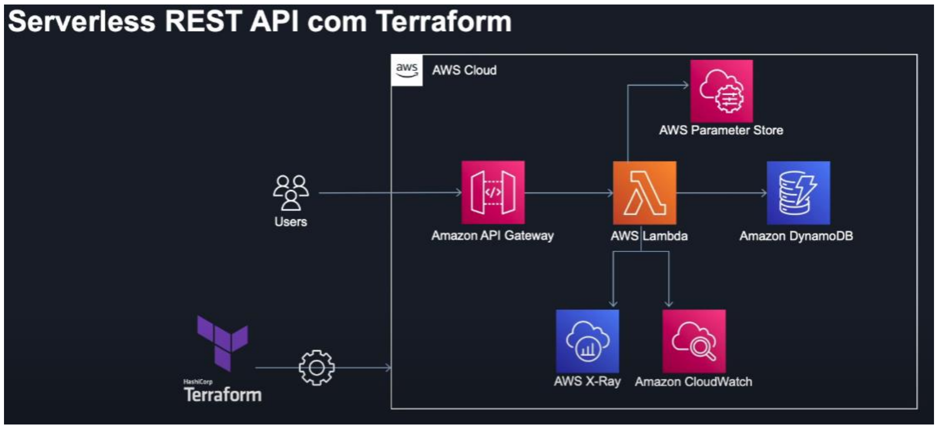
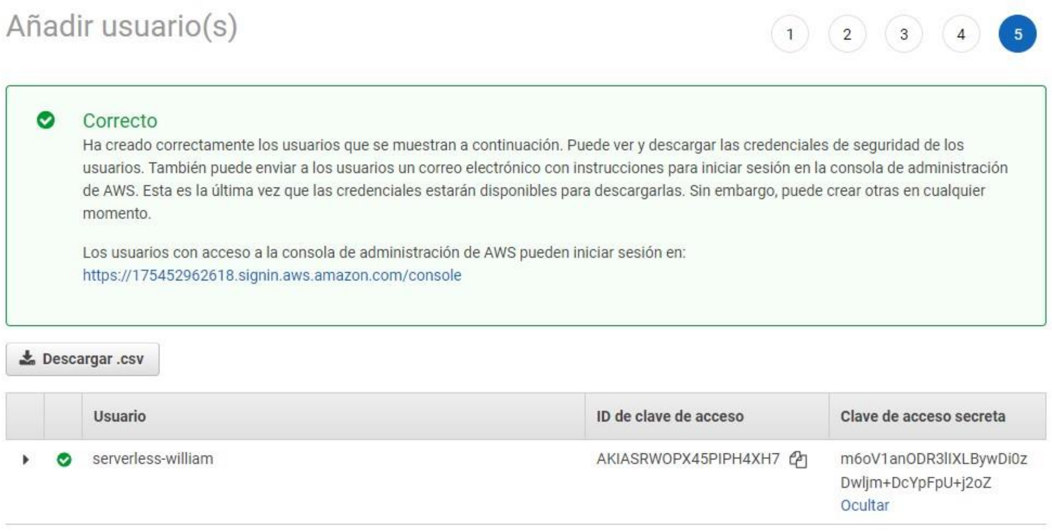


INTEGRANTES:

- William Andres Yaruro Cuan
- Jose Camilo Ricardo Viloria
- Brian Andres Acuña Donado

PRIMERO CREAR UN USUARIO Y GRUPO PARA ESTE TALLER.



Correcto

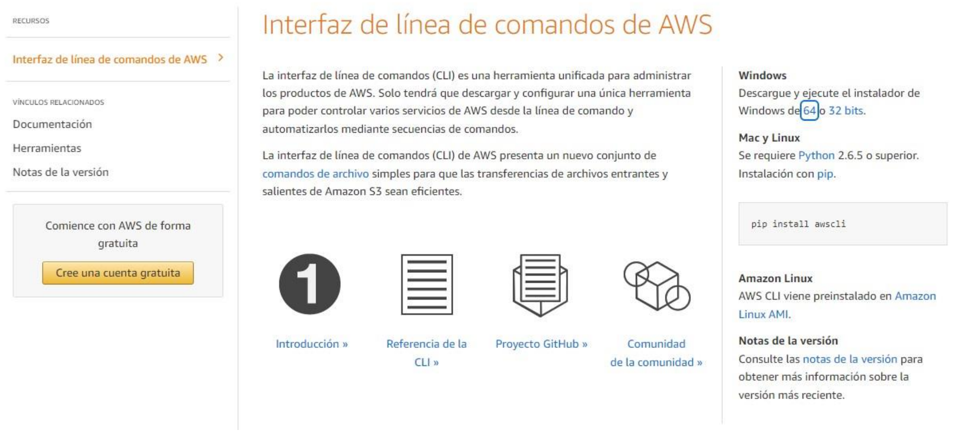
Ha creado correctamente los usuarios que se muestran a continuación. Puede ver y descargar las credenciales de seguridad de los usuarios. También puede enviar a los usuarios un correo electrónico con instrucciones para iniciar sesión en la consola de administración de AWS. Esta es la última vez que las credenciales estarán disponibles para descargarlas. Sin embargo, puede crear otras en cualquier momento.

Los usuarios con acceso a la consola de administración de AWS pueden iniciar sesión en: https://175452962618.signin.aws.amazon.com/console

Descargar .csv

Usuario	ID de clave de acceso	Clave de acceso secreta
serverless-william	AKIASRWOPX45PIPH4XH7	m6oV1anODR3lIXLBywDi0z Dwljm+DcYpFpU+j2oZ Ocultar

DESCARGAR E INSTALLAR AWS CLI


RECURSOS

Interfaz de línea de comandos de AWS >

VÍNCULOS RELACIONADOS

Documentación

Herramientas

Notas de la versión

Comience con AWS de forma gratuita

Cree una cuenta gratuita

Interfaz de línea de comandos de AWS

La interfaz de línea de comandos (CLI) es una herramienta unificada para administrar los productos de AWS. Solo tendrá que descargar y configurar una única herramienta para poder controlar varios servicios de AWS desde la línea de comando y automatizarlos mediante secuencias de comandos.

La interfaz de línea de comandos (CLI) de AWS presenta un nuevo conjunto de comandos de archivo simples para que las transferencias de archivos entrantes y salientes de Amazon S3 sean eficientes.

Introducción » · Referencia de la CLI » · Proyecto GitHub » · Comunidad de la comunidad »

Windows
Descargue y ejecute el instalador de Windows de 64 o 32 bits.

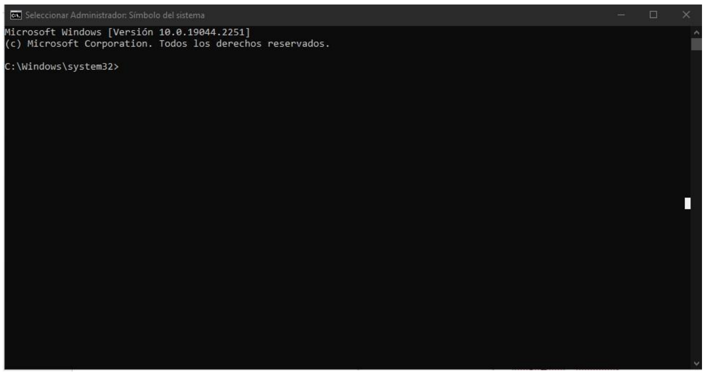
Mac y Linux
Se requiere Python 2.6.5 o superior.
Instalación con pip.

```
pip install awscli
```

Amazon Linux
AWS CLI viene preinstalado en Amazon Linux AMI.

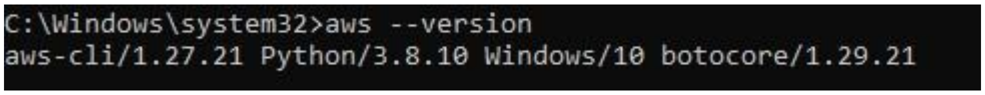
Notas de la versión
Consulte las notas de la versión para obtener más información sobre la versión más reciente.

Una vez finalizada la instalación vamos a proceder a abrir la consola de comandos (Windows + R) (Type: cmd)



```
Microsoft Windows [Versión 10.0.19044.2251]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Windows\system32>
```

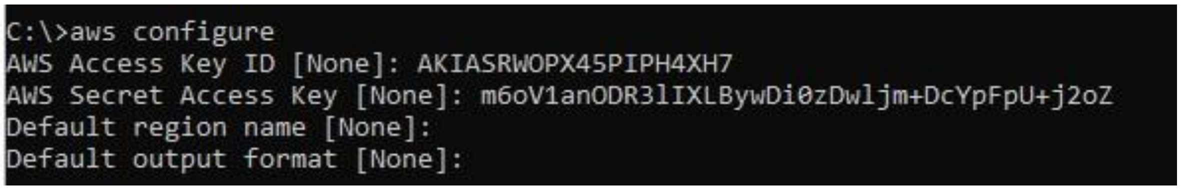
Verificamos que esté instalado correctamente aws CLI



```
C:\Windows\system32>aws --version
aws-cli/1.27.21 Python/3.8.10 Windows/10 botocore/1.29.21
```

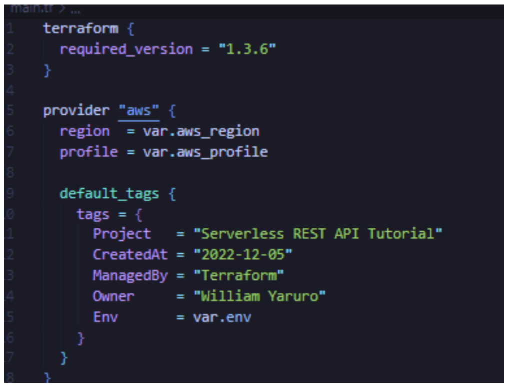
Utilizamos el cmd "aws configure" para configurar aws….

Configuramos nuestras credenciales de nuestro usuario…



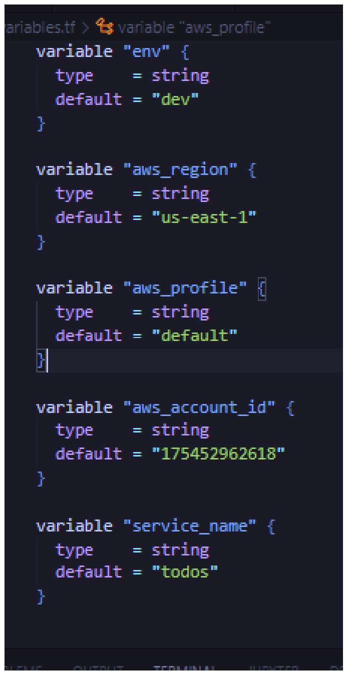
```
C:\>aws configure
AWS Access Key ID [None]: AKIASRWOPX45PIPH4XH7
AWS Secret Access Key [None]: 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
Default region name [None]:
Default output format [None]:
```

Configuramos nuestro archivo main.tf:



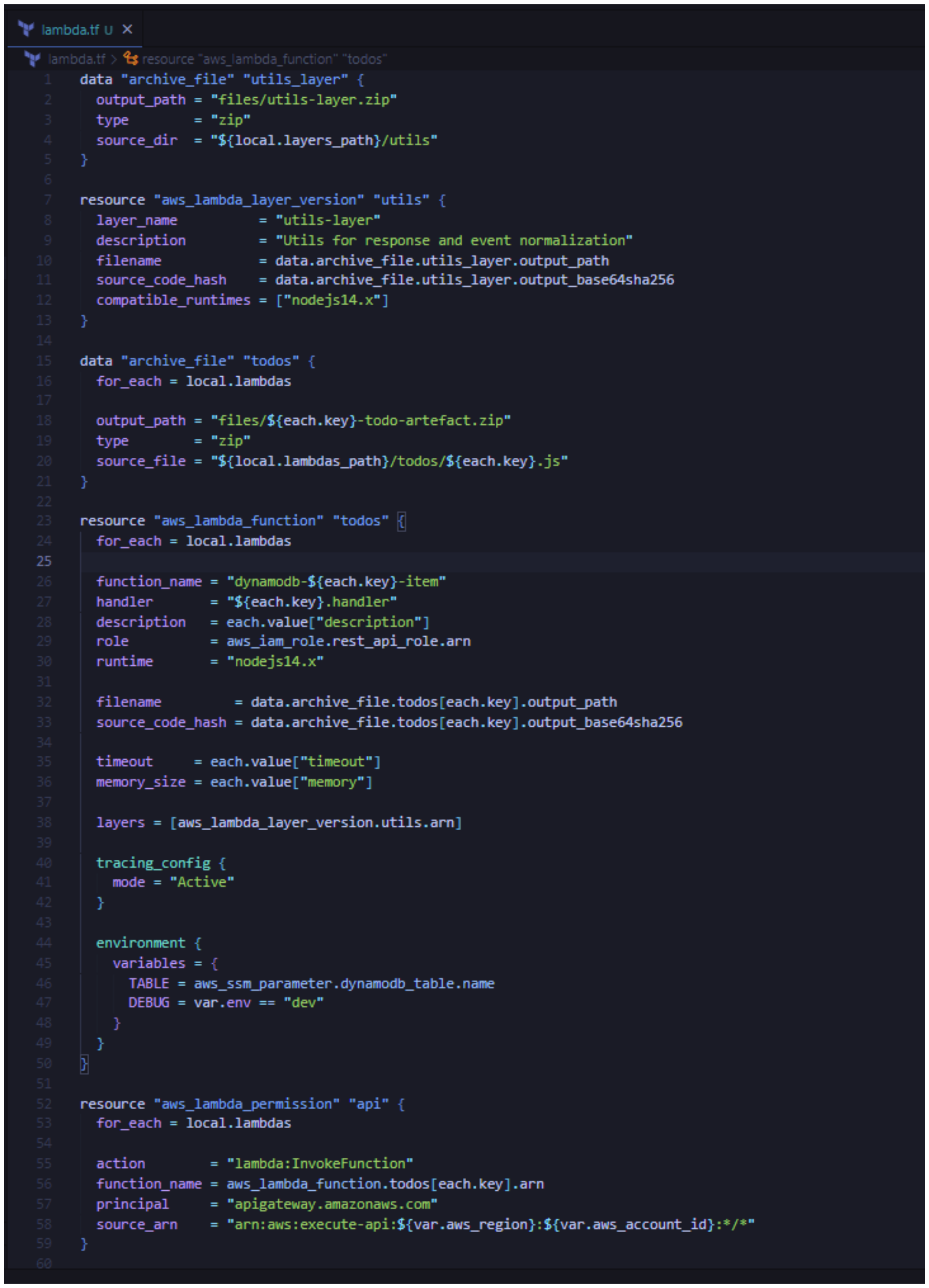
```
main.tf > ...
terraform {
  required_version = "1.3.6"
}

provider "aws" {
  region  = var.aws_region
  profile = var.aws_profile

  default_tags {
    tags = {
      Project   = "Serverless REST API Tutorial"
      CreatedAt = "2022-12-05"
      ManagedBy = "Terraform"
      Owner     = "William Yaruro"
      Env       = var.env
    }
  }
}
```

En un archivo de env, almaceno los datos que utilizare, para ser llamado en las diferentes partes…



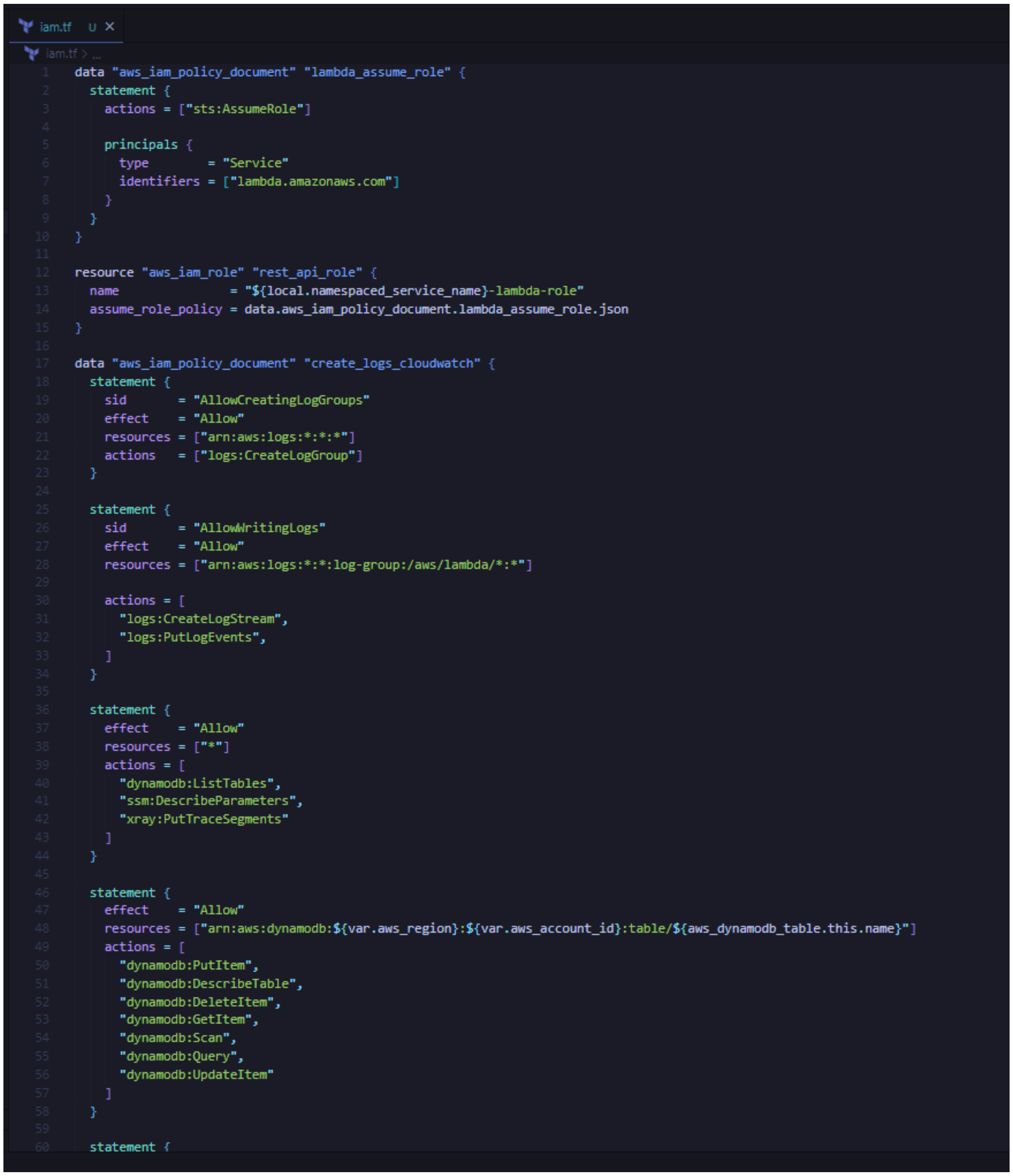
```
ariables.tf > variable "aws_profile"
variable "env" {
  type    = string
  default = "dev"
}

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

variable "aws_profile" {
  type    = string
  default = "default"
}

variable "aws_account_id" {
  type    = string
  default = "175452962618"
}

variable "service_name" {
  type    = string
  default = "todos"
}
```

Creo mi archivo lambda.tf el cual va a contener toda la información sobre las lambdas que voy a utilizar durante este proyecto.



