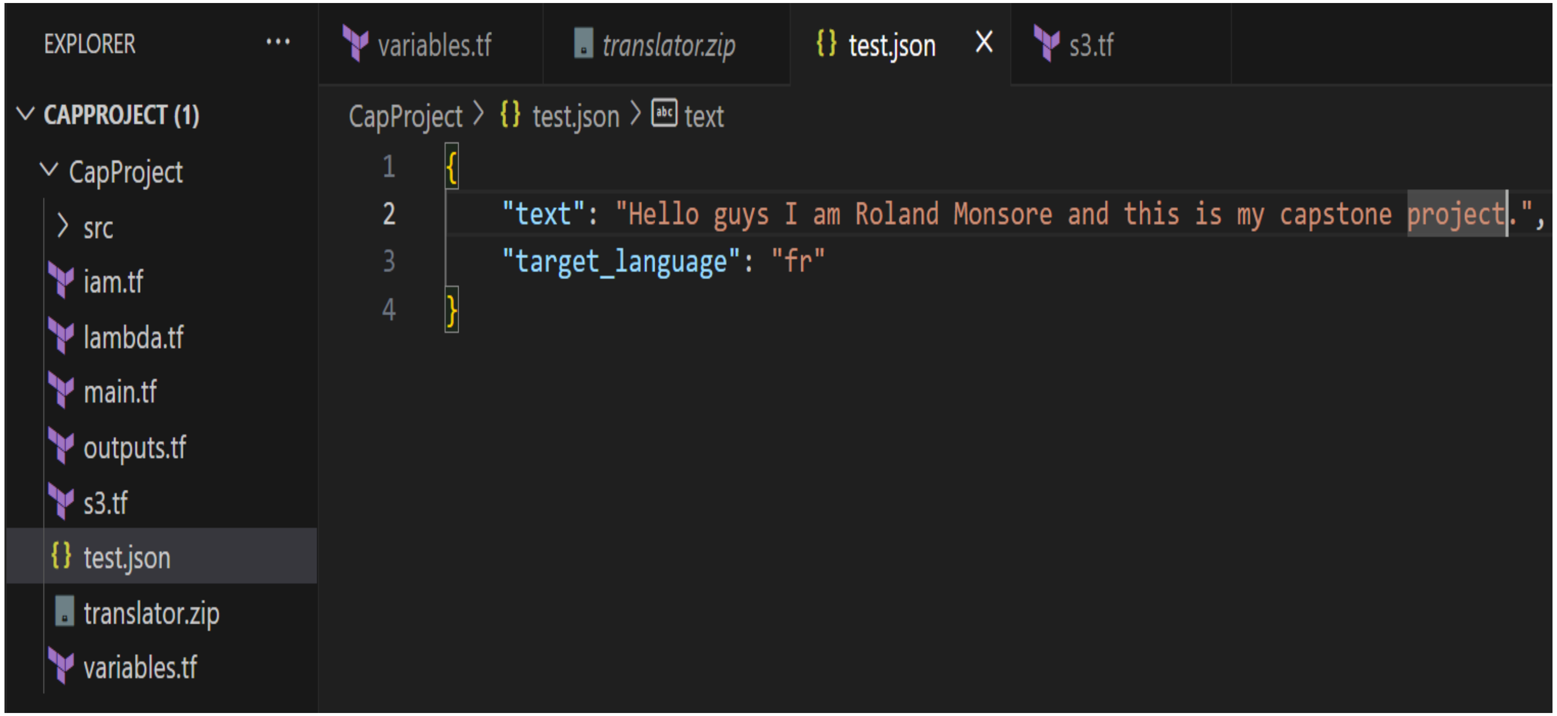
Output JSON

ACTIVITIES

Developed Python script (Lambda handler) using Boto3.

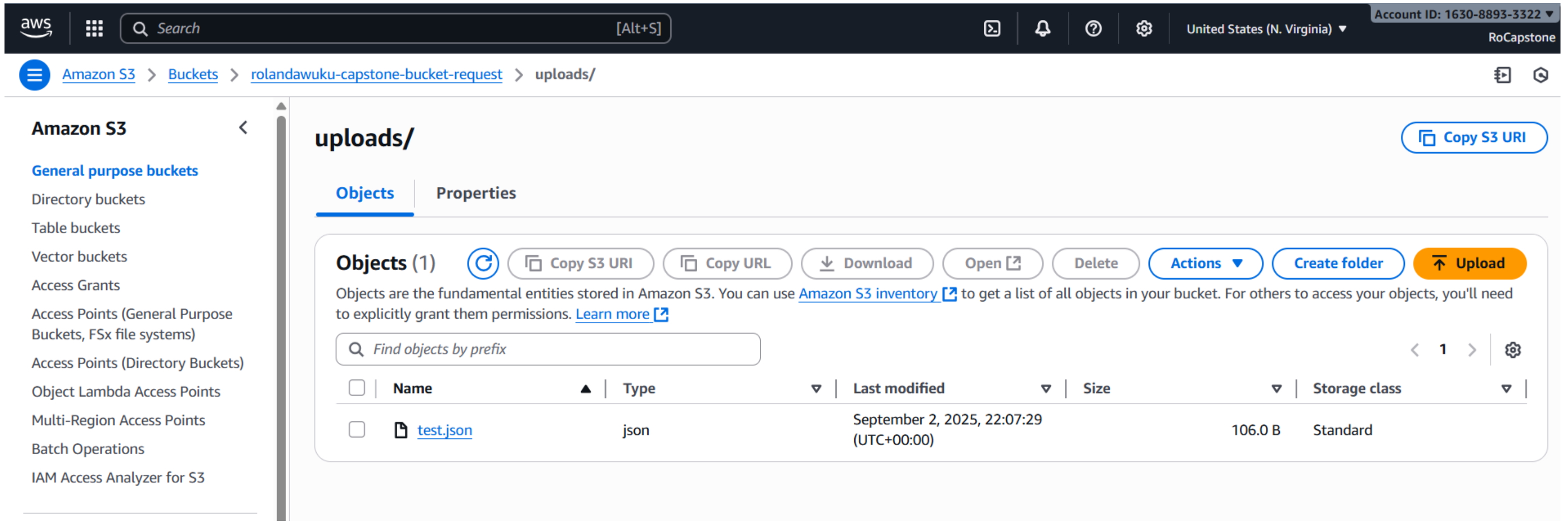
Reads input JSON file with text + target language. | Uses Amazon Translate API. | Stores translated JSON in response-bucket.

