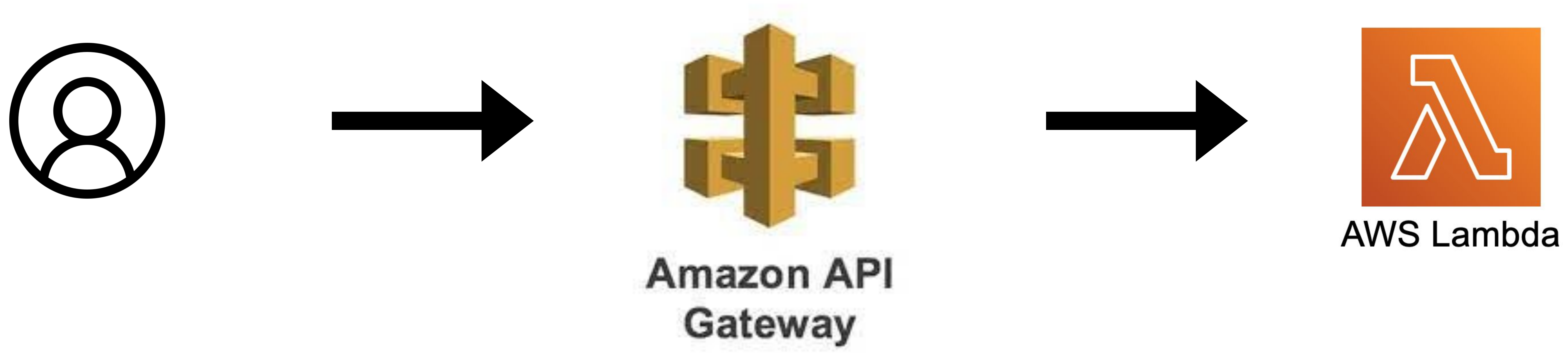
Walk through Lambda deployment with Terraform