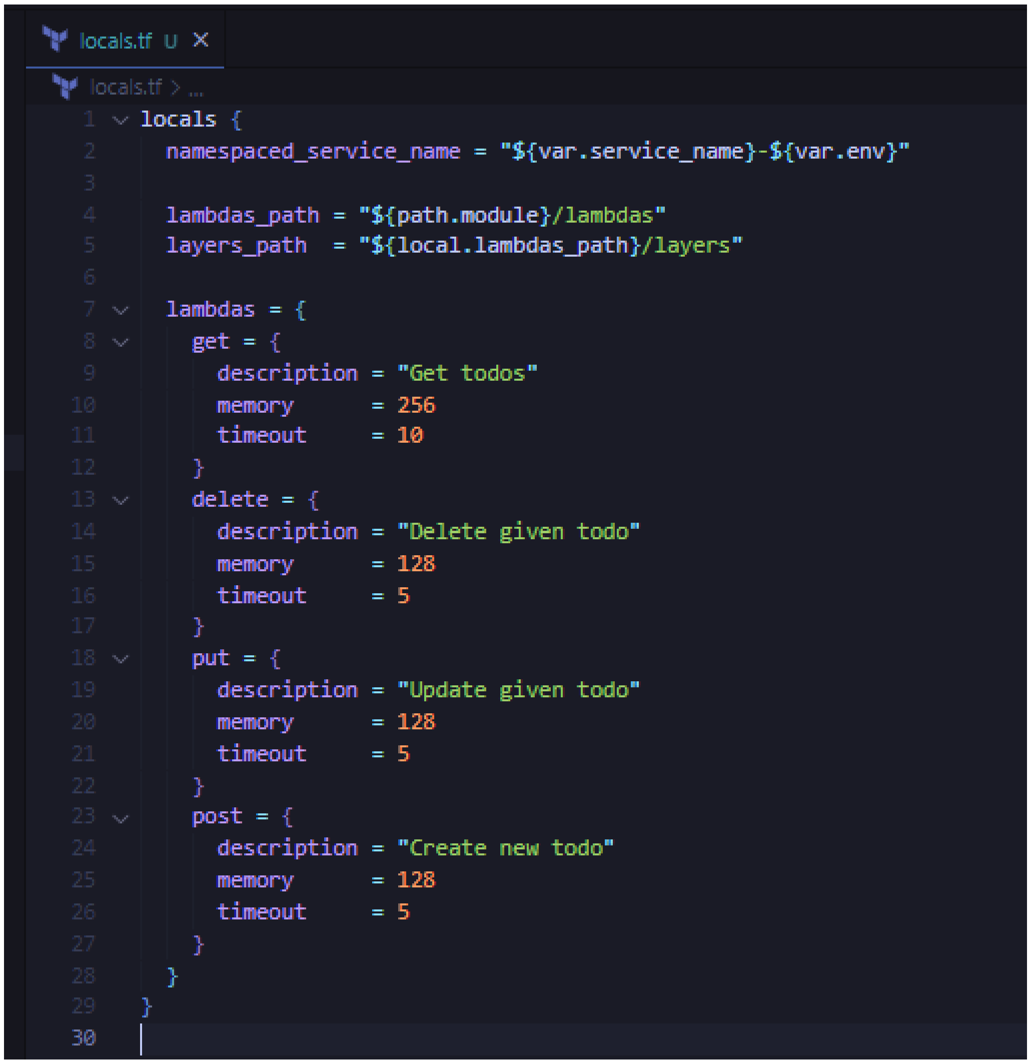
```
data "archive_file" "utils_layer" {
  output_path = "files/utils-layer.zip"
  type        = "zip"
  source_dir  = "${local.layers_path}/utils"
}

resource "aws_lambda_layer_version" "utils" {
  layer_name          = "utils-layer"
  description         = "Utils for response and event normalization"
  filename            = data.archive_file.utils_layer.output_path
  source_code_hash    = data.archive_file.utils_layer.output_base64sha256
  compatible_runtimes = ["nodejs14.x"]
}

data "archive_file" "todos" {
  for_each = local.lambdas

  output_path = "files/${each.key}-todo-artefact.zip"
  type        = "zip"
  source_file = "${local.lambdas_path}/todos/${each.key}.js"
}

resource "aws_lambda_function" "todos" {
  for_each = local.lambdas

  function_name = "dynamodb-${each.key}-item"
  handler       = "${each.key}.handler"
  description   = each.value["description"]
  role          = aws_iam_role.rest_api_role.arn
  runtime       = "nodejs14.x"

  filename         = data.archive_file.todos[each.key].output_path
  source_code_hash = data.archive_file.todos[each.key].output_base64sha256

  timeout     = each.value["timeout"]
  memory_size = each.value["memory"]

  layers = [aws_lambda_layer_version.utils.arn]

  tracing_config {
    mode = "Active"
  }

  environment {
    variables = {
      TABLE = aws_ssm_parameter.dynamodb_table.name
      DEBUG = var.env == "dev"
    }
  }
}

resource "aws_lambda_permission" "api" {
  for_each = local.lambdas

  action        = "lambda:InvokeFunction"
  function_name = aws_lambda_function.todos[each.key].arn
  principal     = "apigateway.amazonaws.com"
  source_arn    = "arn:aws:execute-api:${var.aws_region}:${var.aws_account_id}:*/*"
}
```

Creamos nuestro archivo el cual va a contener todos los permisos del proyecto:



```
data "aws_iam_policy_document" "lambda_assume_role" {
  statement {
    actions = ["sts:AssumeRole"]

    principals {
      type        = "Service"
      identifiers = ["lambda.amazonaws.com"]
    }
  }
}

resource "aws_iam_role" "rest_api_role" {
  name               = "${local.namespaced_service_name}-lambda-role"
  assume_role_policy = data.aws_iam_policy_document.lambda_assume_role.json
}

data "aws_iam_policy_document" "create_logs_cloudwatch" {
  statement {
    sid       = "AllowCreatingLogGroups"
    effect    = "Allow"
    resources = ["arn:aws:logs:*:*:*"]
    actions   = ["logs:CreateLogGroup"]
  }

  statement {
    sid       = "AllowWritingLogs"
    effect    = "Allow"
    resources = ["arn:aws:logs:*:*:log-group:/aws/lambda/*:*"]

    actions = [
      "logs:CreateLogStream",
      "logs:PutLogEvents",
    ]
  }

  statement {
    effect    = "Allow"
    resources = ["*"]
    actions = [
      "dynamodb:ListTables",
      "ssm:DescribeParameters",
      "xray:PutTraceSegments"
    ]
  }

  statement {
    effect    = "Allow"
    resources = ["arn:aws:dynamodb:${var.aws_region}:${var.aws_account_id}:table/${aws_dynamodb_table.this.name}"]
    actions = [
      "dynamodb:PutItem",
      "dynamodb:DescribeTable",
      "dynamodb:DeleteItem",
      "dynamodb:GetItem",
      "dynamodb:Scan",
      "dynamodb:Query",
      "dynamodb:UpdateItem"
    ]
  }

  statement {
```

Siguiente a la creación del archivo de permisos con IAM, definimos un archivo que contenga todos los locales que serán utilizados en el proyecto



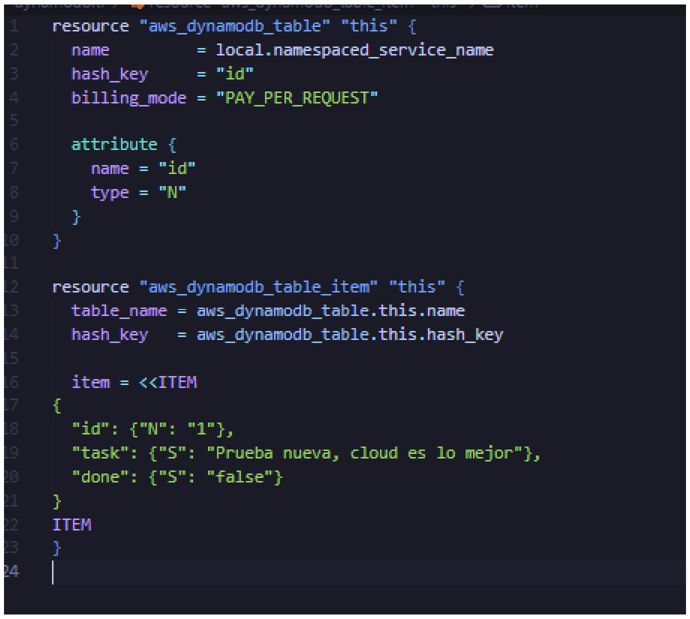
```
locals {
  namespaced_service_name = "${var.service_name}-${var.env}"

  lambdas_path = "${path.module}/lambdas"
  layers_path  = "${local.lambdas_path}/layers"

  lambdas = {
    get = {
      description = "Get todos"
      memory      = 256
      timeout     = 10
    }
    delete = {
      description = "Delete given todo"
      memory      = 128
      timeout     = 5
    }
    put = {
      description = "Update given todo"
      memory      = 128
      timeout     = 5
    }
    post = {
      description = "Create new todo"
      memory      = 128
      timeout     = 5
    }
  }
}
```

Los locals van a concadenar el ambiente con el services name, esto para diferenciar el cada tipo de solicitud.

En la parte de abajo declaro un verbo para cada una de la lambda que estaremos utilizando.

Después de las anteriores configuraciones, vamos a proceder con la creación de la base de datos, en este caso creamos un archivo llamado dynamo.tf

ATENCION: Es importante definir el archivo billing_mode para el costo de las solicitudes a la bd.

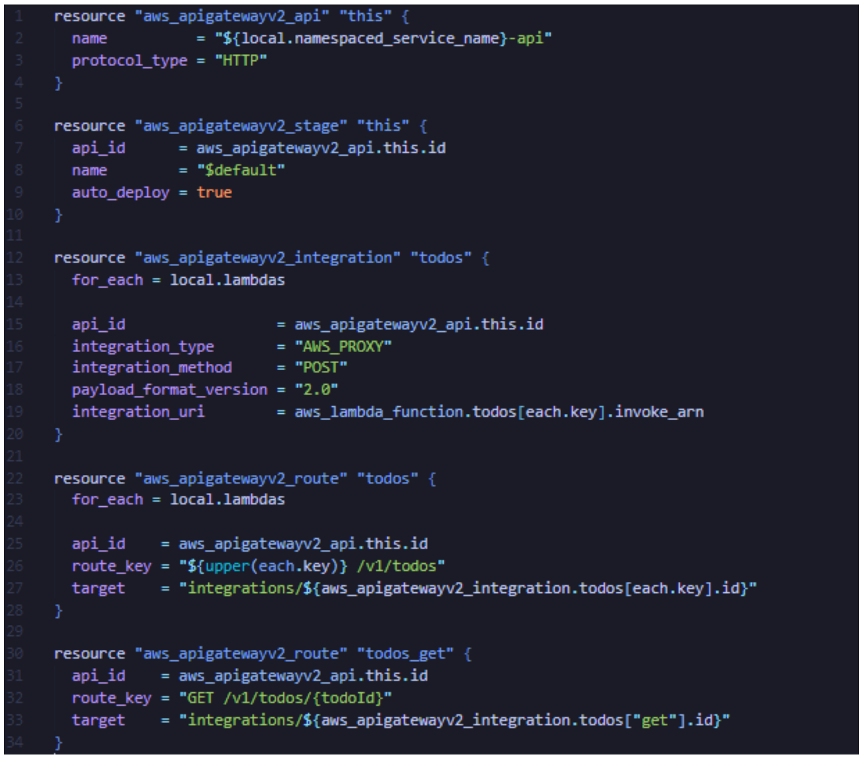


```
resource "aws_dynamodb_table" "this" {
  name         = local.namespaced_service_name
  hash_key     = "id"
  billing_mode = "PAY_PER_REQUEST"

  attribute {
    name = "id"
    type = "N"
  }
}

resource "aws_dynamodb_table_item" "this" {
  table_name = aws_dynamodb_table.this.name
  hash_key   = aws_dynamodb_table.this.hash_key

  item = <<ITEM
{
  "id": {"N": "1"},
  "task": {"S": "Prueba nueva, cloud es lo mejor"},
  "done": {"S": "false"}
}
ITEM
}
```

Una vez finalizada la configuración de la base de datos, es de suma importancia configurar el Amazon API Gateway que nos sirve para integrar los servicios.

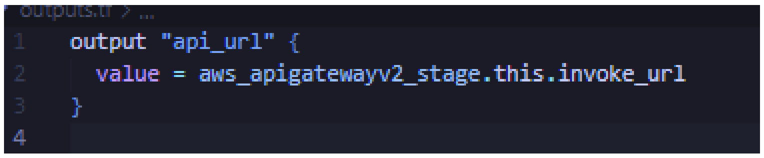
Para esto vamos a crear un archivo llamado api.tf, el cual va a contener esta configuración.

Es importante declarar el tipo de protocolo el cual va a realizar.



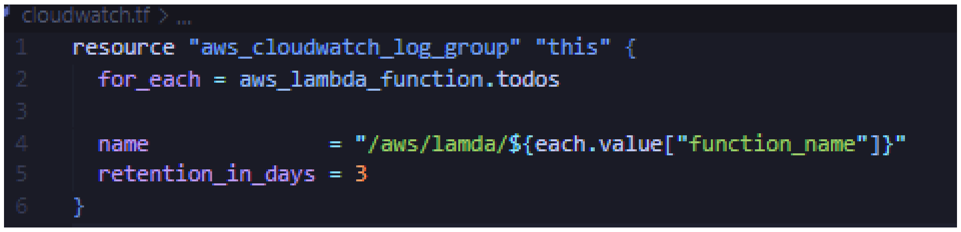
```
resource "aws_apigatewayv2_api" "this" {
  name          = "${local.namespaced_service_name}-api"
  protocol_type = "HTTP"
}

resource "aws_apigatewayv2_stage" "this" {
  api_id      = aws_apigatewayv2_api.this.id
  name        = "$default"
  auto_deploy = true
}

resource "aws_apigatewayv2_integration" "todos" {
  for_each = local.lambdas

  api_id                 = aws_apigatewayv2_api.this.id
  integration_type       = "AWS_PROXY"
  integration_method     = "POST"
  payload_format_version = "2.0"
  integration_uri        = aws_lambda_function.todos[each.key].invoke_arn
}

resource "aws_apigatewayv2_route" "todos" {
  for_each = local.lambdas

  api_id    = aws_apigatewayv2_api.this.id
  route_key = "${upper(each.key)} /v1/todos"
  target    = "integrations/${aws_apigatewayv2_integration.todos[each.key].id}"
}

resource "aws_apigatewayv2_route" "todos_get" {
  api_id    = aws_apigatewayv2_api.this.id
  route_key = "GET /v1/todos/{todoId}"
  target    = "integrations/${aws_apigatewayv2_integration.todos["get"].id}"
}
```

Crearemos un archivo llamado outpouts.tf que será el resultado que le dara la url.



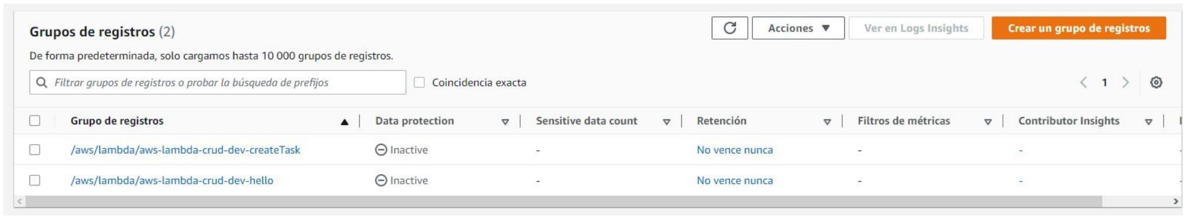
```
output "api_url" {
  value = aws_apigatewayv2_stage.this.invoke_url
}
```

Ahora vamos a configurar el servicio Cloud Watch, para esto vamos a crear un archivo con este mismo nombre agregando la extensión .tf



```
cloudwatch.tf > ...
resource "aws_cloudwatch_log_group" "this" {
  for_each = aws_lambda_function.todos

  name              = "/aws/lamda/${each.value["function_name"]}"
  retention_in_days = 3
}
```

Este servicio nos permite monitorear recursos y aplicaciones (LOGS)



Acciones
Ver en Logs Insights
Crear un grupo de registros
Filtrar grupos de registros o probar la búsqueda de prefijos
Coincidencia exacta
1

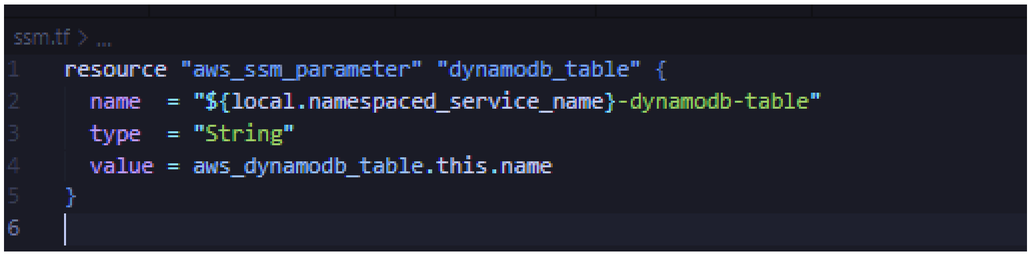
Grupos de registros (2)

De forma predeterminada, solo cargamos hasta 10 000 grupos de registros.

