Lab 6. Terraform variables

Background:

Here, we will use terraform variables to parameterize our terraform templates and make then dynamic rather than just supplying static values.

Tasks:

1. Make a directory called ‘lab2’ underneath the terraform-labs directory.

2. Change into the directory.

3.Create the following files: *main.tf resource.tf and vars.tf*

Here is the source code for the main.tf file:

```
provider "aws" {
    access_key = "AKIAIZAHH7GJN6ASXVVA"
    secret_key = "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"
    region = "${var.region}"

}
```

Here is the source code for the resource.tf file.

```
resource "aws_vpc" "main_vpc" {
    cidr_block = "${var.vpc_cidr}"
    instance_tenancy = "default"
    tags {
        Name = "Main"
        Location = "London"
    }
}

resource "aws_subnet" "vpc_subnet" {
    vpc_id = "${aws_vpc.main_vpc.id}"
    cidr_block = "${var.vpc_cidr}"
    availability_zone = "us-east-1a"
    tags {
        Name = "subnet1"
   }
}
```

Here is the source code for vars.tf

```
variable "region" {
    default = "us-east-1"
}

variable "vpc_cidr" {
    default = "192.168.0.0/16"
}

variable "vpc_subnet_cidr" {
    default = "192.168.100.0/24"
}

variable "ami_instance" {
    default = "ami-0ac019f4fcb7cb7e6"
}

variable "ami_instance_type" {
    default = "t2.micro"
}

variable "aws_availability_zone" {
    default = "us-east-1a"
}
```

4. Run the following commands:

```
> terraform get
 > terraform init
```

Note that the '>' refers to the bash shell prompt and is not part of the command.

This command initializes the terraform directory structure.

5. Run the following command:

```
> terraform plan
```

This should print out what actions terraform will take.

6. Run the following command:

```
> terraform apply
```

Assuming that this works correctly, AWS will create a VPC, and a subnet located defined in the vpc.

7.. Run the following:

```
> terraform destroy
```

This will now destroy the formerly created AWS vpc, and all subnets.