JSON input:



```json
{
    "text": "Hello guys I am Roland Monsore and this is my capstone project.",
    "target_language": "fr"
}
```

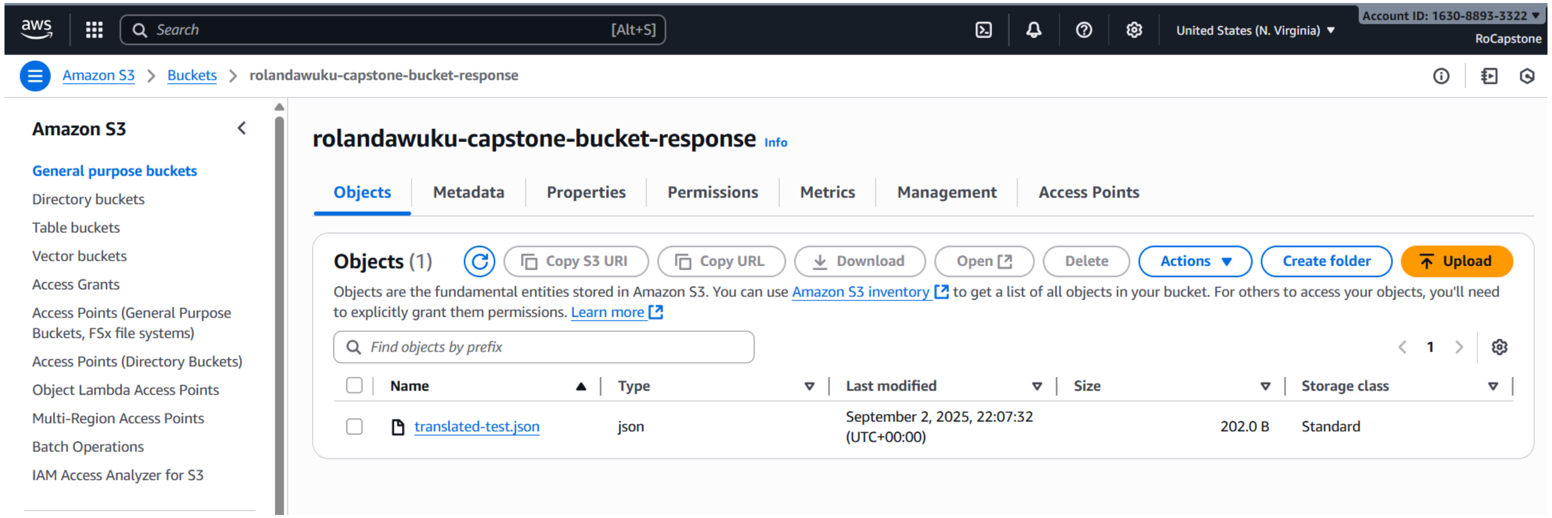
Picture of upload to request bucket:


aws
Search [Alt+S]
United States (N. Virginia)
Account ID: 1630-8893-3322
RoCapstone
Amazon S3 > Buckets > rolandawuku-capstone-bucket-request > uploads/
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Vector buckets
Access Grants
Access Points (General Purpose Buckets, FSx file systems)
Access Points (Directory Buckets)
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
uploads/
Copy S3 URI
Objects
Properties
Objects (1)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix
1

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

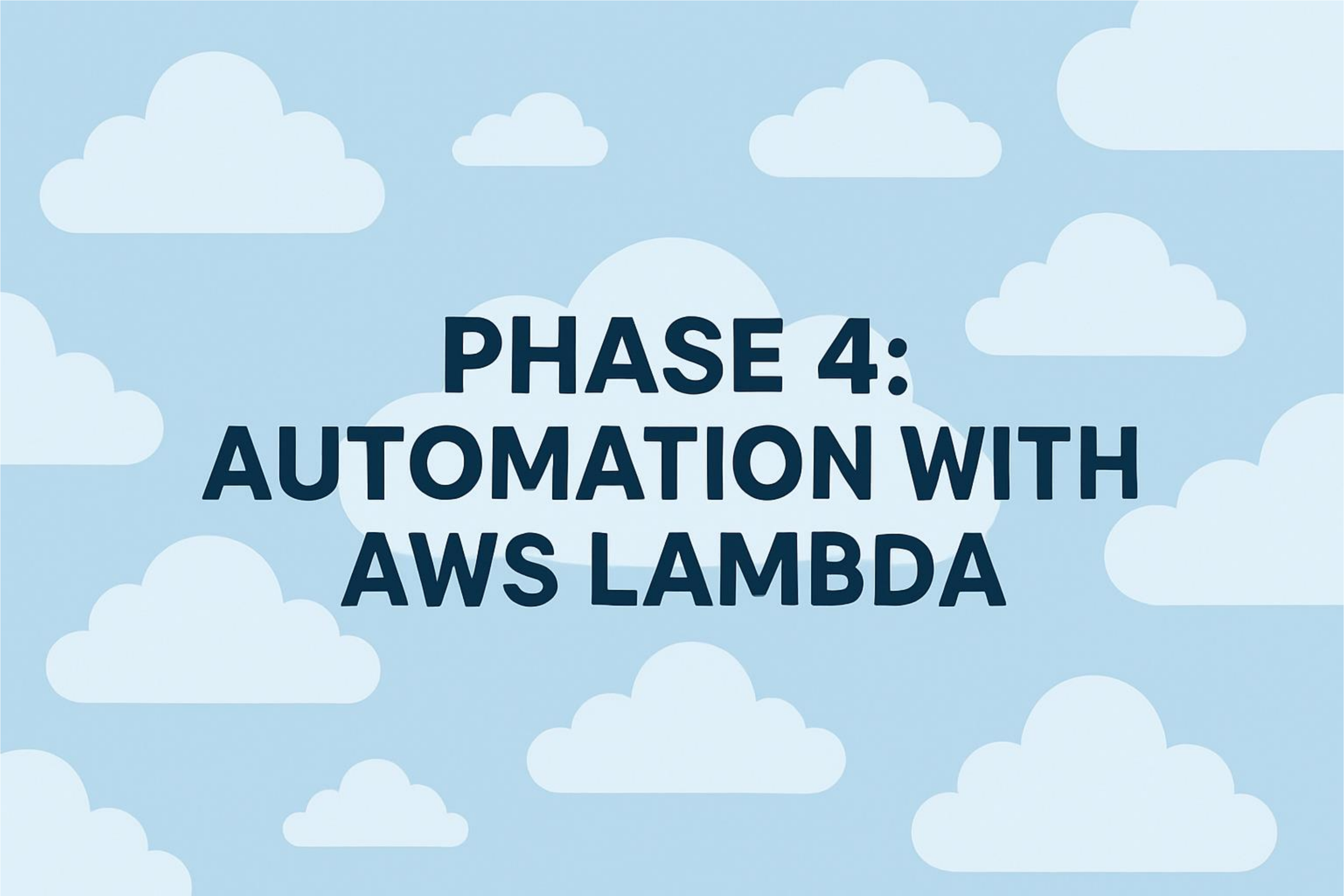
Name	Type	Last modified	Size	Storage class
test.json	json	September 2, 2025, 22:07:29 (UTC+00:00)	106.0 B	Standard