Grupo de registros	Data protection	Sensitive data count	Retención	Filtros de métricas	Contributor Insights
/aws/lambda/aws-lambda-crud-dev-createTask	Inactive	-	No vence nunca	-	-
/aws/lambda/aws-lambda-crud-dev-hello	Inactive	-	No vence nunca	-	-

Aquí podremos monitorear todas las solicitudes que hagamos a los endpoints.

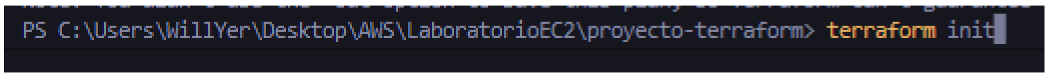
Ahora crearemos un archivo llamado ssm.tf para la configuración del Parameter Store.



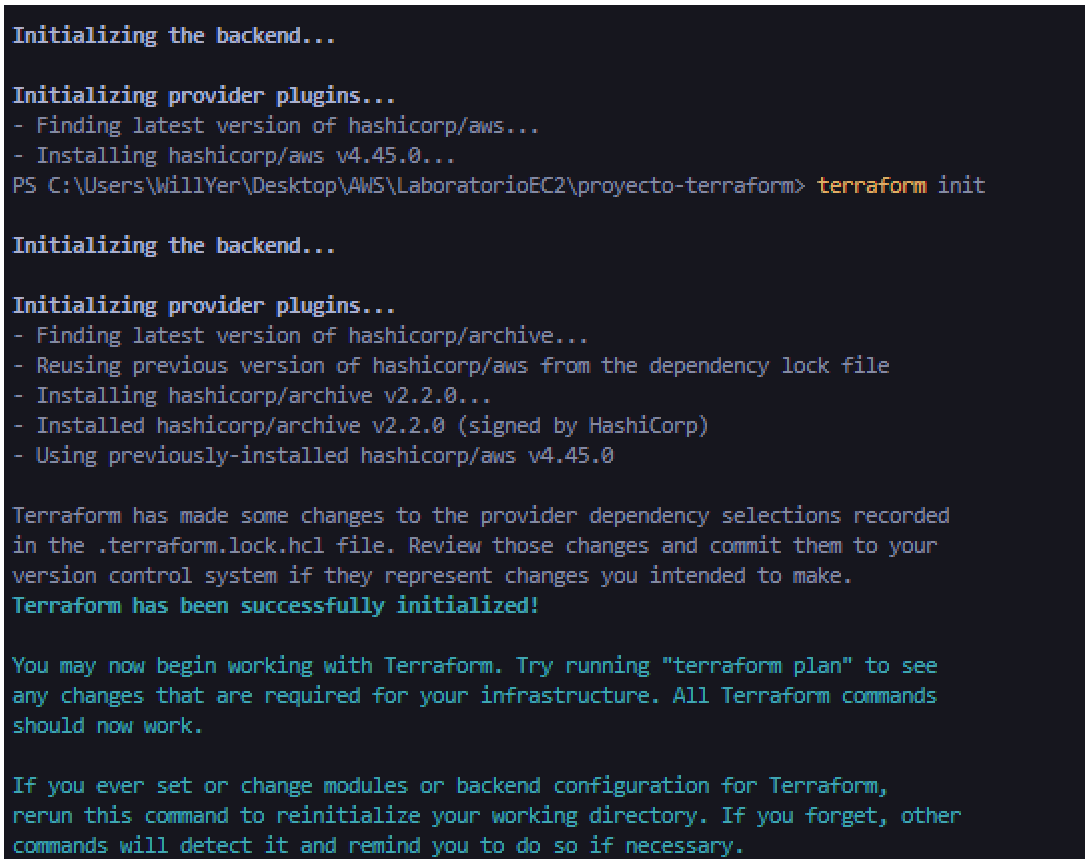
```
ssm.tf > ...
resource "aws_ssm_parameter" "dynamodb_table" {
  name  = "${local.namespaced_service_name}-dynamodb-table"
  type  = "String"
  value = aws_dynamodb_table.this.name
}
```

Al finalizar esta configuración, podemos dar por cerrada la parte de la configuración de los servicios.

Iniciamos este proyecto inicializando nuestro terraform…

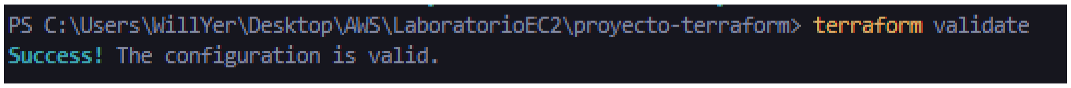


```
PS C:\Users\WillYer\Desktop\AWS\LaboratorioEC2\proyecto-terraform> terraform init
```

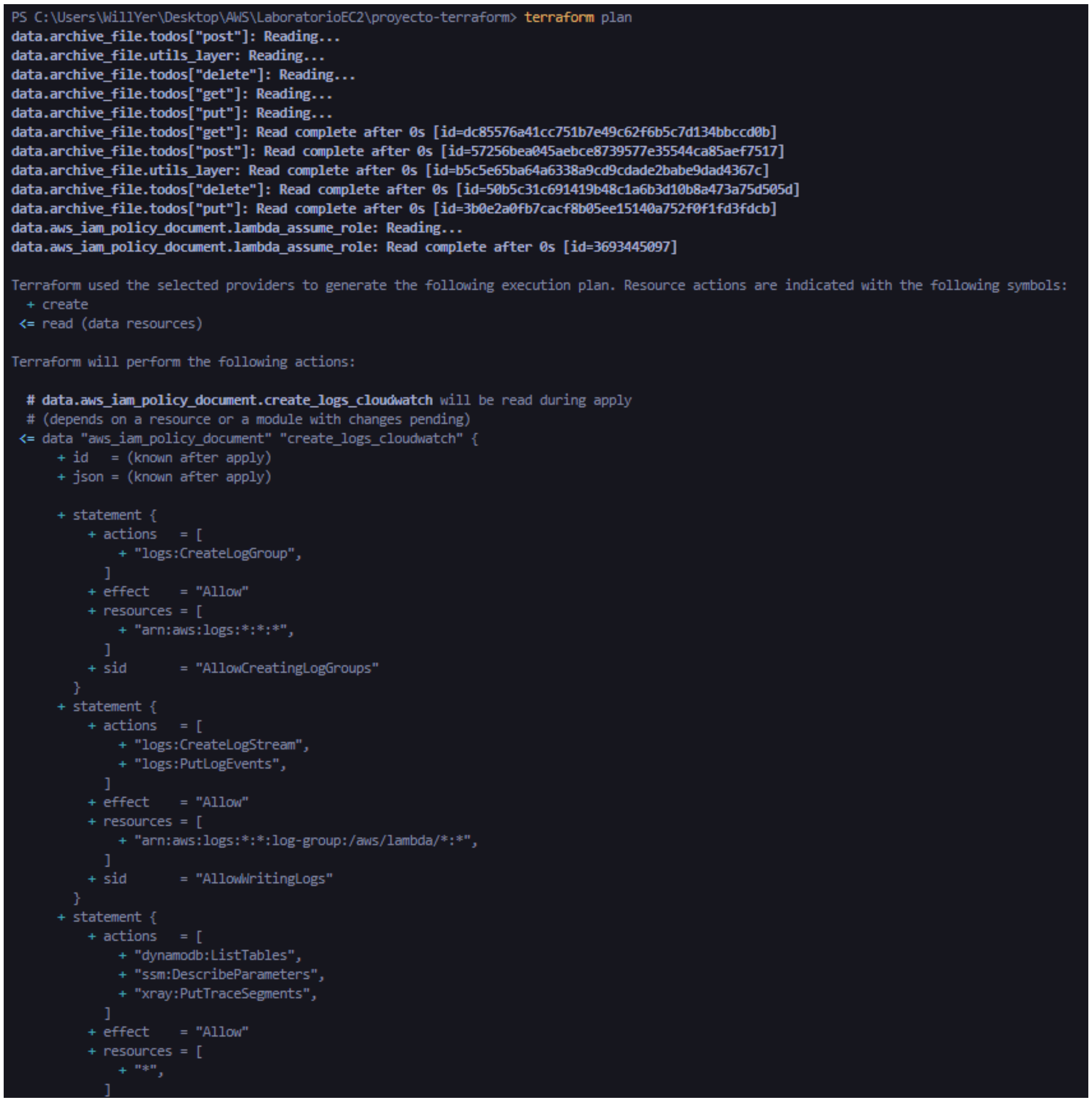


```
Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v4.45.0...
PS C:\Users\WillYer\Desktop\AWS\LaboratorioEC2\proyecto-terraform> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/archive...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Installing hashicorp/archive v2.2.0...
- Installed hashicorp/archive v2.2.0 (signed by HashiCorp)
- Using previously-installed hashicorp/aws v4.45.0

Terraform has made some changes to the provider dependency selections recorded
in the .terraform.lock.hcl file. Review those changes and commit them to your
version control system if they represent changes you intended to make.
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.
```

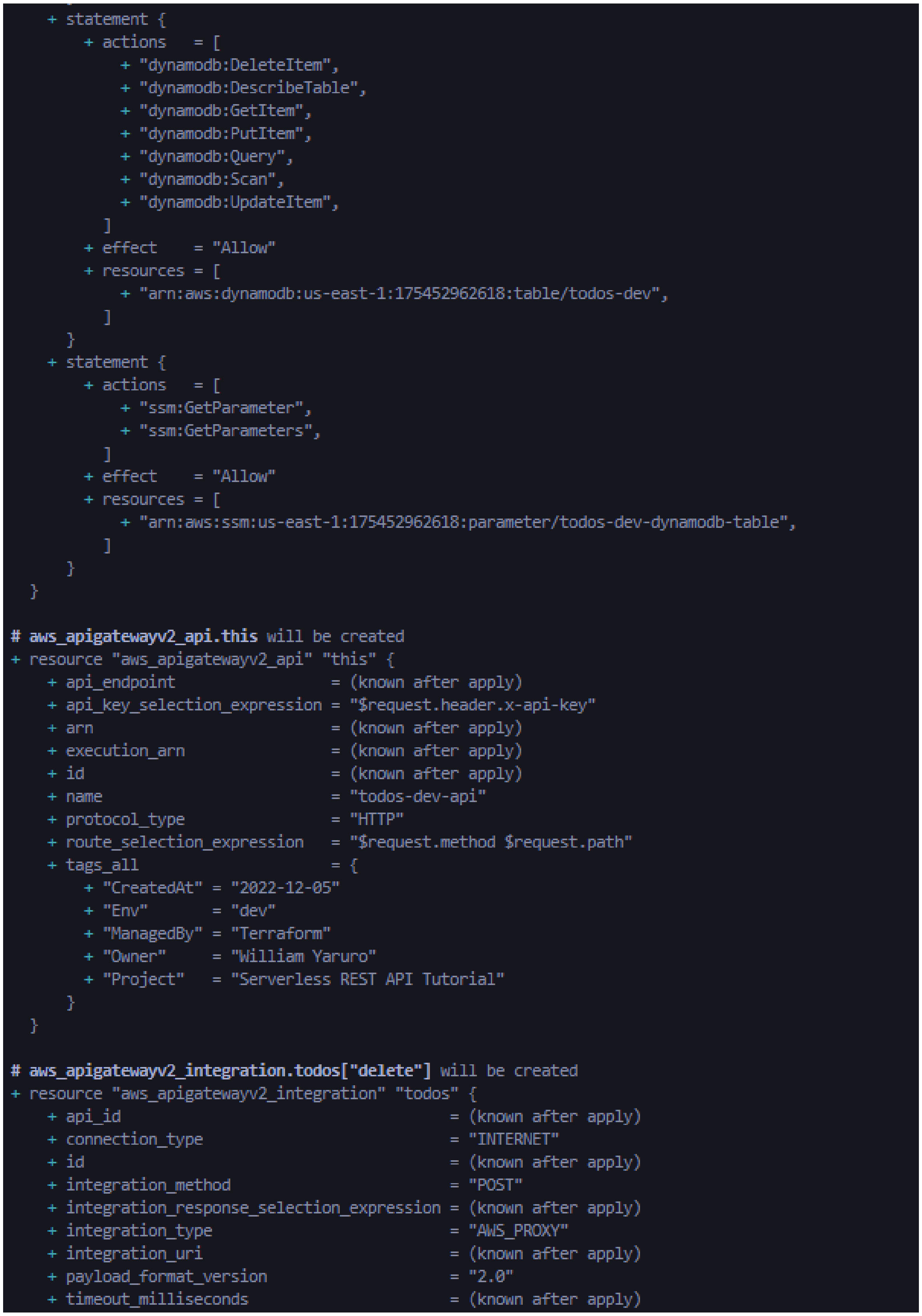
Validamos que las configuraciones estén correctamente.



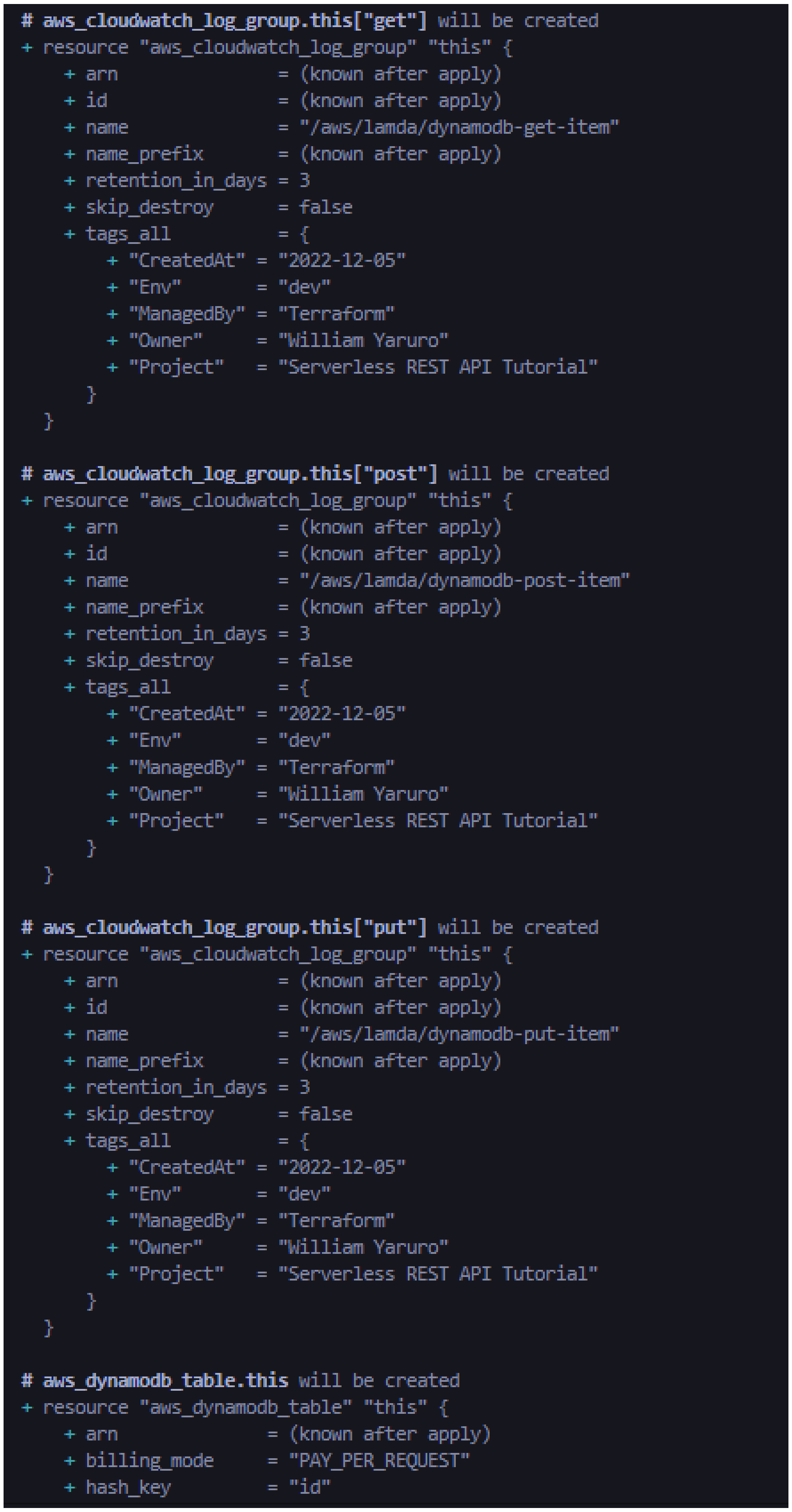
```
PS C:\Users\WillYer\Desktop\AWS\LaboratorioEC2\proyecto-terraform> terraform validate
Success! The configuration is valid.
```