Ease lambda deployment replicate with terraform module

Voathnak Lim on Oct 16, 2022

Demonstration

Demonstrate a Hello World Lambda with Terraform



How to get started

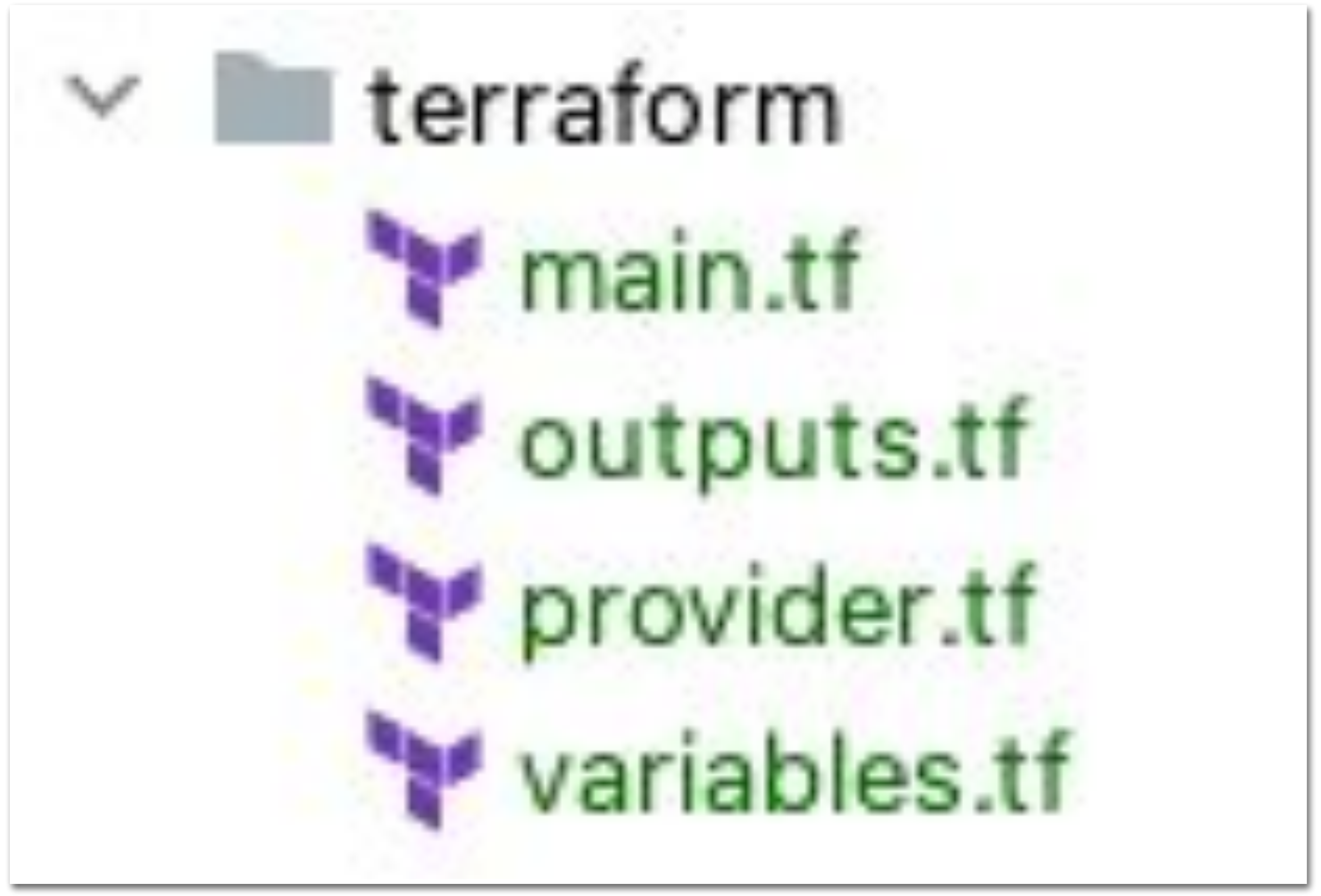
What you need

- Terraform cli installed in your environment
    - *https://www.terraform.io/downloads*
- AWS account
    - *https://aws.amazon.com/resources/create-account/*

How to prepare your directory structure

Setting up project directory structure

1. Create a project directory
2. Create a "terraform" directory inside project dir
3. Create terraform config file(s) such as:
   - provider resource (AWS)
   - api-gateway resource
   - Lambda function resources
4. Add a lambda function code