Translated JSON output file in response bucket:


aws
Search [Alt+S]
United States (N. Virginia)
Account ID: 1630-8893-3322
RoCapstone
Amazon S3 > Buckets > rolandawuku-capstone-bucket-response
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Vector buckets
Access Grants
Access Points (General Purpose Buckets, FSx file systems)
Access Points (Directory Buckets)
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
rolandawuku-capstone-bucket-response Info
Objects
Metadata
Properties
Permissions
Metrics
Management
Access Points
Objects (1)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix
1

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

Name	Type	Last modified	Size	Storage class
translated-test.json	json	September 2, 2025, 22:07:32 (UTC+00:00)	202.0 B	Standard

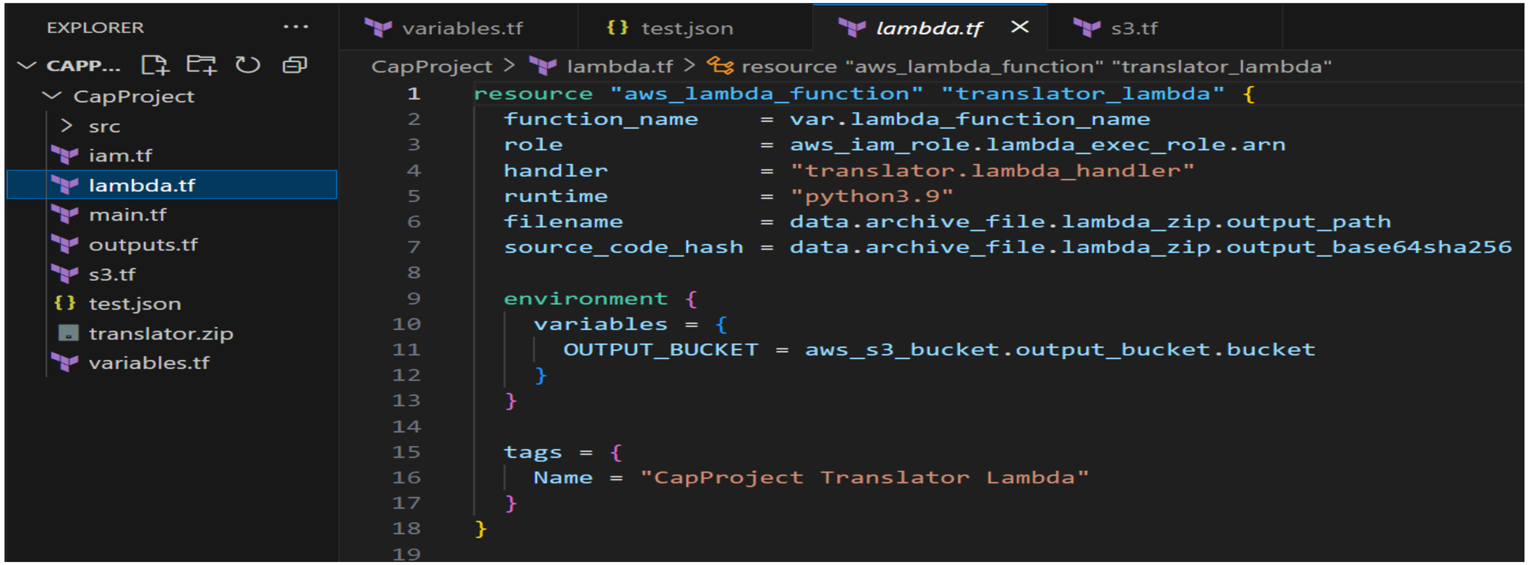


PHASE 4: AUTOMATION WITH AWS LAMBDA

Activities

Packaged the Python script using Terraform’s archive_file.

Deployed AWS Lambda function with: 256 MB memory | 60-second timeout | Environment variable pointing to output bucket



```
resource "aws_lambda_function" "translator_lambda" {
  function_name    = var.lambda_function_name
  role             = aws_iam_role.lambda_exec_role.arn
  handler          = "translator.lambda_handler"
  runtime          = "python3.9"
  filename         = data.archive_file.lambda_zip.output_path
  source_code_hash = data.archive_file.lambda_zip.output_base64sha256

  environment {
    variables = {
      OUTPUT_BUCKET = aws_s3_bucket.output_bucket.bucket
    }
  }

  tags = {
    Name = "CapProject Translator Lambda"
  }
}

```

Configured S3 event notification: Whenever a file is uploaded to the input bucket, it triggers Lambda automatically.