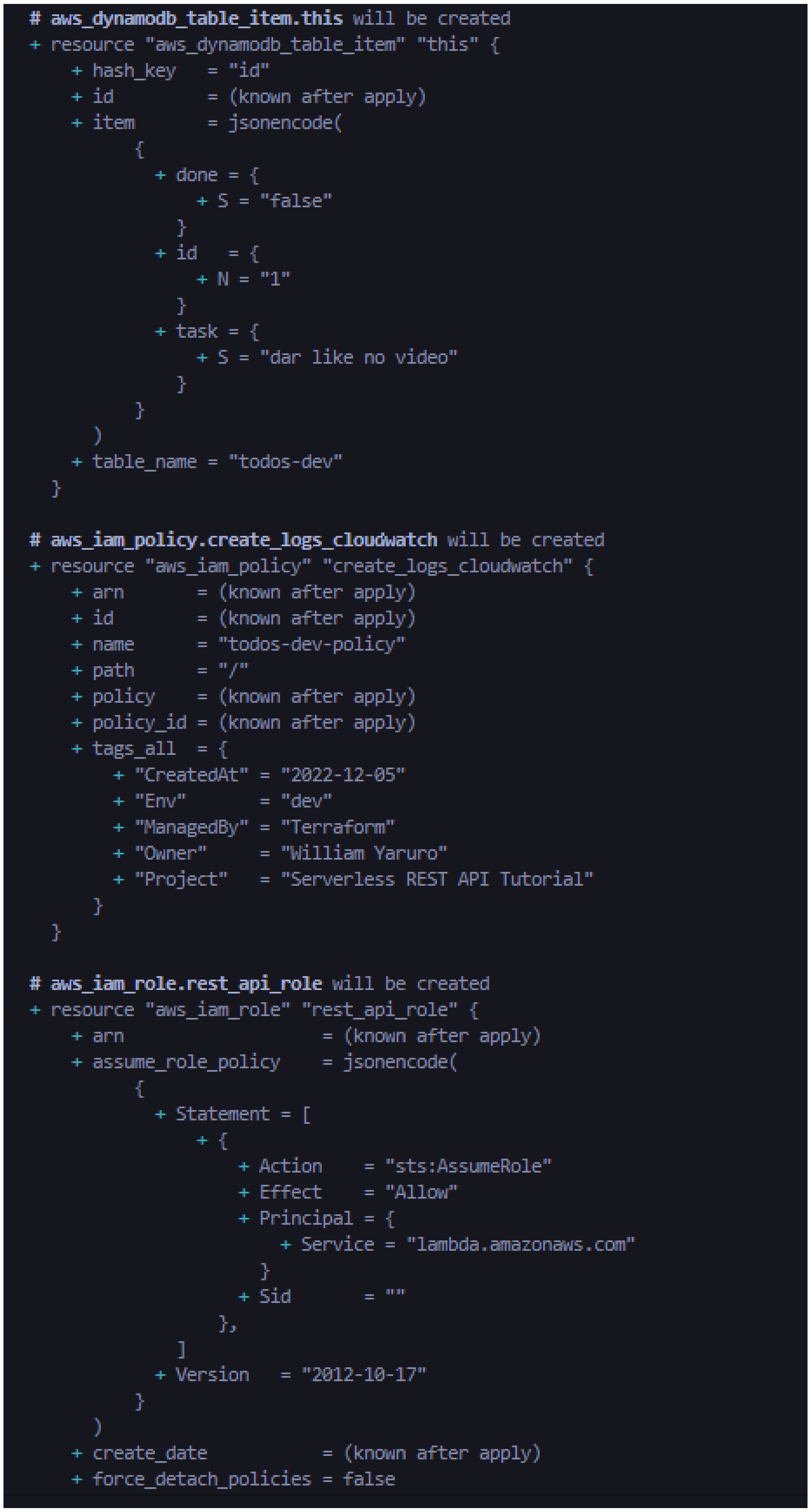


```
PS C:\Users\WillYer\Desktop\AWS\LaboratorioEC2\proyecto-terraform> terraform plan
data.archive_file.todos["post"]: Reading...
data.archive_file.utils_layer: Reading...
data.archive_file.todos["delete"]: Reading...
data.archive_file.todos["get"]: Reading...
data.archive_file.todos["put"]: Reading...
data.archive_file.todos["get"]: Read complete after 0s [id=dc85576a41cc751b7e49c62f6b5c7d134bbccd0b]
data.archive_file.todos["post"]: Read complete after 0s [id=57256bea045aebce8739577e35544ca85aef7517]
data.archive_file.utils_layer: Read complete after 0s [id=b5c5e65ba64a6338a9cd9cdade2babe9dad4367c]
data.archive_file.todos["delete"]: Read complete after 0s [id=50b5c31c691419b48c1a6b3d10b8a473a75d505d]
data.archive_file.todos["put"]: Read complete after 0s [id=3b0e2a0fb7cacf8b05ee15140a752f0f1fd3fdcb]
data.aws_iam_policy_document.lambda_assume_role: Reading...
data.aws_iam_policy_document.lambda_assume_role: Read complete after 0s [id=3693445097]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create
 <= read (data resources)

Terraform will perform the following actions:

  # data.aws_iam_policy_document.create_logs_cloudwatch will be read during apply
  # (depends on a resource or a module with changes pending)
 <= data "aws_iam_policy_document" "create_logs_cloudwatch" {
      + id   = (known after apply)
      + json = (known after apply)

      + statement {
          + actions   = [
              + "logs:CreateLogGroup",
            ]
          + effect    = "Allow"
          + resources = [
              + "arn:aws:logs:*:*:*",
            ]
          + sid       = "AllowCreatingLogGroups"
        }
      + statement {
          + actions   = [
              + "logs:CreateLogStream",
              + "logs:PutLogEvents",
            ]
          + effect    = "Allow"
          + resources = [
              + "arn:aws:logs:*:*:log-group:/aws/lambda/*:*",
            ]
          + sid       = "AllowWritingLogs"
        }
      + statement {
          + actions   = [
              + "dynamodb:ListTables",
              + "ssm:DescribeParameters",
              + "xray:PutTraceSegments",
            ]
          + effect    = "Allow"
          + resources = [
              + "*",
            ]
```

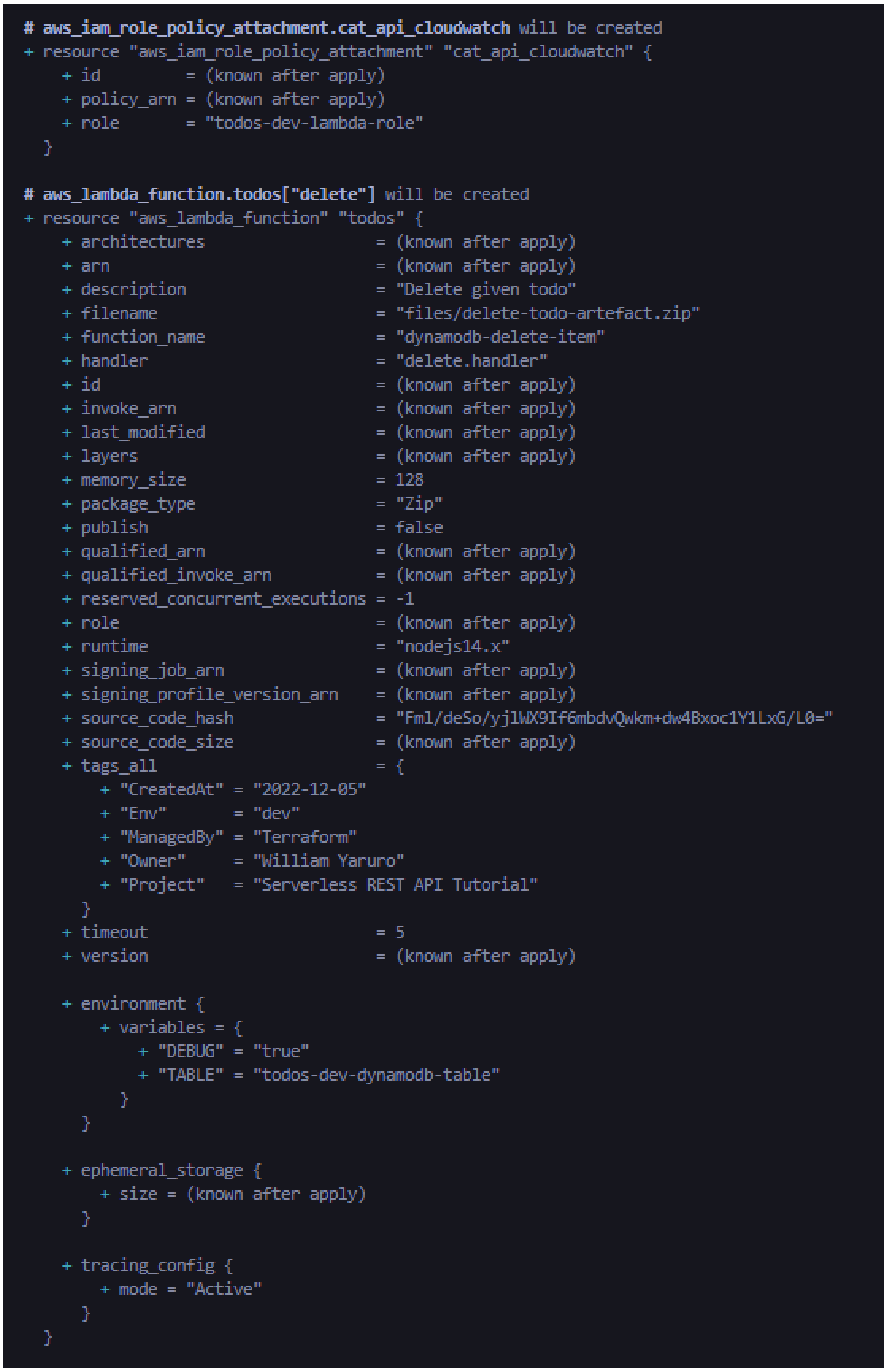


```
    + statement {
        + actions   = [
            + "dynamodb:DeleteItem",
            + "dynamodb:DescribeTable",
            + "dynamodb:GetItem",
            + "dynamodb:PutItem",
            + "dynamodb:Query",
            + "dynamodb:Scan",
            + "dynamodb:UpdateItem",
          ]
        + effect    = "Allow"
        + resources = [
            + "arn:aws:dynamodb:us-east-1:175452962618:table/todos-dev",
          ]
      }
    + statement {
        + actions   = [
            + "ssm:GetParameter",
            + "ssm:GetParameters",
          ]
        + effect    = "Allow"
        + resources = [
            + "arn:aws:ssm:us-east-1:175452962618:parameter/todos-dev-dynamodb-table",
          ]
      }
  }

# aws_apigatewayv2_api.this will be created
+ resource "aws_apigatewayv2_api" "this" {
    + api_endpoint                 = (known after apply)
    + api_key_selection_expression = "$request.header.x-api-key"
    + arn                          = (known after apply)
    + execution_arn                = (known after apply)
    + id                           = (known after apply)
    + name                         = "todos-dev-api"
    + protocol_type                = "HTTP"
    + route_selection_expression   = "$request.method $request.path"
    + tags_all                     = {
        + "CreatedAt" = "2022-12-05"
        + "Env"       = "dev"
        + "ManagedBy" = "Terraform"
        + "Owner"     = "William Yaruro"
        + "Project"   = "Serverless REST API Tutorial"
      }
  }

# aws_apigatewayv2_integration.todos["delete"] will be created
+ resource "aws_apigatewayv2_integration" "todos" {
    + api_id                                    = (known after apply)
    + connection_type                           = "INTERNET"
    + id                                        = (known after apply)
    + integration_method                        = "POST"
    + integration_response_selection_expression = (known after apply)
    + integration_type                          = "AWS_PROXY"
    + integration_uri                           = (known after apply)
    + payload_format_version                    = "2.0"
    + timeout_milliseconds                      = (known after apply)
```

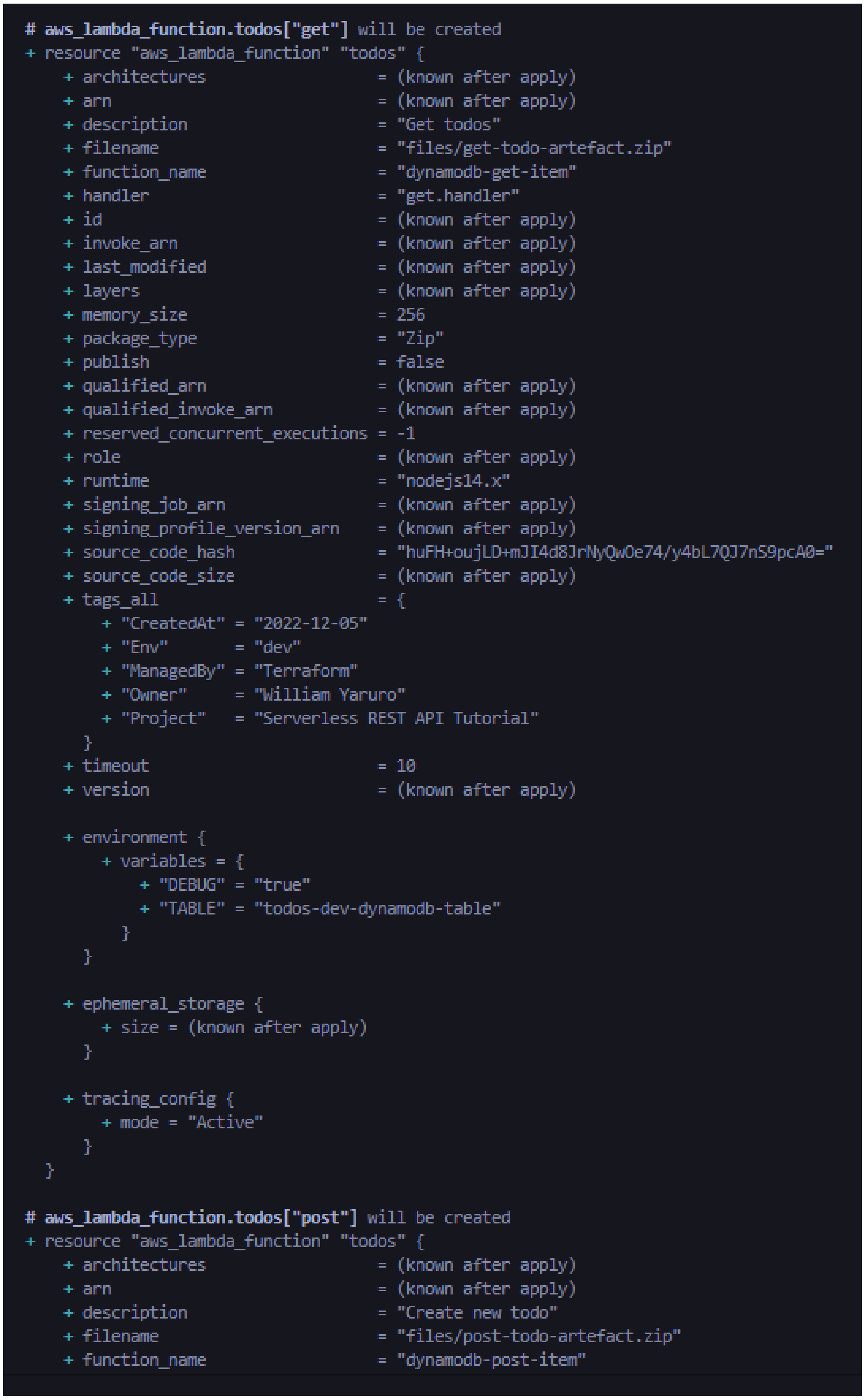

```
# aws_cloudwatch_log_group.this["get"] will be created
+ resource "aws_cloudwatch_log_group" "this" {
    + arn               = (known after apply)
    + id                = (known after apply)
    + name              = "/aws/lamda/dynamodb-get-item"
    + name_prefix       = (known after apply)
    + retention_in_days = 3
    + skip_destroy      = false
    + tags_all          = {
        + "CreatedAt" = "2022-12-05"
        + "Env"       = "dev"
        + "ManagedBy" = "Terraform"
        + "Owner"     = "William Yaruro"
        + "Project"   = "Serverless REST API Tutorial"
      }
  }

# aws_cloudwatch_log_group.this["post"] will be created
+ resource "aws_cloudwatch_log_group" "this" {
    + arn               = (known after apply)
    + id                = (known after apply)
    + name              = "/aws/lamda/dynamodb-post-item"
    + name_prefix       = (known after apply)
    + retention_in_days = 3
    + skip_destroy      = false
    + tags_all          = {
        + "CreatedAt" = "2022-12-05"
        + "Env"       = "dev"
        + "ManagedBy" = "Terraform"
        + "Owner"     = "William Yaruro"
        + "Project"   = "Serverless REST API Tutorial"
      }
  }

# aws_cloudwatch_log_group.this["put"] will be created
+ resource "aws_cloudwatch_log_group" "this" {
    + arn               = (known after apply)
    + id                = (known after apply)
    + name              = "/aws/lamda/dynamodb-put-item"
    + name_prefix       = (known after apply)
    + retention_in_days = 3
    + skip_destroy      = false
    + tags_all          = {
        + "CreatedAt" = "2022-12-05"
        + "Env"       = "dev"
        + "ManagedBy" = "Terraform"
        + "Owner"     = "William Yaruro"
        + "Project"   = "Serverless REST API Tutorial"
      }
  }

# aws_dynamodb_table.this will be created
+ resource "aws_dynamodb_table" "this" {
    + arn              = (known after apply)
    + billing_mode     = "PAY_PER_REQUEST"
    + hash_key         = "id"
```



