

David Ezeji
12/05/2019

Terraform Project

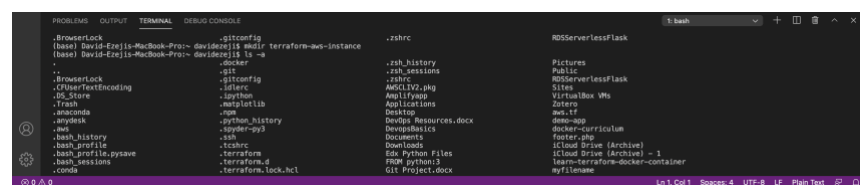
Overview: Created a configuration file in Terraform to deploy resources to AWS

1. Configured the AWS CLI, so Terraform could deploy resources to AWS



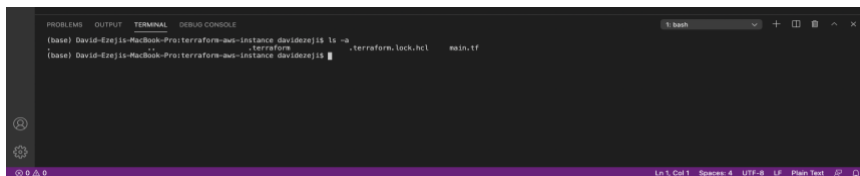
```
David-Ezeji@MacBook-Pro:~$ aws configure
AWS Access Key ID [leave blank]:
AWS Secret Access Key [leave blank]:
Default region name [us-east-1]:
Default output format [json]:
```

2. Created directory for terraform configuration files



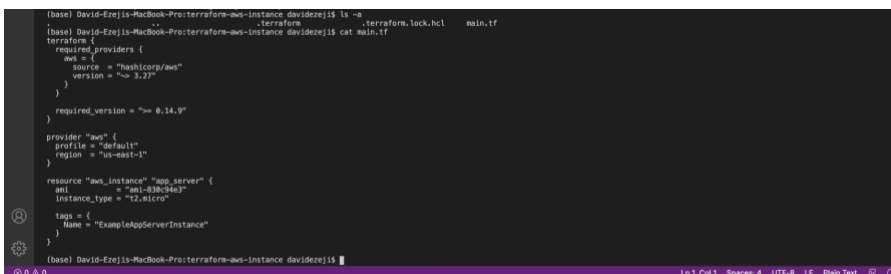
```
David-Ezeji@MacBook-Pro:~$ mkdir terraform-aws-instance
David-Ezeji@MacBook-Pro:~$ cd terraform-aws-instance
David-Ezeji@MacBook-Pro:~/terraform-aws-instance$ ls -la
total 12
drwxr-xr-x 3 David-Ezeji 4096 Nov 5 12:04 .
drwxr-xr-x 1 David-Ezeji 4096 Nov 5 12:04 ..
-rw-r--r-- 1 David-Ezeji  117 Nov 5 12:04 main.tf
```

3. Created configuration file for AWS resources (main.tf)



```
David-Ezeji@MacBook-Pro:~/terraform-aws-instance$ touch main.tf
```

4. Wrote commands in configuration file to create resources in AWS



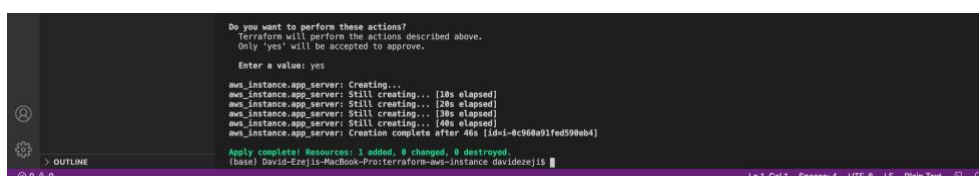
```
David-Ezeji@MacBook-Pro:~/terraform-aws-instance$ cat main.tf
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "~> 3.22"
    }
  }
  required_version = ">= 0.14.9"
}

provider "aws" {
  profile = "default"
  region = "us-east-1"
}

resource "aws_instance" "app_server" {
  ami           = "ami-028434e3"
  instance_type = "t2.micro"

  tags = {
    Name = "ExampleAppServerInstance"
  }
}
```

5. Created the infrastructure

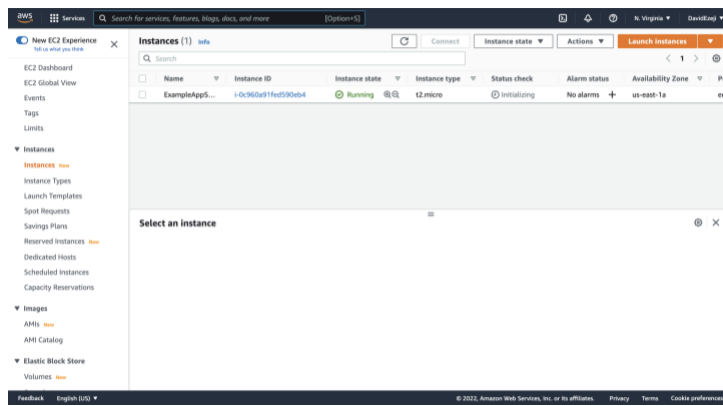


```
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

aws_instance.app_server: Creating...
aws_instance.app_server: Still creating... [10s elapsed]
aws_instance.app_server: Still creating... [20s elapsed]
aws_instance.app_server: Still creating... [30s elapsed]
aws_instance.app_server: Still creating... [40s elapsed]
aws_instance.app_server: Creation complete after 46s [id=i-4c960a91fed594eb4]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```



- Created a file for variables to be used so the values in the configuration file will be flexible and able to be reused

```
(base) David-Ezejis-MacBook-Pro:terraform-aws-instance davidezejis$ cat variables.tf
variable "instance_name" {
  description = "Value of the Name tag for the EC2 instance"
  type = string
  default = "ExampleAppServerInstance"
}
```

```
(base) David-Ezejis-MacBook-Pro:terraform-aws-instance davidezejis$ cat main.tf
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "~> 3.27"
    }
  }
  required_version = ">= 0.14.9"
}

provider "aws" {
  profile = "default"
  region = "us-east-1"
}

resource "aws_instance" "app_server" {
  ami = "ami-048ff3da02834bdc"
  instance_type = "t2.micro"

  tags = {
    Name = "var.instance_name"
  }
}
```

- Now the variables can be changed continuously when recreating the infrastructure

```
(base) David-Ezejis-MacBook-Pro:terraform-aws-instance davidezejis$ terraform apply -var "instance_name=David's_instance"
aws_instance.app_server: Refreshing state... [id=i-0c960a91fed590eb4]

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.

Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
(base) David-Ezejis-MacBook-Pro:terraform-aws-instance davidezejis$ vi main.tf
(base) David-Ezejis-MacBook-Pro:terraform-aws-instance davidezejis$
```