Automated flow:

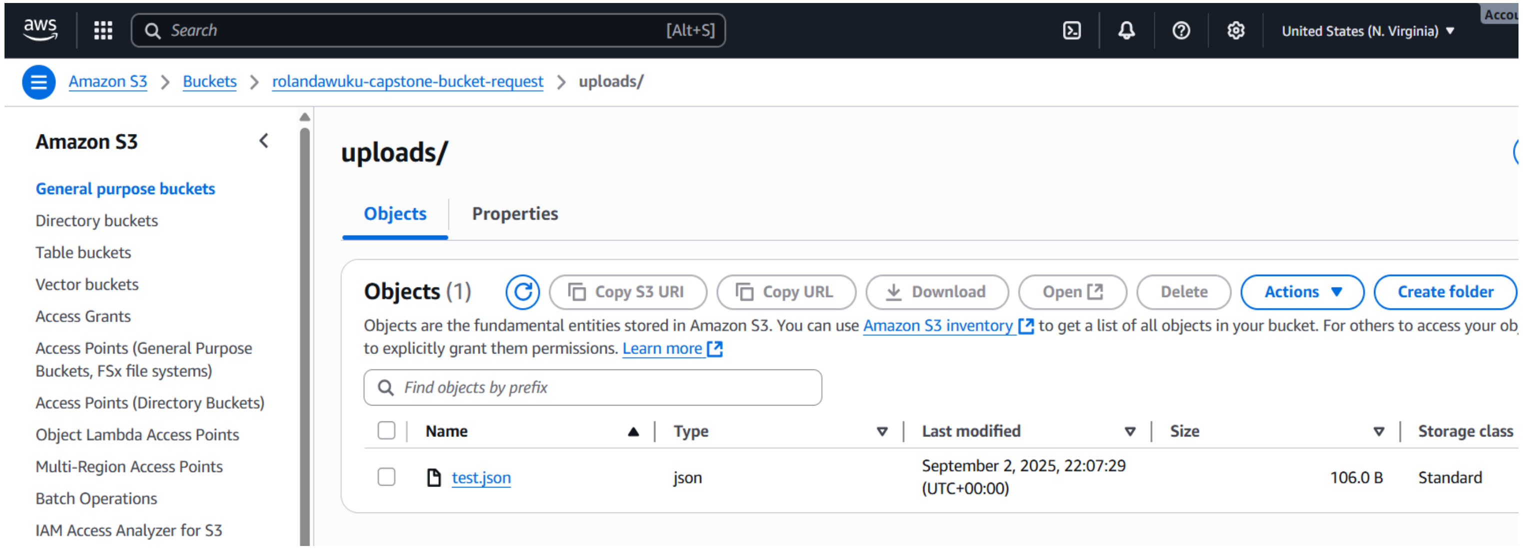
User uploads JSON → S3 triggers Lambda → Lambda (translator.py) translates → Result stored in response-bucket → CloudWatch logs capture execution details.

PHASE 5

TESTING, DEBUGGING & DOCUMENTATION

ACTIVITIES

Uploaded sample JSON files to input bucket.

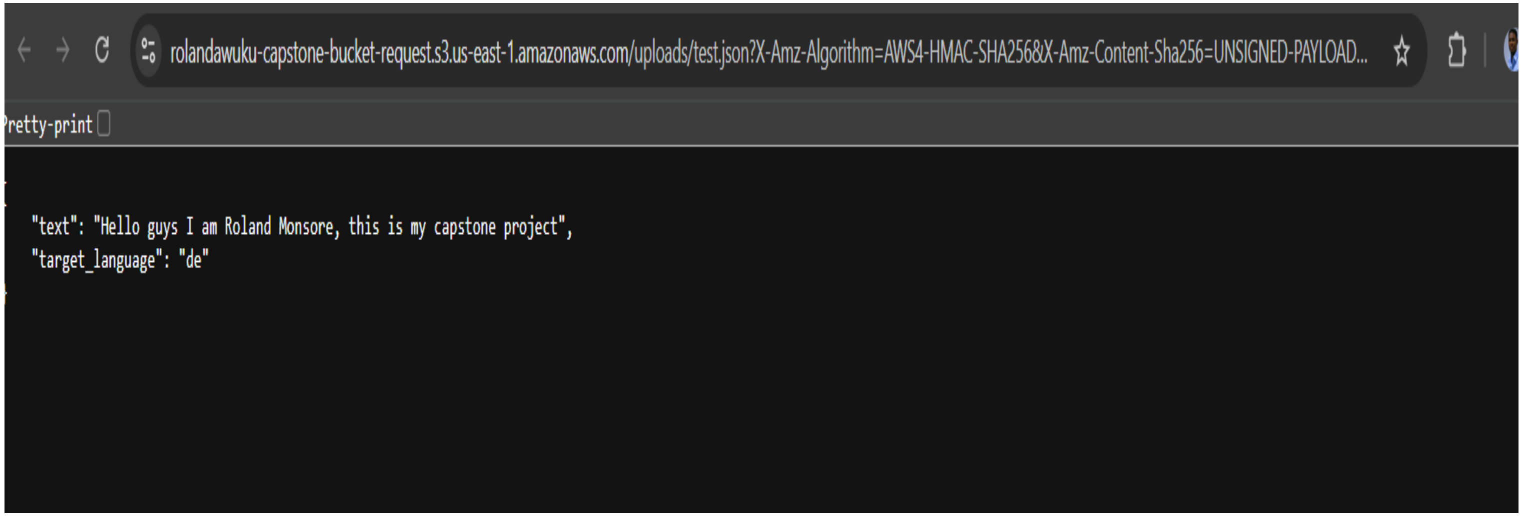

aws
Search [Alt+S]
United States (N. Virginia) ▼
Accou
Amazon S3 > Buckets > rolandawuku-capstone-bucket-request > uploads/
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Vector buckets
Access Grants
Access Points (General Purpose Buckets, FSx file systems)
Access Points (Directory Buckets)
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Objects
Properties
Copy S3 URI
Copy URL
Download
Open
Delete
Actions ▼
Create folder
Find objects by prefix

uploads/

Objects (1)