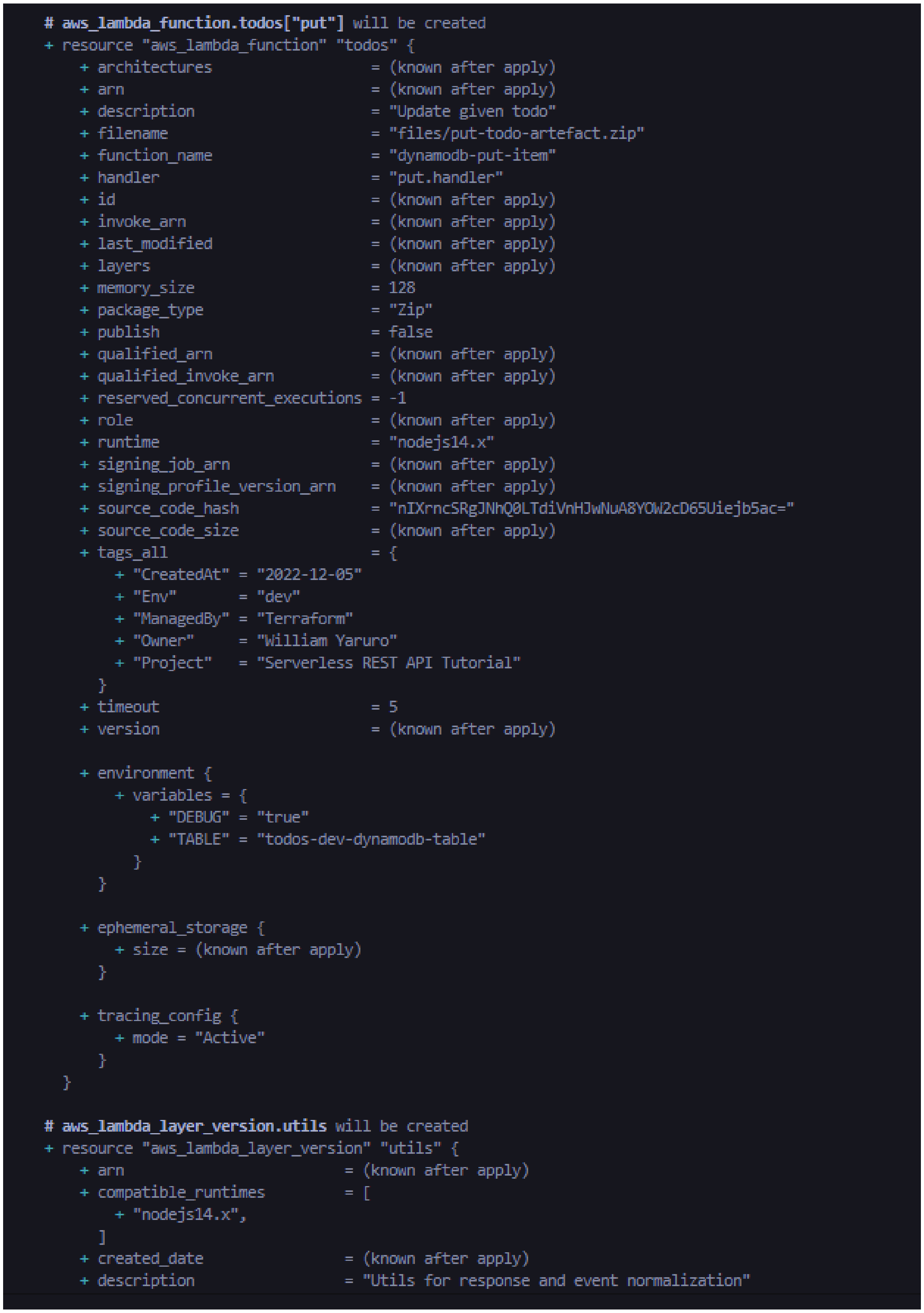
```
# aws_dynamodb_table_item.this will be created
+ resource "aws_dynamodb_table_item" "this" {
    + hash_key   = "id"
    + id         = (known after apply)
    + item       = jsonencode(
          {
            + done = {
                + S = "false"
              }
            + id   = {
                + N = "1"
              }
            + task = {
                + S = "dar like no video"
              }
          }
      )
    + table_name = "todos-dev"
  }

# aws_iam_policy.create_logs_cloudwatch will be created
+ resource "aws_iam_policy" "create_logs_cloudwatch" {
    + arn       = (known after apply)
    + id        = (known after apply)
    + name      = "todos-dev-policy"
    + path      = "/"
    + policy    = (known after apply)
    + policy_id = (known after apply)
    + tags_all  = {
        + "CreatedAt" = "2022-12-05"
        + "Env"       = "dev"
        + "ManagedBy" = "Terraform"
        + "Owner"     = "William Yaruro"
        + "Project"   = "Serverless REST API Tutorial"
      }
  }

# aws_iam_role.rest_api_role will be created
+ resource "aws_iam_role" "rest_api_role" {
    + arn                   = (known after apply)
    + assume_role_policy    = jsonencode(
          {
            + Statement = [
                + {
                    + Action    = "sts:AssumeRole"
                    + Effect    = "Allow"
                    + Principal = {
                        + Service = "lambda.amazonaws.com"
                      }
                    + Sid       = ""
                  },
              ]
            + Version   = "2012-10-17"
          }
      )
    + create_date           = (known after apply)
    + force_detach_policies = false
```



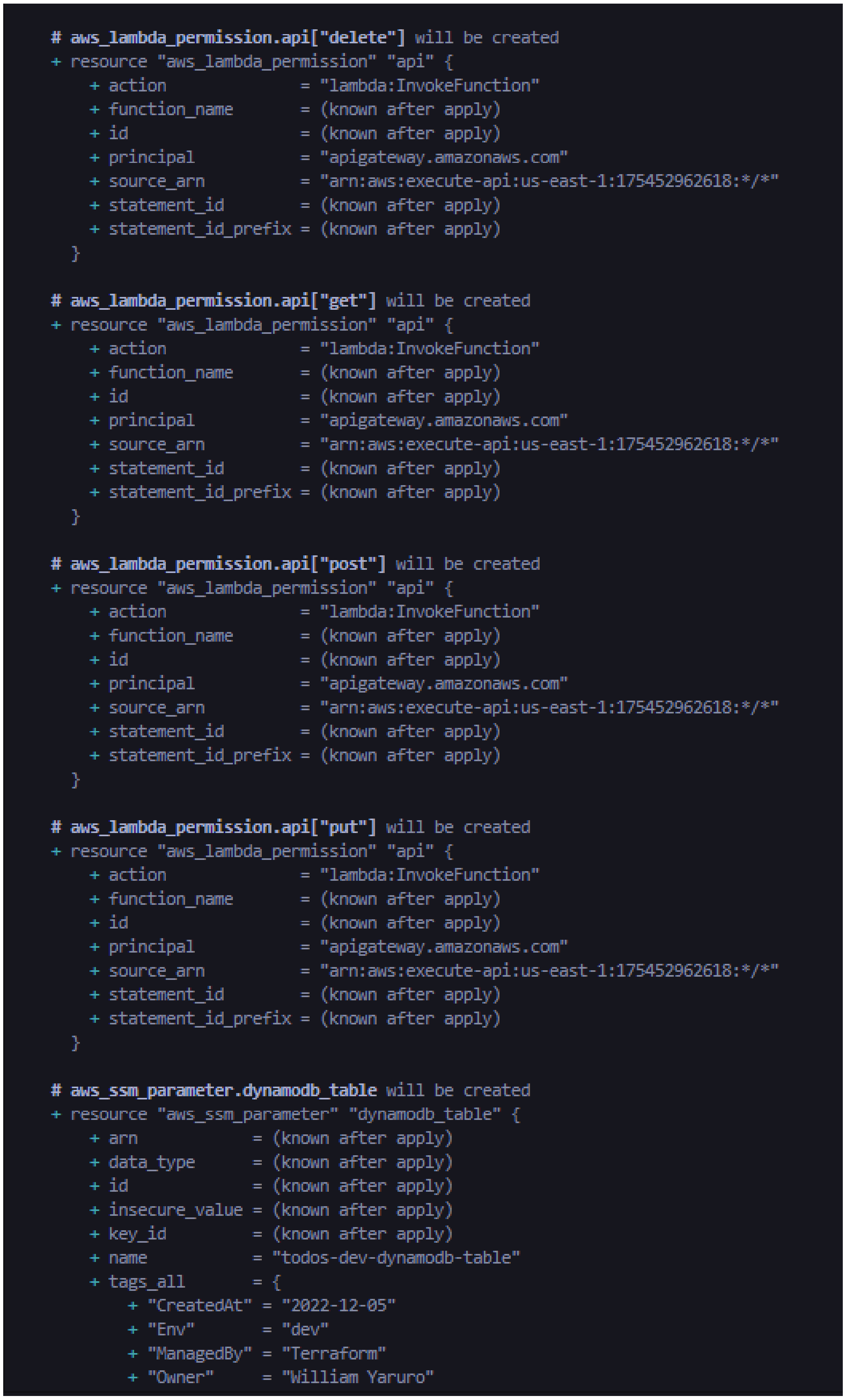
```
# aws_iam_role_policy_attachment.cat_api_cloudwatch will be created
+ resource "aws_iam_role_policy_attachment" "cat_api_cloudwatch" {
    + id         = (known after apply)
    + policy_arn = (known after apply)
    + role       = "todos-dev-lambda-role"
  }

# aws_lambda_function.todos["delete"] will be created
+ resource "aws_lambda_function" "todos" {
    + architectures                  = (known after apply)
    + arn                            = (known after apply)
    + description                    = "Delete given todo"
    + filename                       = "files/delete-todo-artefact.zip"
    + function_name                  = "dynamodb-delete-item"
    + handler                        = "delete.handler"
    + id                             = (known after apply)
    + invoke_arn                     = (known after apply)
    + last_modified                  = (known after apply)
    + layers                         = (known after apply)
    + memory_size                    = 128
    + package_type                   = "Zip"
    + publish                        = false
    + qualified_arn                  = (known after apply)
    + qualified_invoke_arn           = (known after apply)
    + reserved_concurrent_executions = -1
    + role                           = (known after apply)
    + runtime                        = "nodejs14.x"
    + signing_job_arn                = (known after apply)
    + signing_profile_version_arn    = (known after apply)
    + source_code_hash               = "Fml/deSo/yjlWX9If6mbdvQwkm+dw4Bxoc1Y1LxG/L0="
    + source_code_size               = (known after apply)
    + tags_all                       = {
        + "CreatedAt" = "2022-12-05"
        + "Env"       = "dev"
        + "ManagedBy" = "Terraform"
        + "Owner"     = "William Yaruro"
        + "Project"   = "Serverless REST API Tutorial"
      }
    + timeout                        = 5
    + version                        = (known after apply)

    + environment {
        + variables = {
            + "DEBUG" = "true"
            + "TABLE" = "todos-dev-dynamodb-table"
          }
      }

    + ephemeral_storage {
        + size = (known after apply)
      }

    + tracing_config {
        + mode = "Active"
      }
  }
```



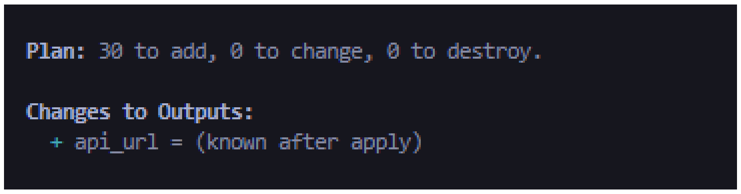
```
# aws_lambda_function.todos["get"] will be created
+ resource "aws_lambda_function" "todos" {
    + architectures                  = (known after apply)
    + arn                            = (known after apply)
    + description                    = "Get todos"
    + filename                       = "files/get-todo-artefact.zip"
    + function_name                  = "dynamodb-get-item"
    + handler                        = "get.handler"
    + id                             = (known after apply)
    + invoke_arn                     = (known after apply)
    + last_modified                  = (known after apply)
    + layers                         = (known after apply)
    + memory_size                    = 256
    + package_type                   = "Zip"
    + publish                        = false
    + qualified_arn                  = (known after apply)
    + qualified_invoke_arn           = (known after apply)
    + reserved_concurrent_executions = -1
    + role                           = (known after apply)
    + runtime                        = "nodejs14.x"
    + signing_job_arn                = (known after apply)
    + signing_profile_version_arn    = (known after apply)
    + source_code_hash               = "huFH+oujLD+mJI4d8JrNyQwOe74/y4bL7QJ7nS9pcA0="
    + source_code_size               = (known after apply)
    + tags_all                       = {
        + "CreatedAt" = "2022-12-05"
        + "Env"       = "dev"
        + "ManagedBy" = "Terraform"
        + "Owner"     = "William Yaruro"
        + "Project"   = "Serverless REST API Tutorial"
      }
    + timeout                        = 10
    + version                        = (known after apply)

    + environment {
        + variables = {
            + "DEBUG" = "true"
            + "TABLE" = "todos-dev-dynamodb-table"
          }
      }

    + ephemeral_storage {
        + size = (known after apply)
      }

    + tracing_config {
        + mode = "Active"
      }
  }

# aws_lambda_function.todos["post"] will be created
+ resource "aws_lambda_function" "todos" {
    + architectures                  = (known after apply)
    + arn                            = (known after apply)
    + description                    = "Create new todo"
    + filename                       = "files/post-todo-artefact.zip"
    + function_name                  = "dynamodb-post-item"
```



```
  # aws_lambda_function.todos["put"] will be created
  + resource "aws_lambda_function" "todos" {
      + architectures                  = (known after apply)
      + arn                            = (known after apply)
      + description                    = "Update given todo"
      + filename                       = "files/put-todo-artefact.zip"
      + function_name                  = "dynamodb-put-item"
      + handler                        = "put.handler"
      + id                             = (known after apply)
      + invoke_arn                     = (known after apply)
      + last_modified                  = (known after apply)
      + layers                         = (known after apply)
      + memory_size                    = 128
      + package_type                   = "Zip"
      + publish                        = false
      + qualified_arn                  = (known after apply)
      + qualified_invoke_arn           = (known after apply)
      + reserved_concurrent_executions = -1
      + role                           = (known after apply)
      + runtime                        = "nodejs14.x"
      + signing_job_arn                = (known after apply)
      + signing_profile_version_arn    = (known after apply)
      + source_code_hash               = "nIXrncSRgJNhQ0LTdiVnHJwNuA8YOW2cD65Uiejb5ac="
      + source_code_size               = (known after apply)
      + tags_all                       = {
          + "CreatedAt" = "2022-12-05"
          + "Env"       = "dev"
          + "ManagedBy" = "Terraform"
          + "Owner"     = "William Yaruro"
          + "Project"   = "Serverless REST API Tutorial"
        }
      + timeout                        = 5
      + version                        = (known after apply)

      + environment {
          + variables = {
              + "DEBUG" = "true"
              + "TABLE" = "todos-dev-dynamodb-table"
            }
        }

      + ephemeral_storage {
          + size = (known after apply)
        }

      + tracing_config {
          + mode = "Active"
        }
    }

  # aws_lambda_layer_version.utils will be created
  + resource "aws_lambda_layer_version" "utils" {
      + arn                         = (known after apply)
      + compatible_runtimes         = [
          + "nodejs14.x",
        ]
      + created_date                = (known after apply)
      + description                 = "Utils for response and event normalization"
```

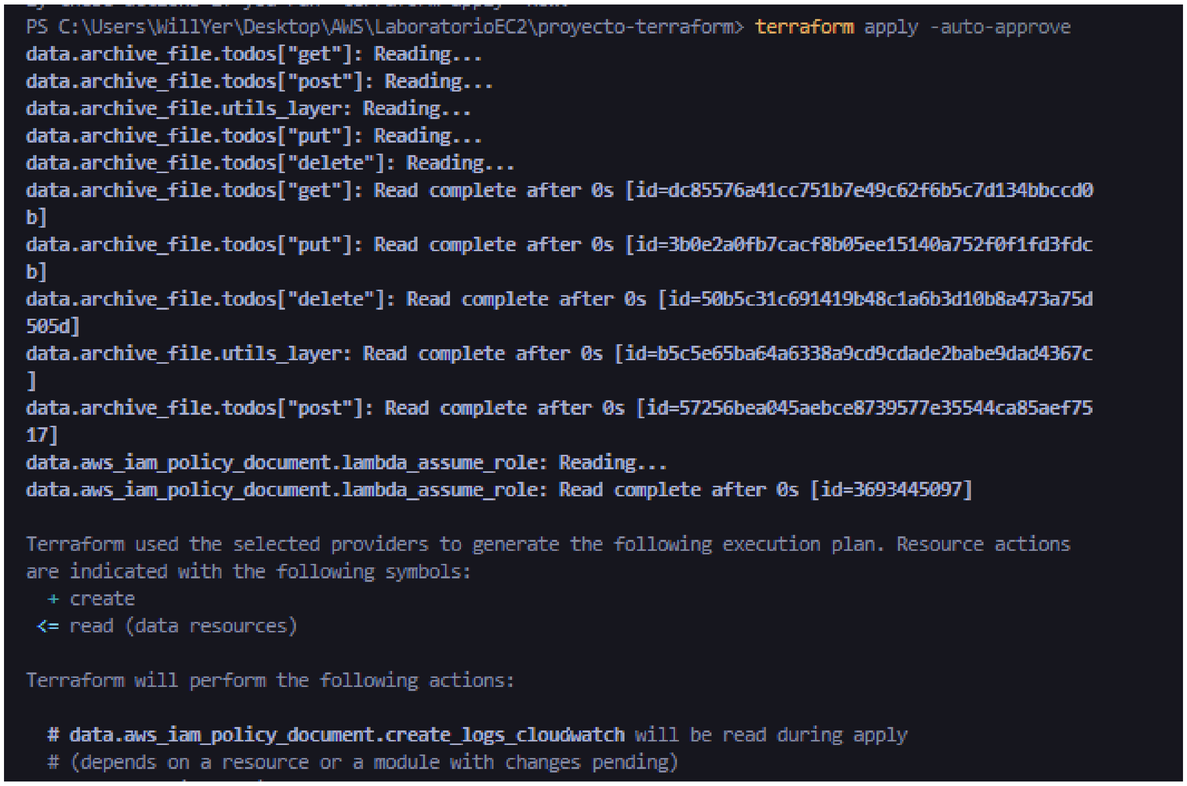


```
# aws_lambda_permission.api["delete"] will be created
+ resource "aws_lambda_permission" "api" {
    + action              = "lambda:InvokeFunction"
    + function_name       = (known after apply)
    + id                  = (known after apply)
    + principal           = "apigateway.amazonaws.com"
    + source_arn          = "arn:aws:execute-api:us-east-1:175452962618:*/*"
    + statement_id        = (known after apply)
    + statement_id_prefix = (known after apply)
  }

# aws_lambda_permission.api["get"] will be created
+ resource "aws_lambda_permission" "api" {
    + action              = "lambda:InvokeFunction"
    + function_name       = (known after apply)
    + id                  = (known after apply)
    + principal           = "apigateway.amazonaws.com"
    + source_arn          = "arn:aws:execute-api:us-east-1:175452962618:*/*"
    + statement_id        = (known after apply)
    + statement_id_prefix = (known after apply)
  }

# aws_lambda_permission.api["post"] will be created
+ resource "aws_lambda_permission" "api" {
    + action              = "lambda:InvokeFunction"
    + function_name       = (known after apply)
    + id                  = (known after apply)
    + principal           = "apigateway.amazonaws.com"
    + source_arn          = "arn:aws:execute-api:us-east-1:175452962618:*/*"
    + statement_id        = (known after apply)
    + statement_id_prefix = (known after apply)
  }

# aws_lambda_permission.api["put"] will be created
+ resource "aws_lambda_permission" "api" {
    + action              = "lambda:InvokeFunction"
    + function_name       = (known after apply)
    + id                  = (known after apply)
    + principal           = "apigateway.amazonaws.com"
    + source_arn          = "arn:aws:execute-api:us-east-1:175452962618:*/*"
    + statement_id        = (known after apply)
    + statement_id_prefix = (known after apply)
  }

# aws_ssm_parameter.dynamodb_table will be created
+ resource "aws_ssm_parameter" "dynamodb_table" {
    + arn            = (known after apply)
    + data_type      = (known after apply)
    + id             = (known after apply)
    + insecure_value = (known after apply)
    + key_id         = (known after apply)
    + name           = "todos-dev-dynamodb-table"
    + tags_all       = {
        + "CreatedAt" = "2022-12-05"
        + "Env"       = "dev"
        + "ManagedBy" = "Terraform"
        + "Owner"     = "William Yaruro"
```