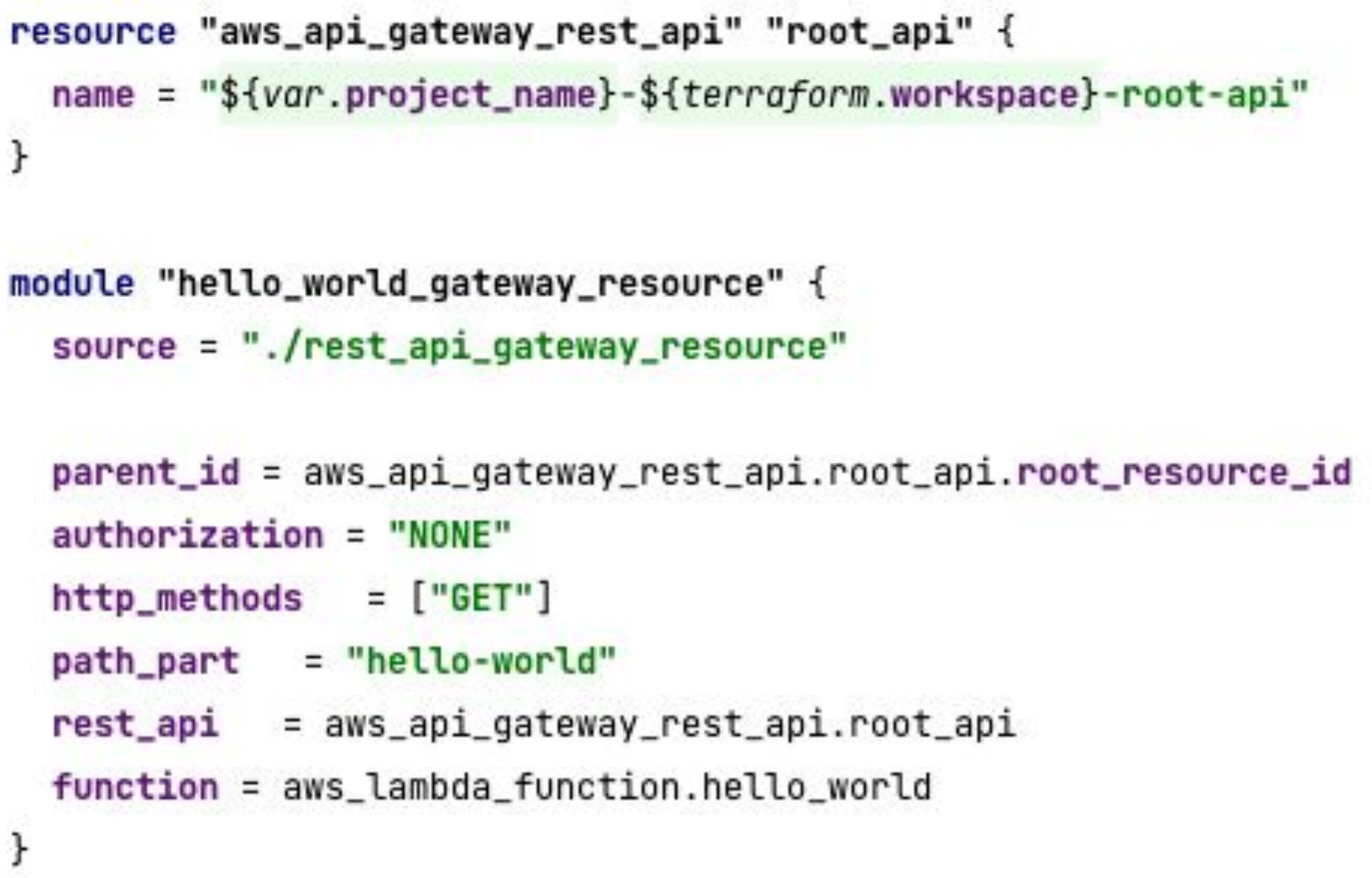


```
terraform
    main.tf
    outputs.tf
    provider.tf
    variables.tf
```