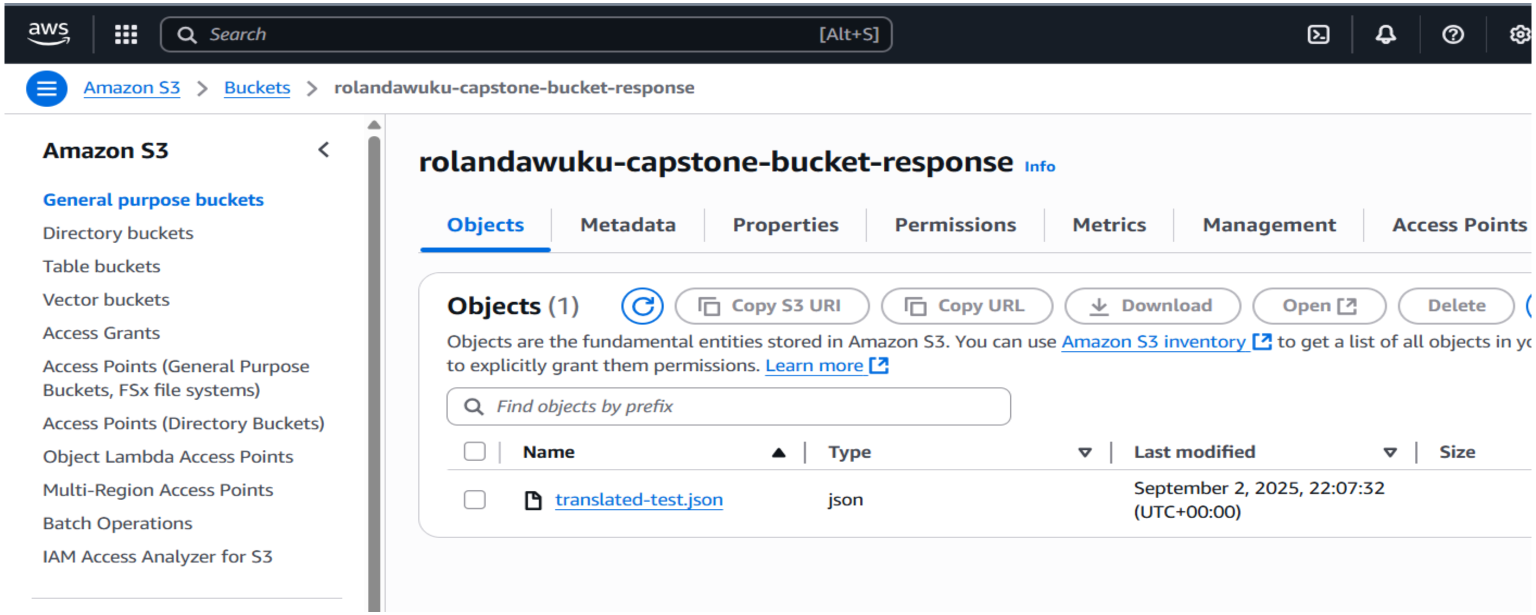
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your ob to explicitly grant them permissions. Learn more

□	Name ▲	Type	Last modified	Size	Storage class
□	test.json	json	September 2, 2025, 22:07:29 (UTC+00:00)	106.0 B	Standard


rolandawuku-capstone-bucket-request.s3.us-east-1.amazonaws.com/uploads/test.json?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Content-Sha256=UNSIGNED-PAYLOAD...
Pretty-print □

```
{
  "text": "Hello guys I am Roland Monsore, this is my capstone project",
  "target_language": "de"
}
```

Verified translation outputs.

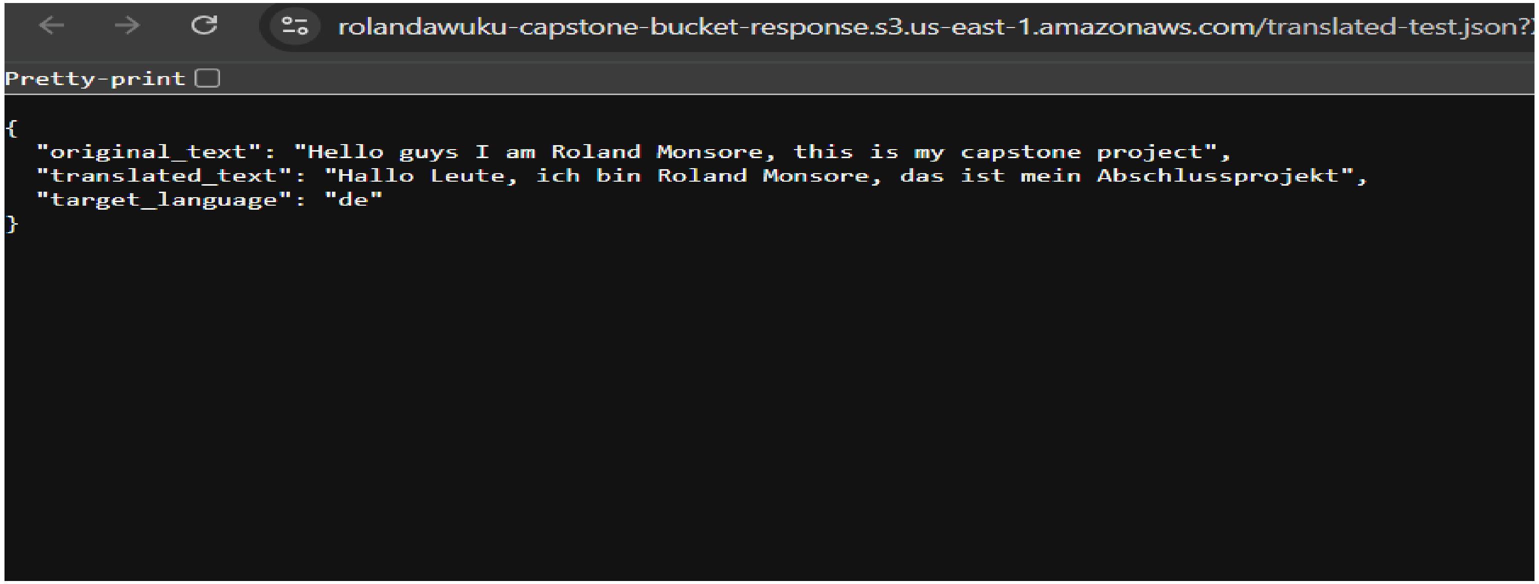


rolandawuku-capstone-bucket-response

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in yo
to explicitly grant them permissions. Learn more

□	Name	Type	Last modified	Size
□	translated-test.json	json	September 2, 2025, 22:07:32 (UTC+00:00)	