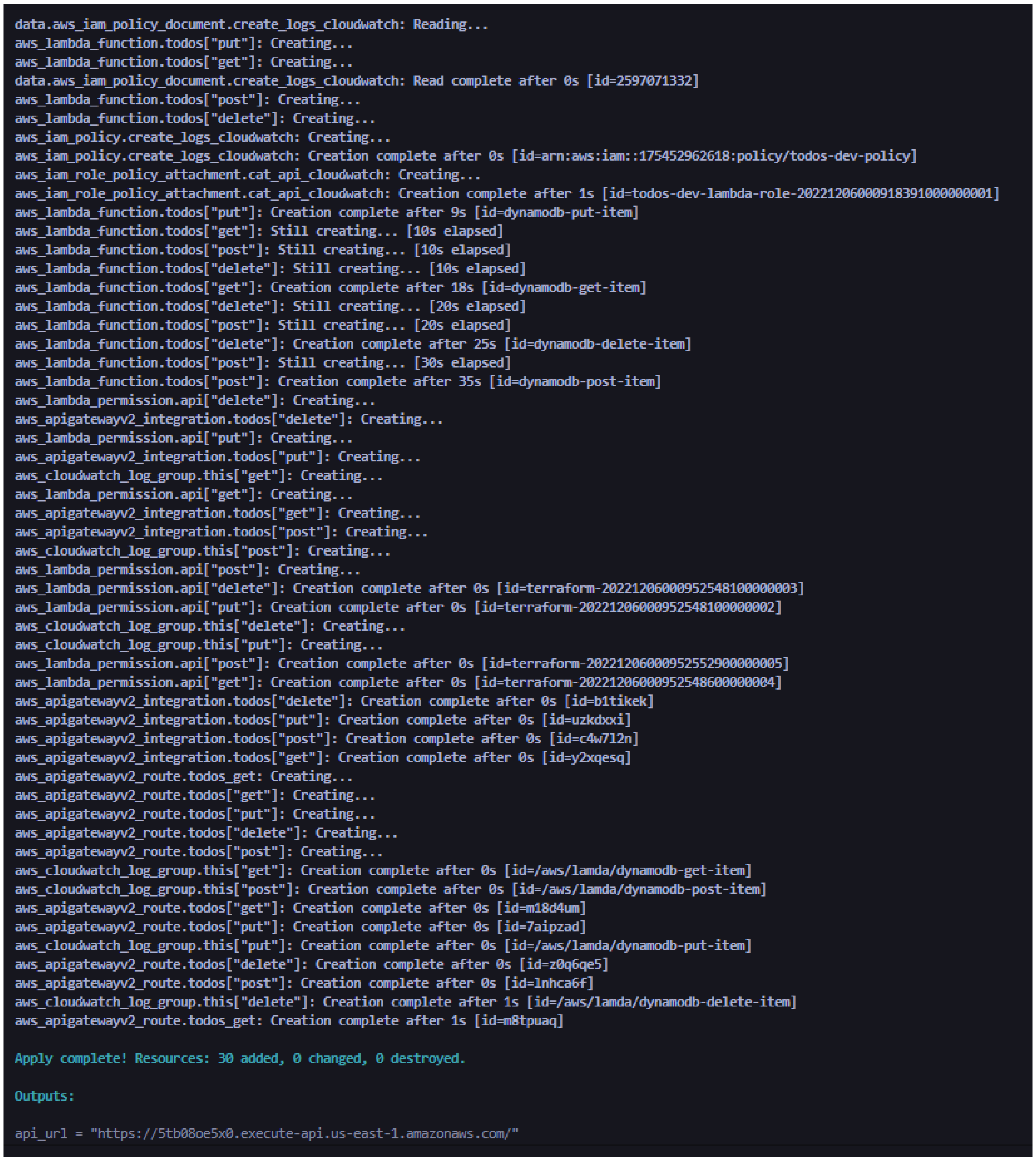


```
Plan: 30 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + api_url = (known after apply)
```

Este nos crea 30 recursos, ahora lo que haremos es aplicar estos cambios.

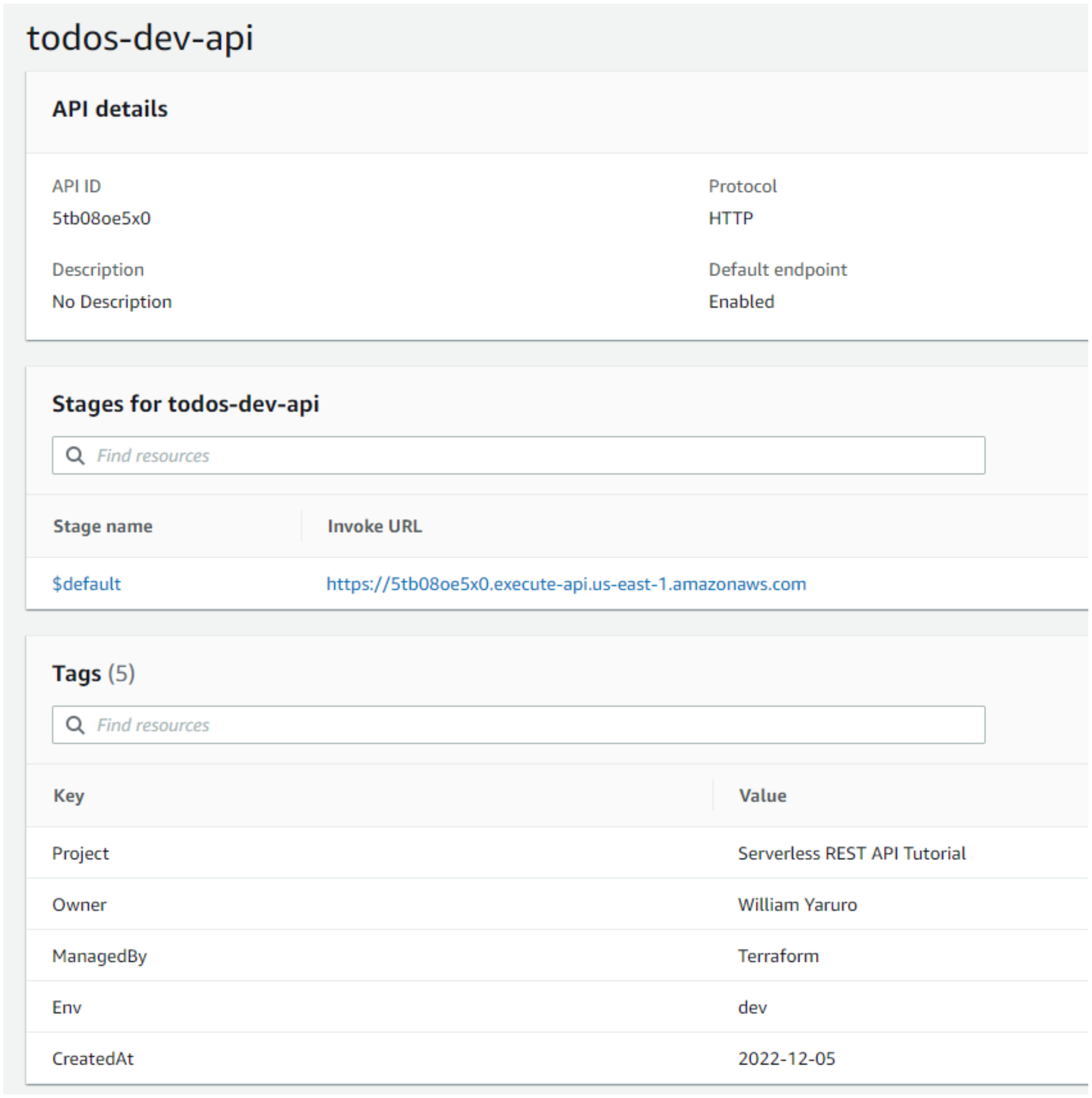


```
PS C:\Users\WillYer\Desktop\AWS\LaboratorioEC2\proyecto-terraform> terraform apply -auto-approve
data.archive_file.todos["get"]: Reading...
data.archive_file.todos["post"]: Reading...
data.archive_file.utils_layer: Reading...
data.archive_file.todos["put"]: Reading...
data.archive_file.todos["delete"]: Reading...
data.archive_file.todos["get"]: Read complete after 0s [id=dc85576a41cc751b7e49c62f6b5c7d134bbccd0
b]
data.archive_file.todos["put"]: Read complete after 0s [id=3b0e2a0fb7cacf8b05ee15140a752f0f1fd3fdc
b]
data.archive_file.todos["delete"]: Read complete after 0s [id=50b5c31c691419b48c1a6b3d10b8a473a75d
505d]
data.archive_file.utils_layer: Read complete after 0s [id=b5c5e65ba64a6338a9cd9cdade2babe9dad4367c
]
data.archive_file.todos["post"]: Read complete after 0s [id=57256bea045aebce8739577e35544ca85aef75
17]
data.aws_iam_policy_document.lambda_assume_role: Reading...
data.aws_iam_policy_document.lambda_assume_role: Read complete after 0s [id=3693445097]

Terraform used the selected providers to generate the following execution plan. Resource actions
are indicated with the following symbols:
  + create
 <= read (data resources)

Terraform will perform the following actions:

  # data.aws_iam_policy_document.create_logs_cloudwatch will be read during apply
  # (depends on a resource or a module with changes pending)
```



```
data.aws_iam_policy_document.create_logs_cloudwatch: Reading...
aws_lambda_function.todos["put"]: Creating...
aws_lambda_function.todos["get"]: Creating...
data.aws_iam_policy_document.create_logs_cloudwatch: Read complete after 0s [id=2597071332]
aws_lambda_function.todos["post"]: Creating...
aws_lambda_function.todos["delete"]: Creating...
aws_iam_policy.create_logs_cloudwatch: Creating...
aws_iam_policy.create_logs_cloudwatch: Creation complete after 0s [id=arn:aws:iam::175452962618:policy/todos-dev-policy]
aws_iam_role_policy_attachment.cat_api_cloudwatch: Creating...
aws_iam_role_policy_attachment.cat_api_cloudwatch: Creation complete after 1s [id=todos-dev-lambda-role-20221206000918391000000001]
aws_lambda_function.todos["put"]: Creation complete after 9s [id=dynamodb-put-item]
aws_lambda_function.todos["get"]: Still creating... [10s elapsed]
aws_lambda_function.todos["post"]: Still creating... [10s elapsed]
aws_lambda_function.todos["delete"]: Still creating... [10s elapsed]
aws_lambda_function.todos["get"]: Creation complete after 18s [id=dynamodb-get-item]
aws_lambda_function.todos["delete"]: Still creating... [20s elapsed]
aws_lambda_function.todos["post"]: Still creating... [20s elapsed]
aws_lambda_function.todos["delete"]: Creation complete after 25s [id=dynamodb-delete-item]
aws_lambda_function.todos["post"]: Still creating... [30s elapsed]
aws_lambda_function.todos["post"]: Creation complete after 35s [id=dynamodb-post-item]
aws_lambda_permission.api["delete"]: Creating...
aws_apigatewayv2_integration.todos["delete"]: Creating...
aws_lambda_permission.api["put"]: Creating...
aws_apigatewayv2_integration.todos["put"]: Creating...
aws_cloudwatch_log_group.this["get"]: Creating...
aws_lambda_permission.api["get"]: Creating...
aws_apigatewayv2_integration.todos["get"]: Creating...
aws_apigatewayv2_integration.todos["post"]: Creating...
aws_cloudwatch_log_group.this["post"]: Creating...
aws_lambda_permission.api["post"]: Creating...
aws_lambda_permission.api["delete"]: Creation complete after 0s [id=terraform-20221206000952548100000003]
aws_lambda_permission.api["put"]: Creation complete after 0s [id=terraform-20221206000952548100000002]
aws_cloudwatch_log_group.this["delete"]: Creating...
aws_cloudwatch_log_group.this["put"]: Creating...
aws_lambda_permission.api["post"]: Creation complete after 0s [id=terraform-20221206000952552900000005]
aws_lambda_permission.api["get"]: Creation complete after 0s [id=terraform-20221206000952548600000004]
aws_apigatewayv2_integration.todos["delete"]: Creation complete after 0s [id=b1tikek]
aws_apigatewayv2_integration.todos["put"]: Creation complete after 0s [id=uzkdxxi]
aws_apigatewayv2_integration.todos["post"]: Creation complete after 0s [id=c4w712n]
aws_apigatewayv2_integration.todos["get"]: Creation complete after 0s [id=y2xqesq]
aws_apigatewayv2_route.todos_get: Creating...
aws_apigatewayv2_route.todos["get"]: Creating...
aws_apigatewayv2_route.todos["put"]: Creating...
aws_apigatewayv2_route.todos["delete"]: Creating...
aws_apigatewayv2_route.todos["post"]: Creating...
aws_cloudwatch_log_group.this["get"]: Creation complete after 0s [id=/aws/lamda/dynamodb-get-item]
aws_cloudwatch_log_group.this["post"]: Creation complete after 0s [id=/aws/lamda/dynamodb-post-item]
aws_apigatewayv2_route.todos["get"]: Creation complete after 0s [id=m18d4um]
aws_apigatewayv2_route.todos["put"]: Creation complete after 0s [id=7aipzad]
aws_cloudwatch_log_group.this["put"]: Creation complete after 0s [id=/aws/lamda/dynamodb-put-item]
aws_apigatewayv2_route.todos["delete"]: Creation complete after 0s [id=z0q6qe5]
aws_apigatewayv2_route.todos["post"]: Creation complete after 0s [id=lnhca6f]
aws_cloudwatch_log_group.this["delete"]: Creation complete after 1s [id=/aws/lamda/dynamodb-delete-item]
aws_apigatewayv2_route.todos_get: Creation complete after 1s [id=m8tpuaq]

Apply complete! Resources: 30 added, 0 changed, 0 destroyed.

Outputs:

api_url = "https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/"
```

Resultados:



todos-dev-api

API details

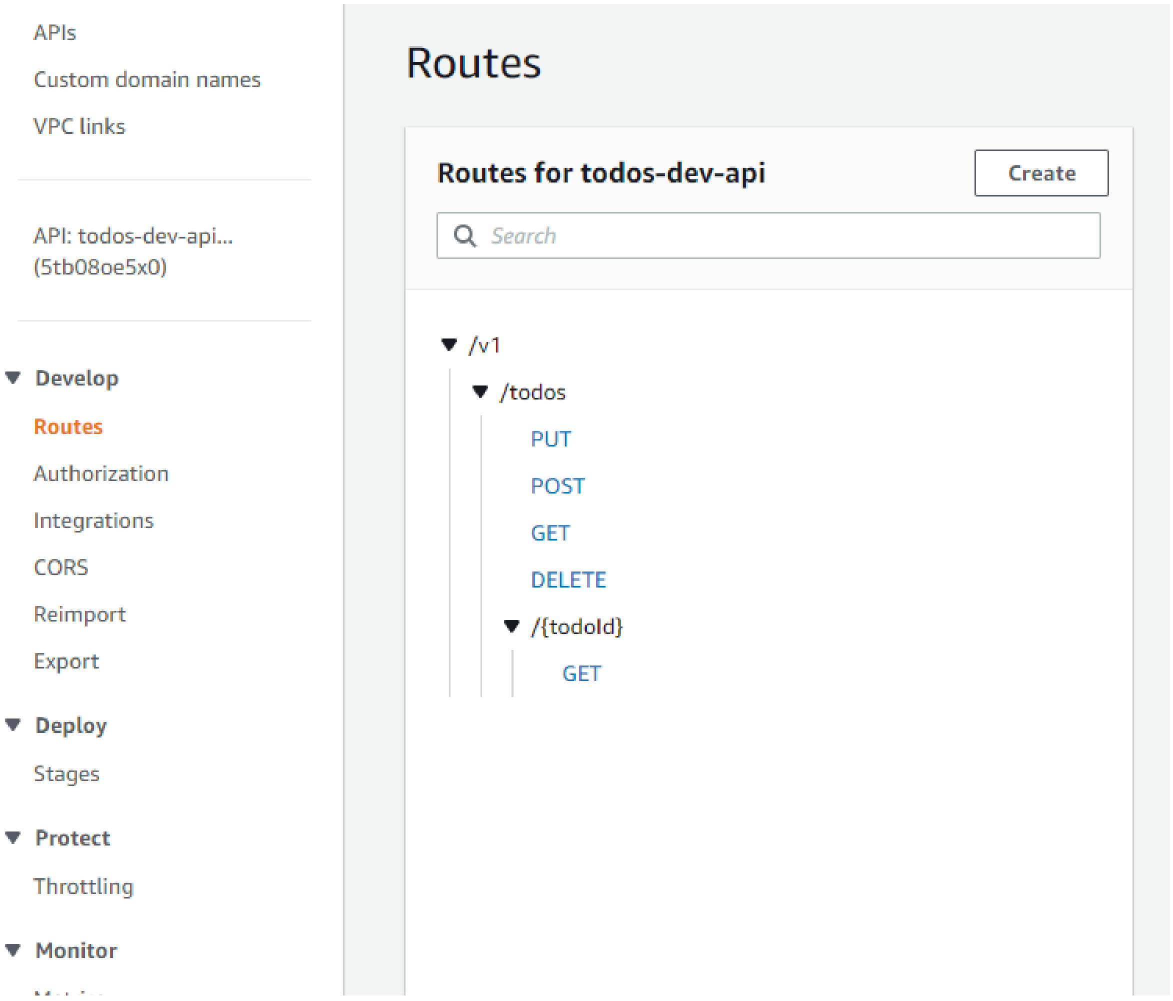
API ID	Protocol
5tb08oe5x0	HTTP
Description	Default endpoint
No Description	Enabled