How Terraform script look like

api gateway resources

This is how you create necessary resources to connect api-gateway with lambda

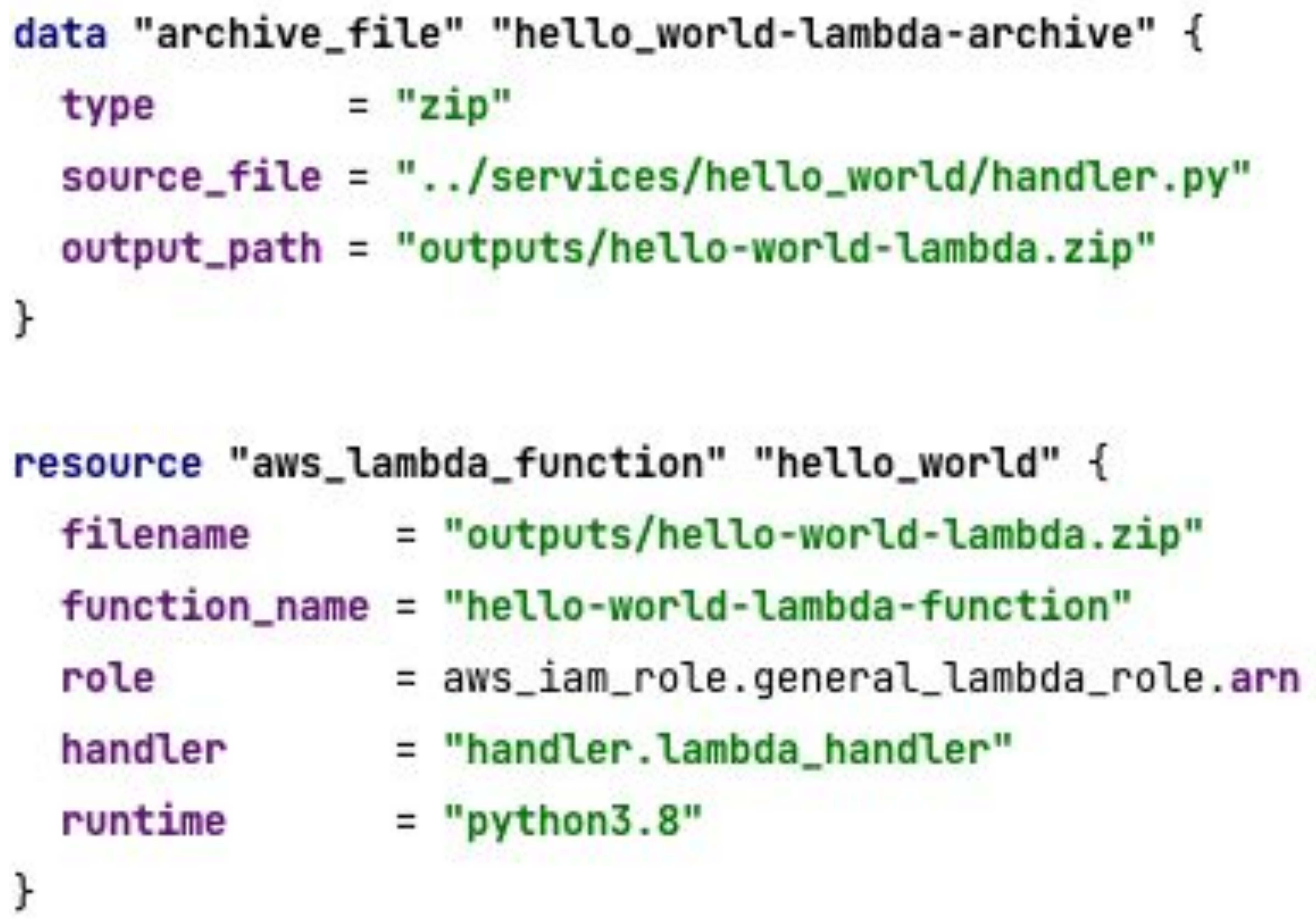


```
resource "aws_api_gateway_rest_api" "root_api" {
  name = "${var.project_name}-${terraform.workspace}-root-api"
}

module "hello_world_gateway_resource" {
  source = "./rest_api_gateway_resource"

  parent_id = aws_api_gateway_rest_api.root_api.root_resource_id
  authorization = "NONE"
  http_methods    = ["GET"]
  path_part    = "hello-world"
  rest_api    = aws_api_gateway_rest_api.root_api
  function = aws_lambda_function.hello_world
}
```

How Terraform script look like

Lambda function resources

This is how you create a lambda function resource in Terraform



```
data "archive_file" "hello_world-lambda-archive" {
  type        = "zip"
  source_file = "../services/hello_world/handler.py"
  output_path = "outputs/hello-world-lambda.zip"
}

resource "aws_lambda_function" "hello_world" {
  filename      = "outputs/hello-world-lambda.zip"
  function_name = "hello-world-lambda-function"
  role          = aws_iam_role.general_lambda_role.arn
  handler       = "handler.lambda_handler"
  runtime       = "python3.8"
}
```