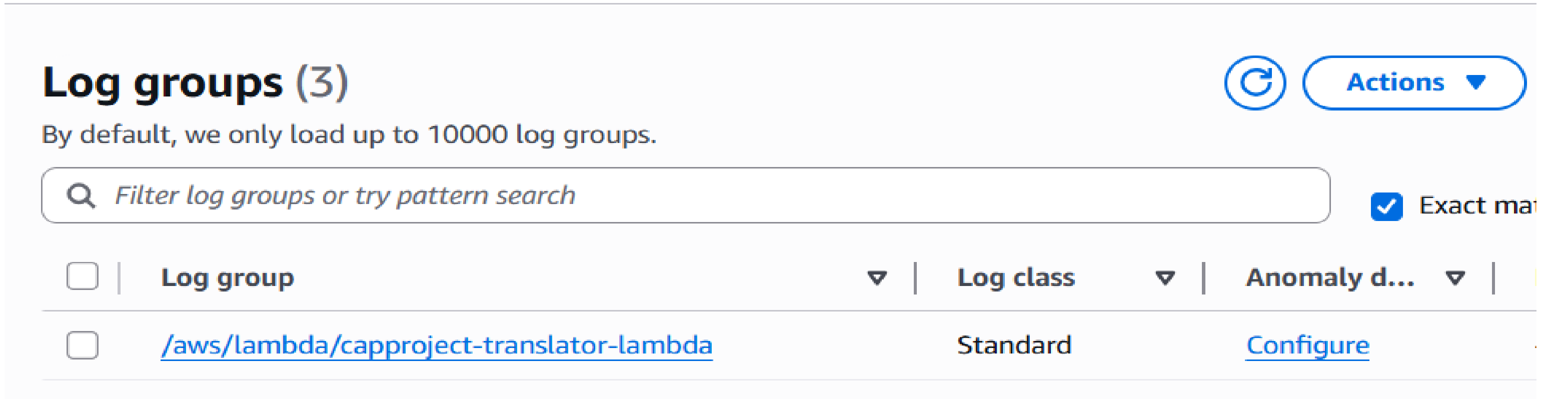


rolandawuku-capstone-bucket-response.s3.us-east-1.amazonaws.com/translated-test.json?

Pretty-print □

```
{
  "original_text": "Hello guys I am Roland Monsore, this is my capstone project",
  "translated_text": "Hallo Leute, ich bin Roland Monsore, das ist mein Abschlussprojekt",
  "target_language": "de"
}
```

Cloudwatch logs.



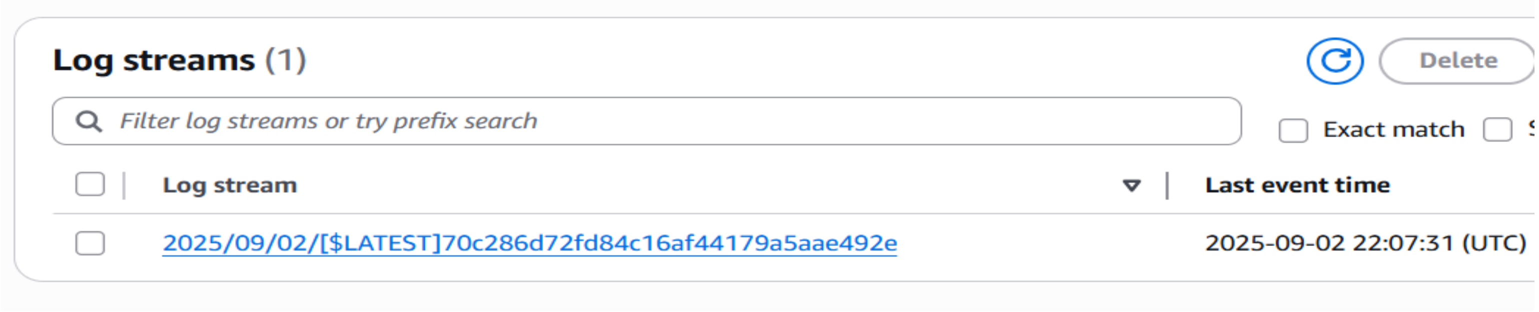
Log groups (3)

By default, we only load up to 10000 log groups.

Filter log groups or try pattern search

☑ Exact ma

□	Log group	Log class	Anomaly d...
□	/aws/lambda/capproject-translator-lambda	Standard	Configure

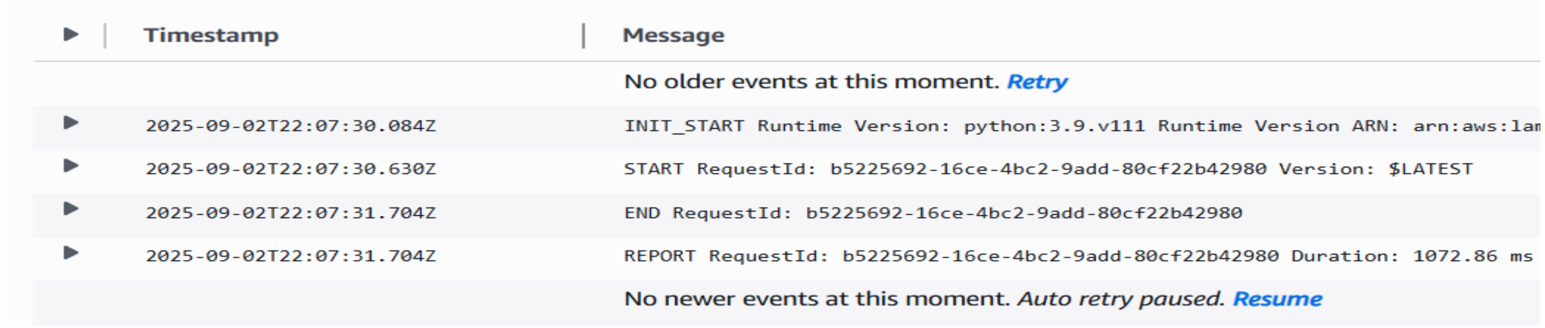


Log streams (1)