Stages for todos-dev-api

Stage name	Invoke URL
$default	https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com

Tags (5)

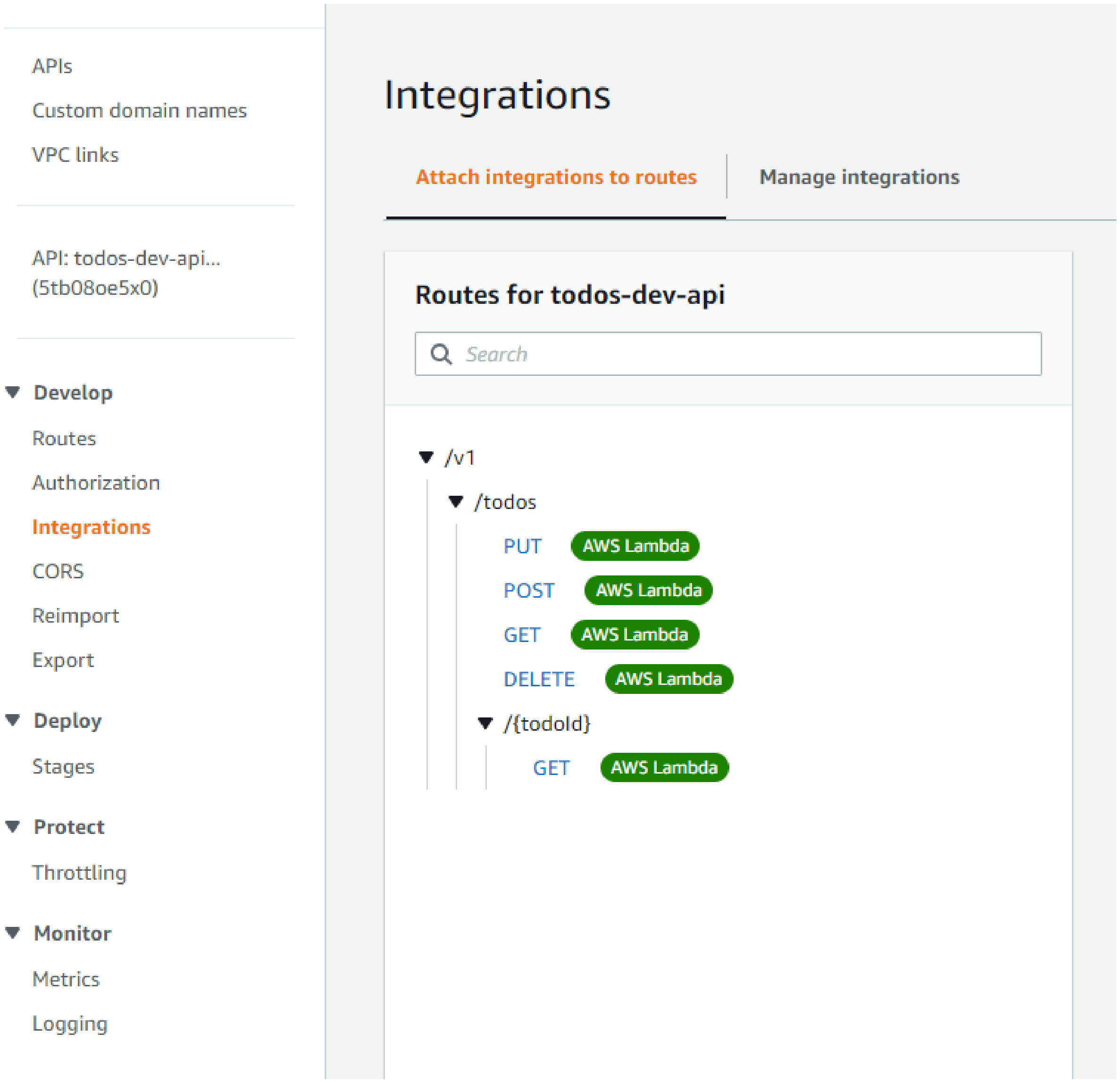
Key	Value
Project	Serverless REST API Tutorial
Owner	William Yaruro
ManagedBy	Terraform
Env	dev
CreatedAt	2022-12-05



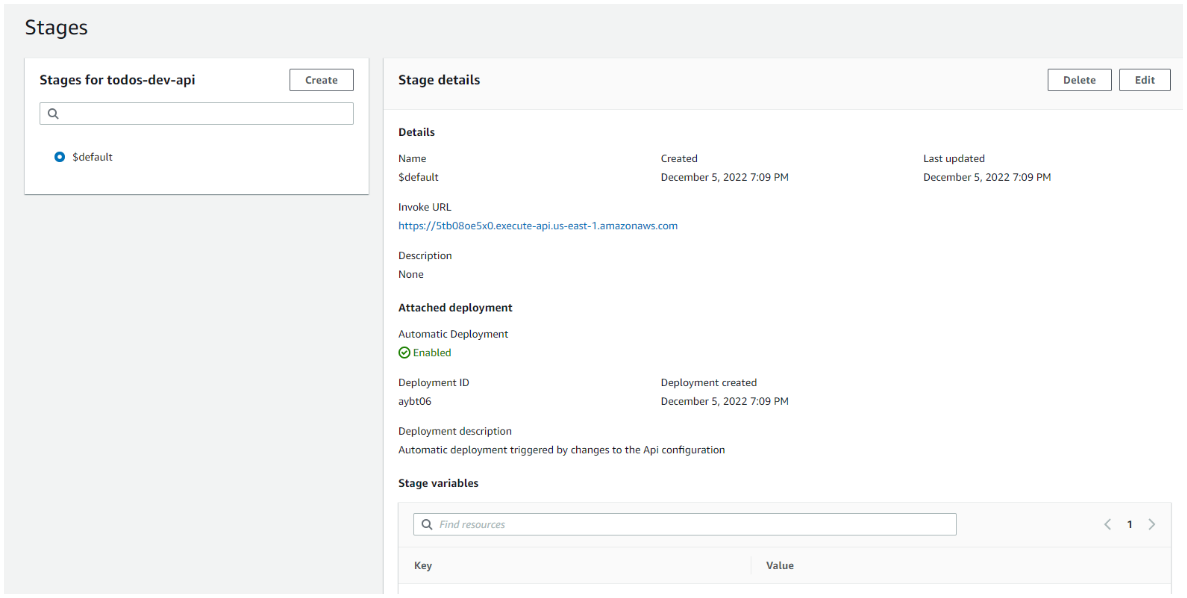
Routes

Routes for todos-dev-api

```
▼ /v1
  ▼ /todos
      PUT
      POST
      GET
      DELETE
    ▼ /{todoId}
        GET
```



```
/v1
  /todos
    PUT     AWS Lambda
    POST    AWS Lambda
    GET     AWS Lambda
    DELETE  AWS Lambda
    /{todoId}
      GET   AWS Lambda
```



Stages

Stages for todos-dev-api

● $default

Stage details

Details

Name	Created	Last updated
$default	December 5, 2022 7:09 PM	December 5, 2022 7:09 PM

Invoke URL
https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com

Description
None

Attached deployment

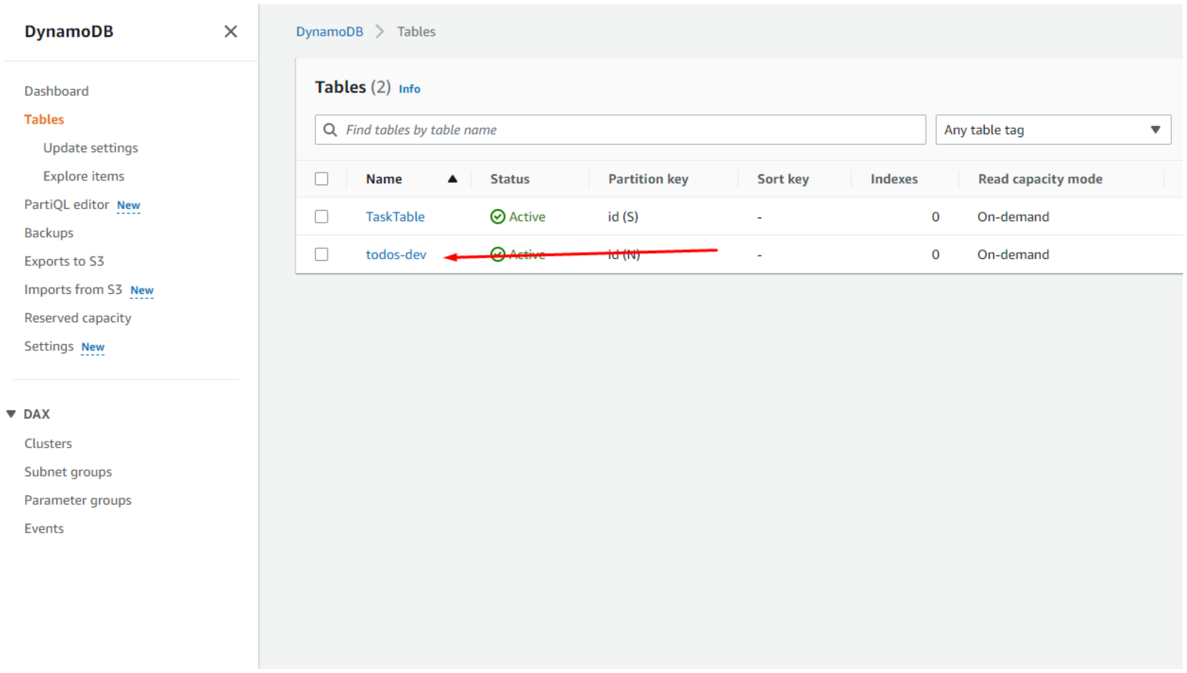
Automatic Deployment
✓ Enabled

Deployment ID	Deployment created
aybt06	December 5, 2022 7:09 PM

Deployment description
Automatic deployment triggered by changes to the Api configuration

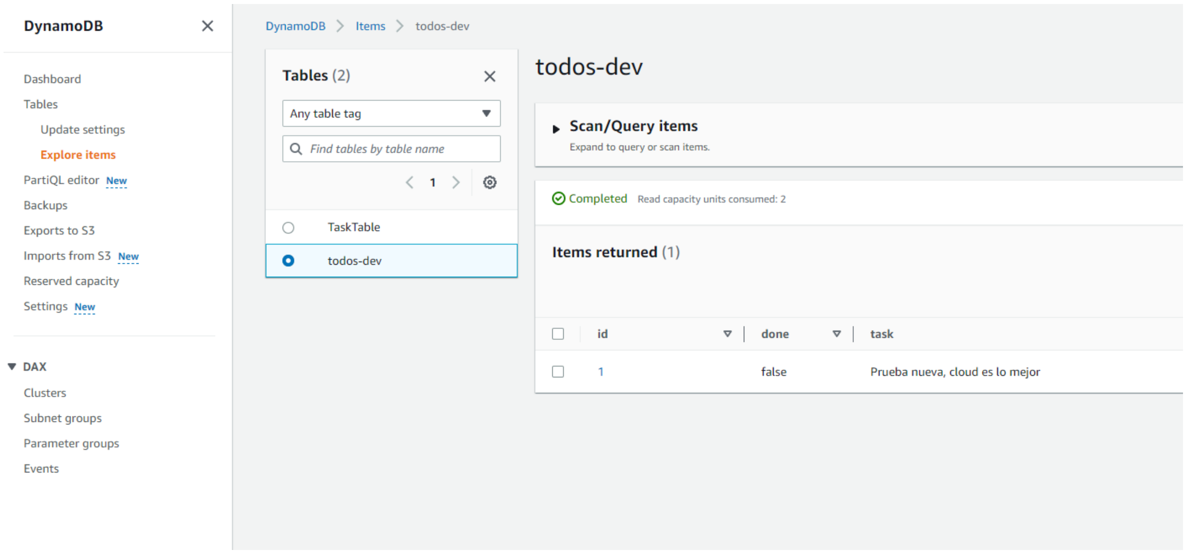
Stage variables

Key	Value



Tables (2) Info

	Name	Status	Partition key	Sort key	Indexes	Read capacity mode
☐	TaskTable	✓ Active	id (S)	-	0	On-demand
☐	todos-dev	✓ Active	id (N)	-	0	On-demand



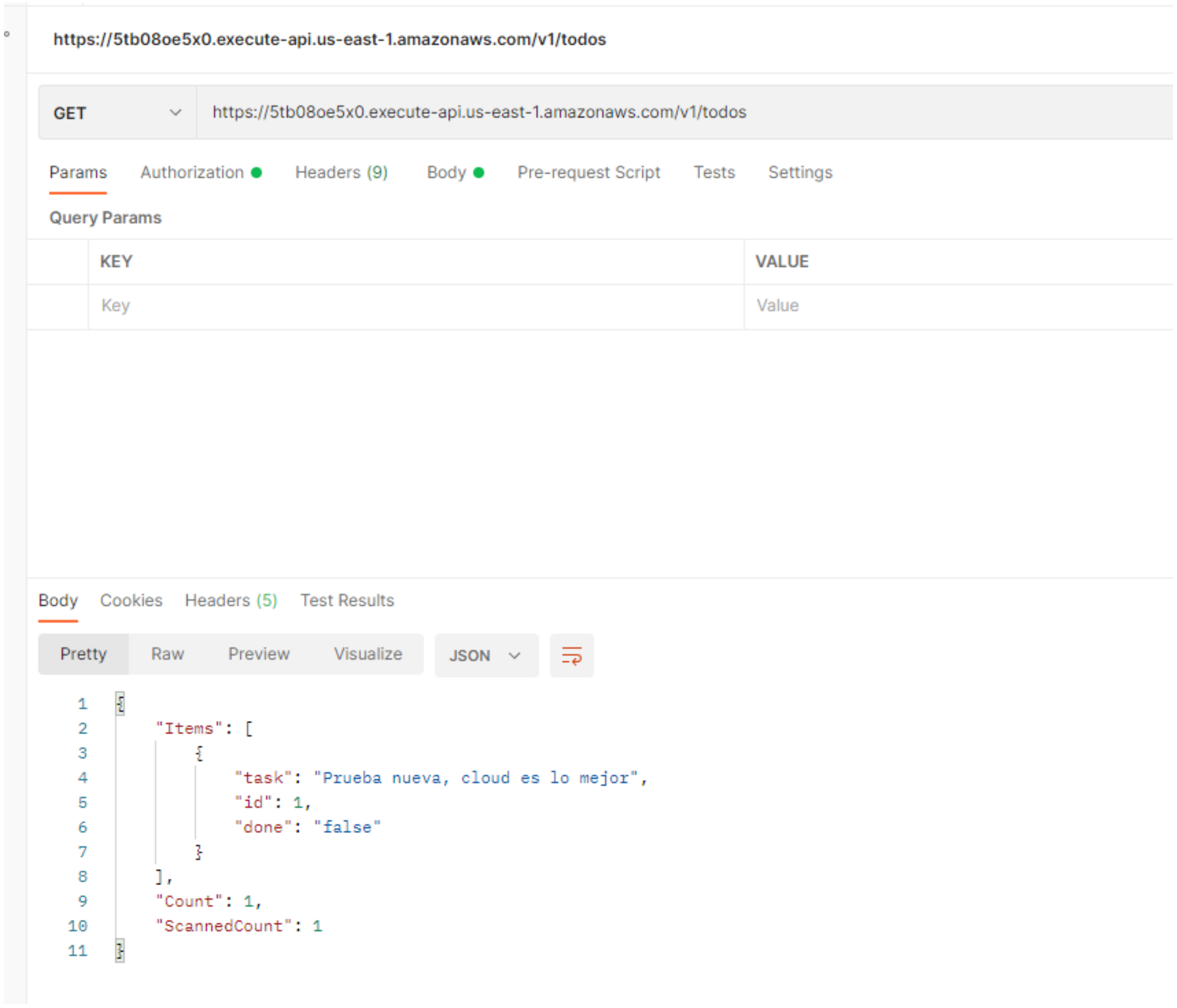
todos-dev

Completed Read capacity units consumed: 2

Items returned (1)

id	done	task
1	false	Prueba nueva, cloud es lo mejor

METODO GET:



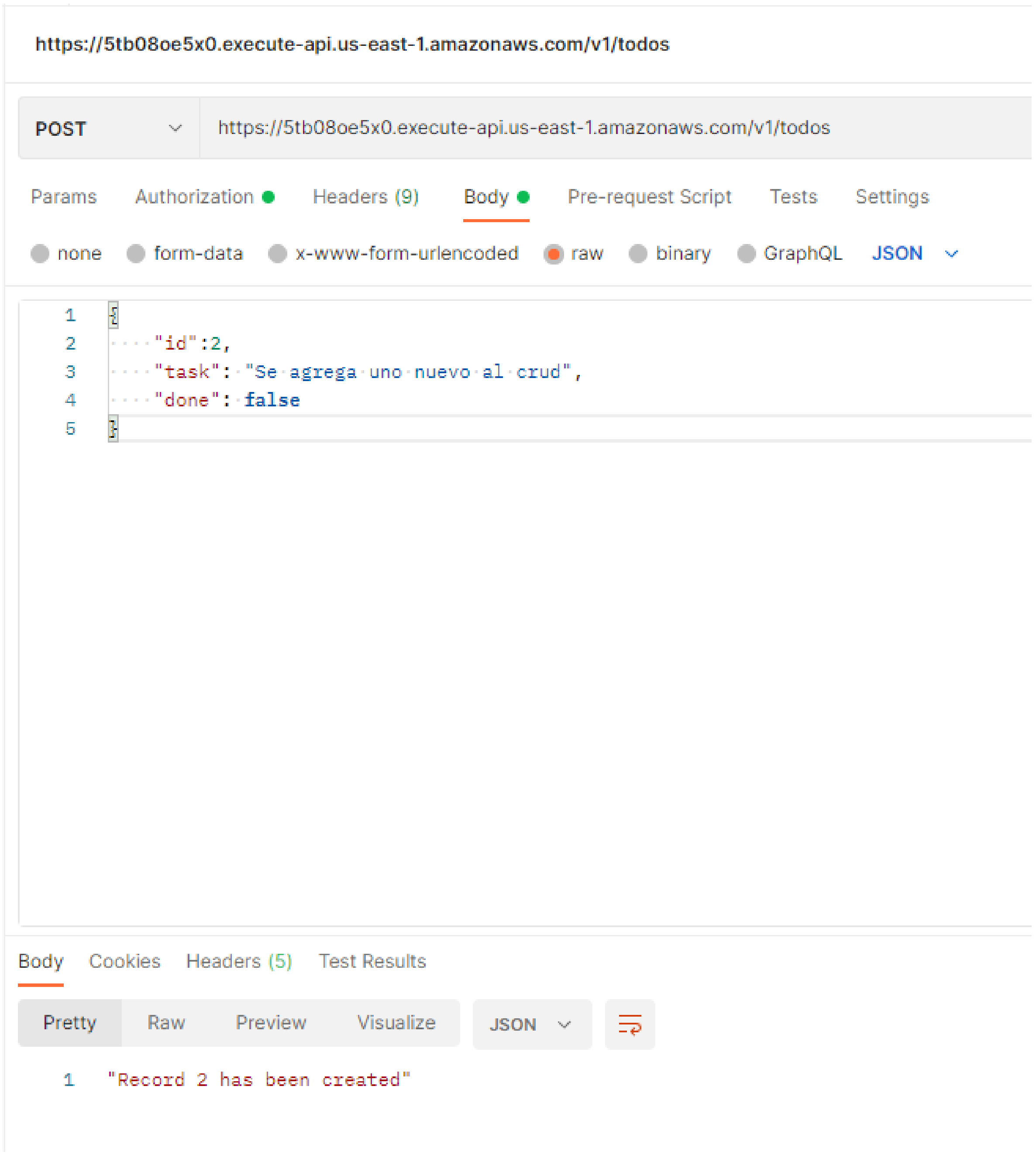
https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/v1/todos

GET https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/v1/todos

KEY	VALUE
Key	Value

```
{
    "Items": [
        {
            "task": "Prueba nueva, cloud es lo mejor",
            "id": 1,
            "done": "false"
        }
    ],
    "Count": 1,
    "ScannedCount": 1
}
```

METODO POST:



https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/v1/todos

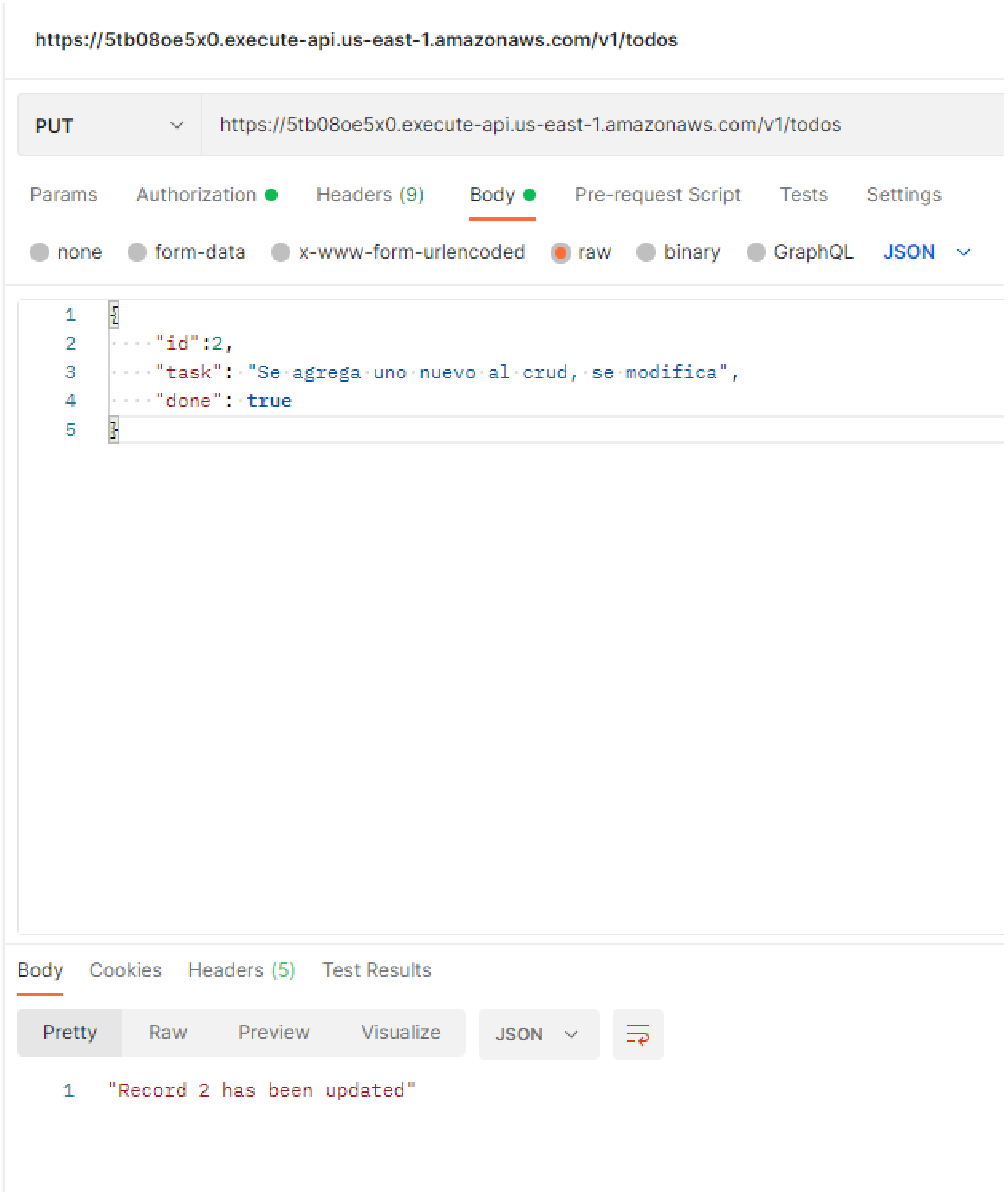
POST https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/v1/todos

none form-data x-www-form-urlencoded [✓ raw] binary GraphQL JSON

```
{
    "id":2,
    "task": "Se agrega uno nuevo al crud",
    "done": false
}
```

```
"Record 2 has been created"
```

Metodo Put:



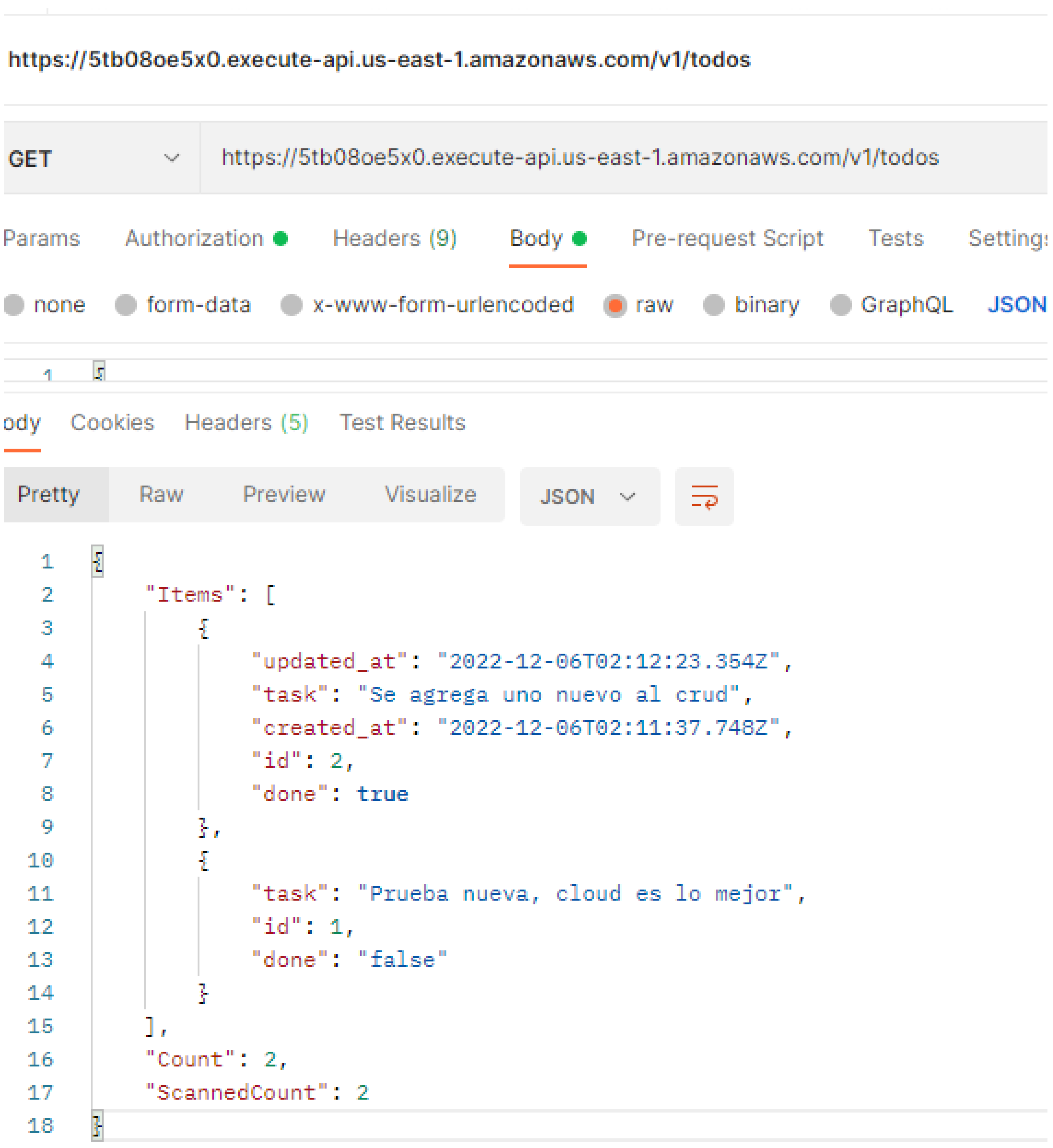
https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/v1/todos

PUT https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/v1/todos

none form-data x-www-form-urlencoded [✓ raw] binary GraphQL JSON

```json
{
    "id":2,
    "task": "Se agrega uno nuevo al crud, se modifica",
    "done": true
}
```

```
"Record 2 has been updated"
```



https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/v1/todos

GET https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/v1/todos

Params | Authorization ● | Headers (9) | Body ● | Pre-request Script | Tests | Setting

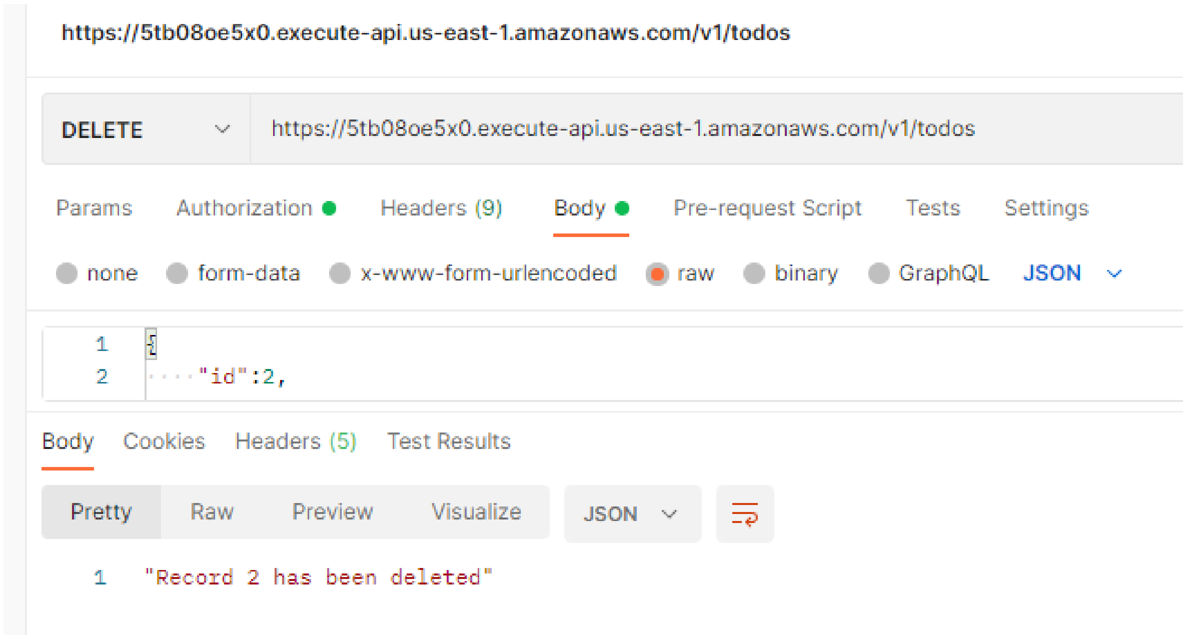
none | form-data | x-www-form-urlencoded | [✓ raw] | binary | GraphQL | JSON

ody | Cookies | Headers (5) | Test Results

Pretty | Raw | Preview | Visualize | JSON

```
{
    "Items": [
        {
            "updated_at": "2022-12-06T02:12:23.354Z",
            "task": "Se agrega uno nuevo al crud",
            "created_at": "2022-12-06T02:11:37.748Z",
            "id": 2,
            "done": true
        },
        {
            "task": "Prueba nueva, cloud es lo mejor",
            "id": 1,
            "done": "false"
        }
    ],
    "Count": 2,
    "ScannedCount": 2
}
```

Metodo Delete:



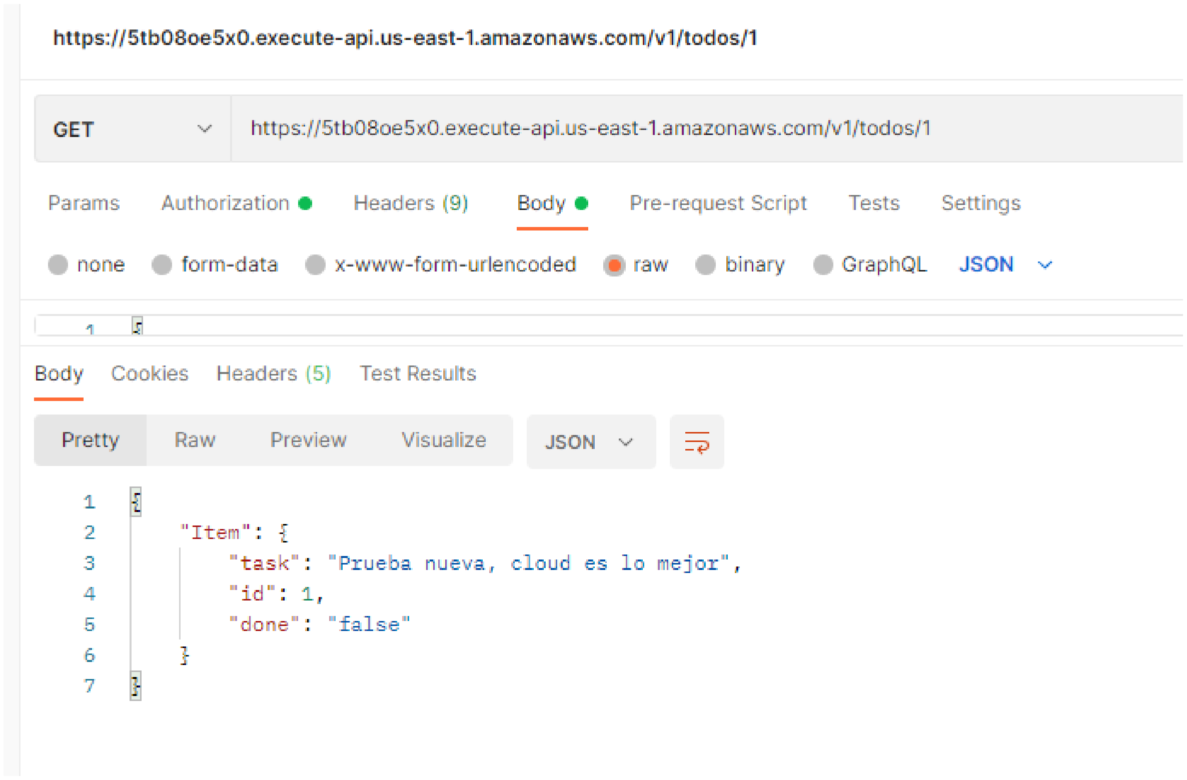
https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/v1/todos

DELETE https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/v1/todos

```
{
    "id":2,
```

```
"Record 2 has been deleted"
```

Metodo Get By Id:



https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/v1/todos/1

GET https://5tb08oe5x0.execute-api.us-east-1.amazonaws.com/v1/todos/1

```
{
    "Item": {
        "task": "Prueba nueva, cloud es lo mejor",
        "id": 1,
        "done": "false"
    }
}
```