How to deploy with Terraform

```
$ terraform init
```

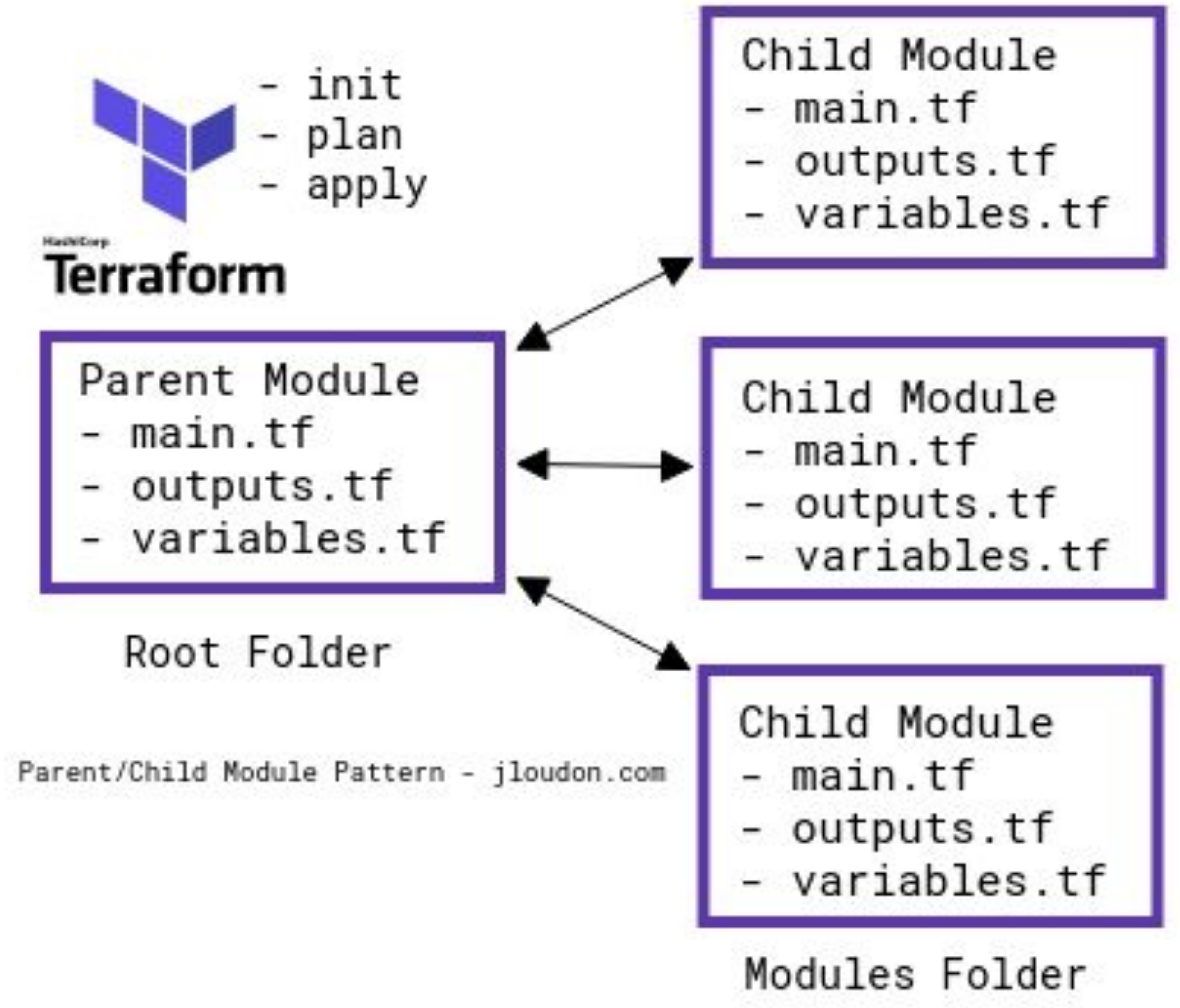
```
$ terraform plan
```

```
$ terraform apply
```

That all it takes!!