Filter log streams or try prefix search

□ Exact match □

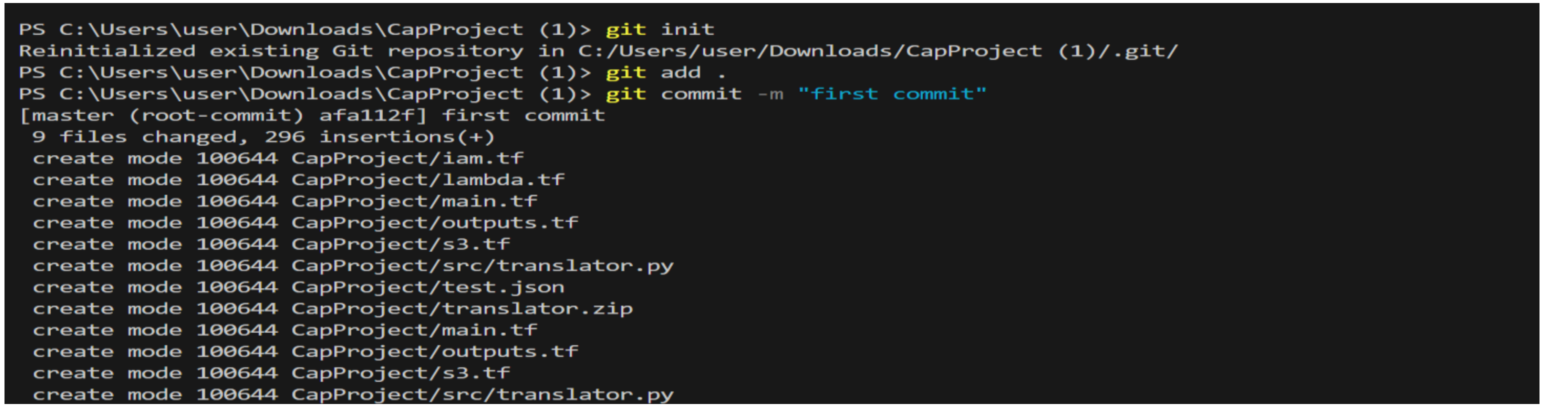
□	Log stream	Last event time
□	2025/09/02/[$LATEST]70c286d72fd84c16af44179a5aae492e	2025-09-02 22:07:31 (UTC)



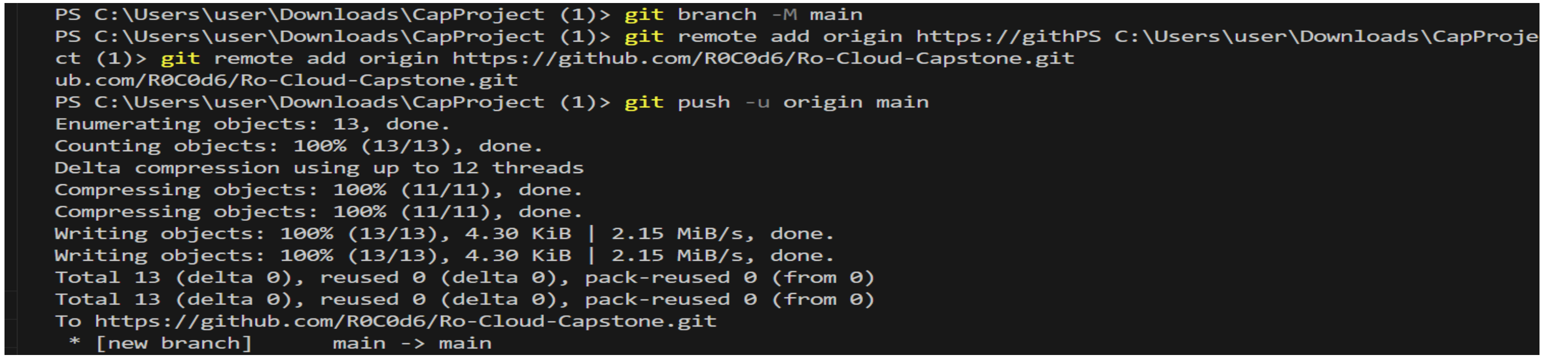
```
▶  Timestamp                   Message
                               No older events at this moment. Retry
▶  2025-09-02T22:07:30.084Z    INIT_START Runtime Version: python:3.9.v111 Runtime Version ARN: arn:aws:lam
▶  2025-09-02T22:07:30.630Z    START RequestId: b5225692-16ce-4bc2-9add-80cf22b42980 Version: $LATEST
▶  2025-09-02T22:07:31.704Z    END RequestId: b5225692-16ce-4bc2-9add-80cf22b42980
▶  2025-09-02T22:07:31.704Z    REPORT RequestId: b5225692-16ce-4bc2-9add-80cf22b42980 Duration: 1072.86 ms
                               No newer events at this moment. Auto retry paused. Resume
```

Finally ran terraform destroy to cleanup resources.

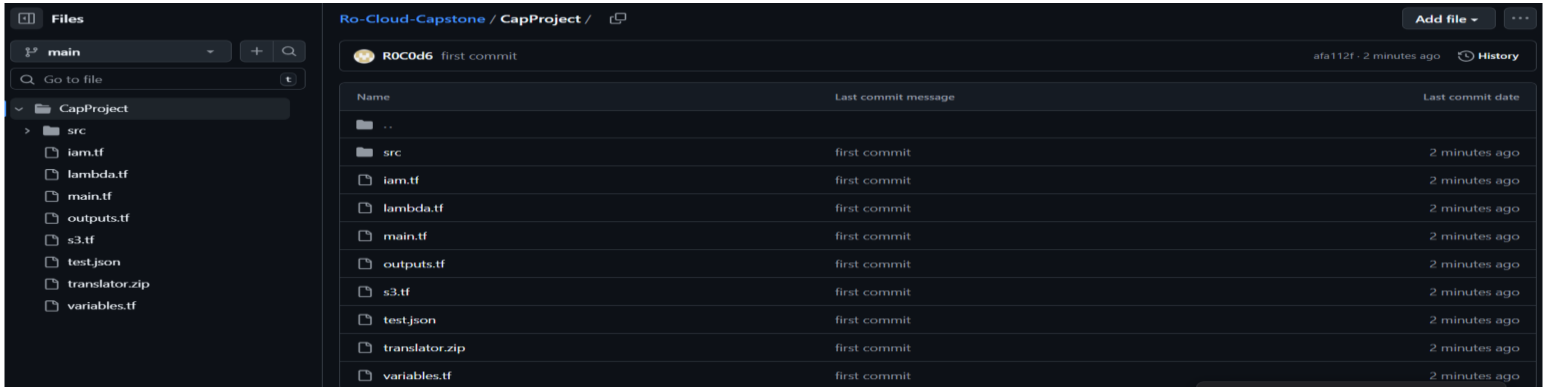
GITHUB REPOSITORY SETUP



```
PS C:\Users\user\Downloads\CapProject (1)> git init
Reinitialized existing Git repository in C:/Users/user/Downloads/CapProject (1)/.git/
PS C:\Users\user\Downloads\CapProject (1)> git add .
PS C:\Users\user\Downloads\CapProject (1)> git commit -m "first commit"
[master (root-commit) afa112f] first commit
 9 files changed, 296 insertions(+)
 create mode 100644 CapProject/iam.tf
 create mode 100644 CapProject/lambda.tf
 create mode 100644 CapProject/main.tf
 create mode 100644 CapProject/outputs.tf
 create mode 100644 CapProject/s3.tf
 create mode 100644 CapProject/src/translator.py
 create mode 100644 CapProject/test.json
 create mode 100644 CapProject/translator.zip
 create mode 100644 CapProject/main.tf
 create mode 100644 CapProject/outputs.tf
 create mode 100644 CapProject/s3.tf
 create mode 100644 CapProject/src/translator.py
```



```
PS C:\Users\user\Downloads\CapProject (1)> git branch -M main
PS C:\Users\user\Downloads\CapProject (1)> git remote add origin https://githPS C:\Users\user\Downloads\CapProje
ct (1)> git remote add origin https://github.com/R0C0d6/Ro-Cloud-Capstone.git
ub.com/R0C0d6/Ro-Cloud-Capstone.git
PS C:\Users\user\Downloads\CapProject (1)> git push -u origin main
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 12 threads
Compressing objects: 100% (11/11), done.
Compressing objects: 100% (11/11), done.
Writing objects: 100% (13/13), 4.30 KiB | 2.15 MiB/s, done.
Writing objects: 100% (13/13), 4.30 KiB | 2.15 MiB/s, done.
Total 13 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Total 13 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/R0C0d6/Ro-Cloud-Capstone.git
 * [new branch]      main -> main
```