Ease your deployment configuration with module

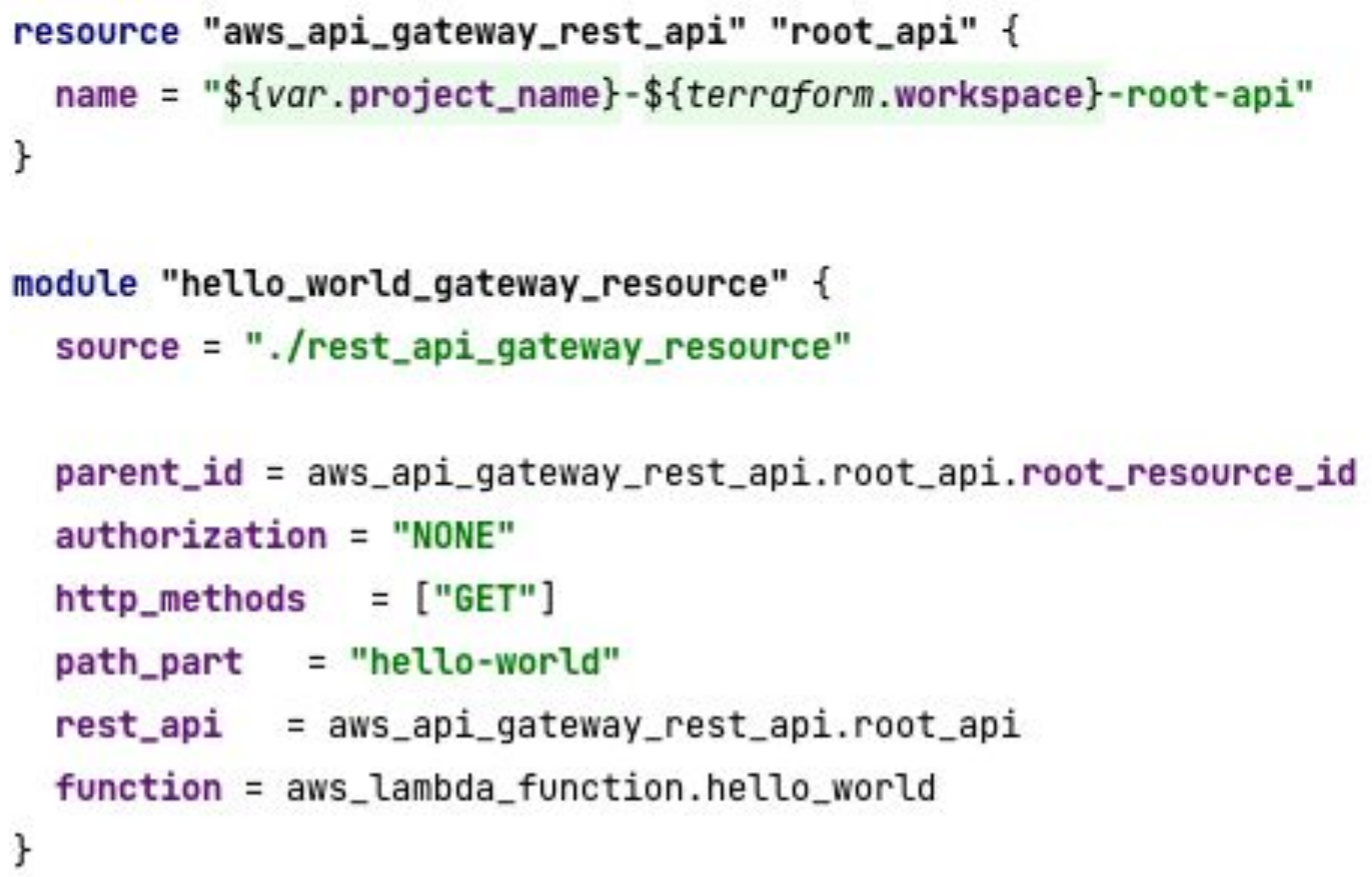
This is how Terraform module look like in directory structure



How Terraform script look like

api gateway resources

This is how you create necessary resources to connect api-gateway with lambda



```
resource "aws_api_gateway_rest_api" "root_api" {
  name = "${var.project_name}-${terraform.workspace}-root-api"
}

module "hello_world_gateway_resource" {
  source = "./rest_api_gateway_resource"

  parent_id = aws_api_gateway_rest_api.root_api.root_resource_id
  authorization = "NONE"
  http_methods    = ["GET"]
  path_part    = "hello-world"
  rest_api    = aws_api_gateway_rest_api.root_api
  function = aws_lambda_function.hello_world
}
```

What have Terraform module help you

Benefits

For Developers

- Help to reduce the time taking on daily deployment tasks
- Building this is similar to building their own serverless framework
- One block of code could meant a lot more (Code reusability)

For the Business

- Because of this the development time and cost can be reduced
- The developers would love this

What have Terraform module help you

This is why

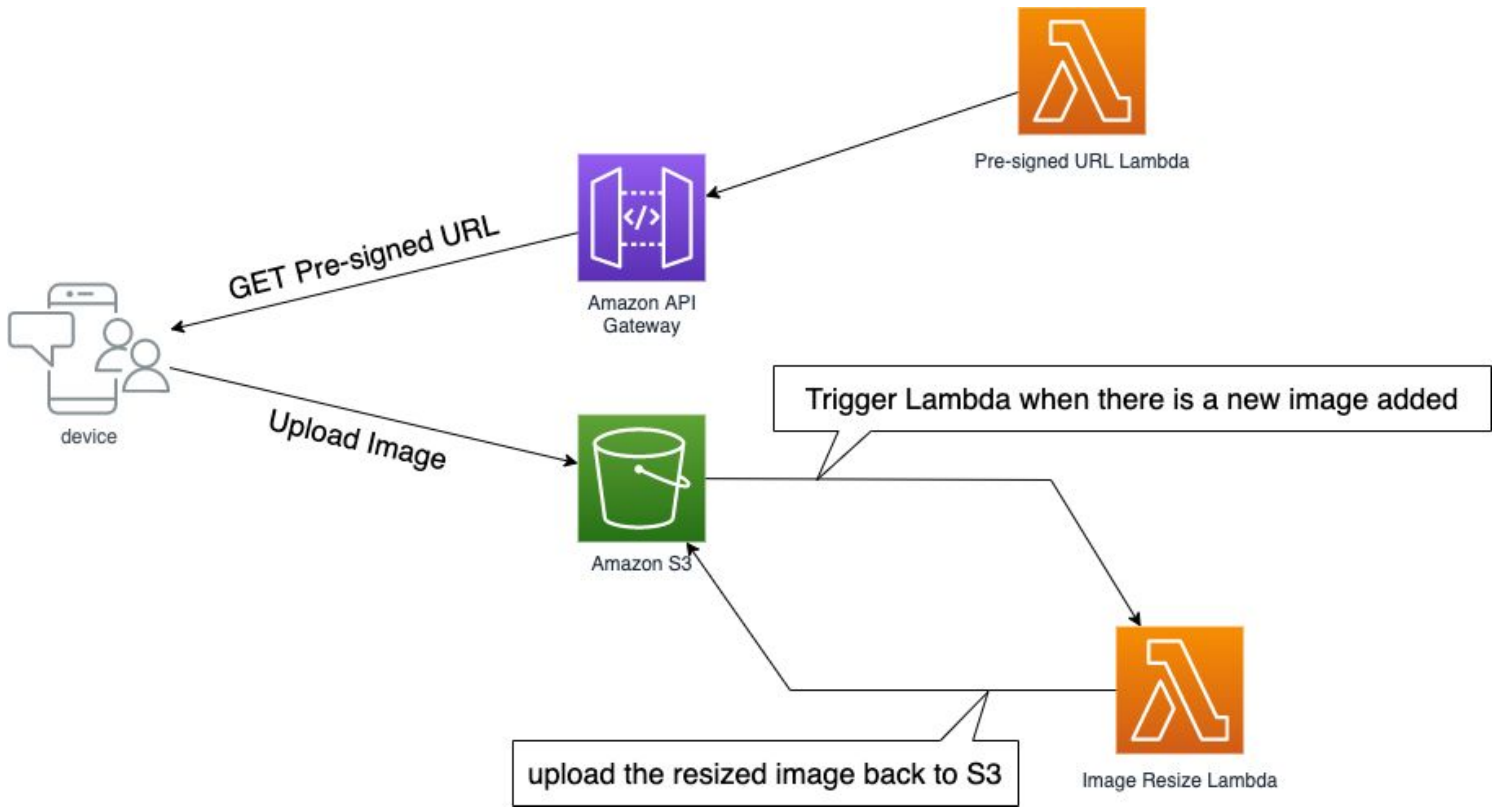
These are the resources that require to create a path of endpoint

- aws_api_gateway_resource
- aws_api_gateway_integration
- aws_api_gateway_integration_response
- aws_api_gateway_method
- aws_api_gateway_method_response
- aws_lambda_permission
- aws_api_gateway_integration_response

aws_api_gateway_integration_response
aws_api_gateway_integration_response
aws_api_gateway_integration
aws_api_gateway_method
aws_api_gateway_method_response
aws_api_gateway_resource
aws_lambda_permis

One block of code could meant a lot more

Just another example



Thank you.