R0C0d6 first commit afa112f · 2 minutes ago

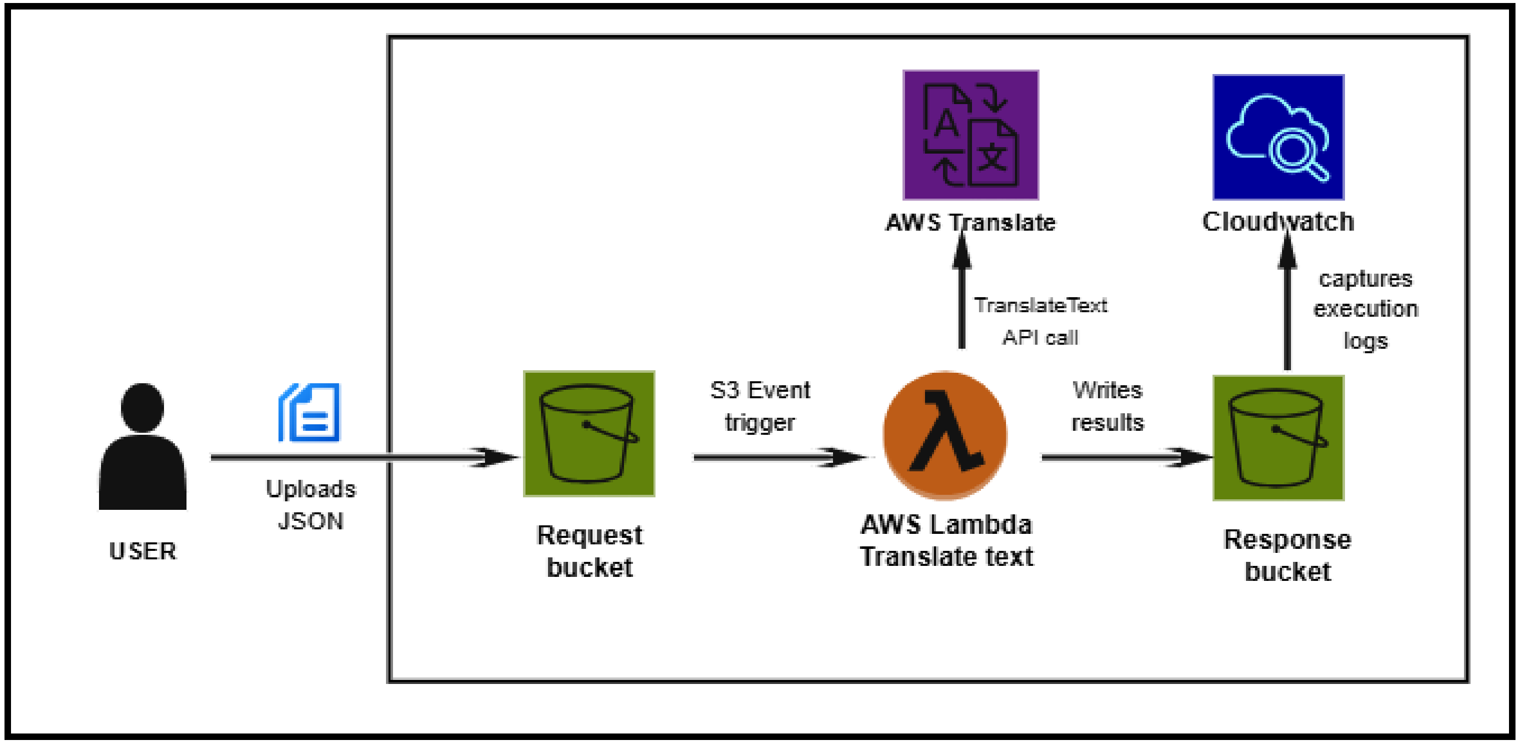
Name	Last commit message	Last commit date
..		
src	first commit	2 minutes ago
iam.tf	first commit	2 minutes ago
lambda.tf	first commit	2 minutes ago
main.tf	first commit	2 minutes ago
outputs.tf	first commit	2 minutes ago
s3.tf	first commit	2 minutes ago
test.json	first commit	2 minutes ago
translator.zip	first commit	2 minutes ago
variables.tf	first commit	2 minutes ago

CONCLUSION

The project successfully implemented a fully serverless translation pipeline that:

- Translates JSON files uploaded to Amazon S3 into a specified target language automatically.
- Saves both the original and translated versions securely in S3.
- Utilizes Amazon Comprehend to detect the source language dynamically.
- Captures execution details and errors in CloudWatch for monitoring and troubleshooting.
- Uses Terraform for reproducible deployments, ensuring consistent infrastructure provisioning.
- Runs within AWS Free Tier limits when handling small-scale test data.
- The resulting system delivers a scalable, cost-effective, and automated solution for real-time language translation, highlighting best practices in serverless computing, Infrastructure as Code (IaC), and cloud-native design.

ARCHITECTURE DIAGRAM


AWS Translate
Cloudwatch
TranslateText API call
captures execution logs
USER
Uploads JSON
Request bucket
S3 Event trigger
AWS Lambda Translate text
Writes results
Response bucket