**Lesson 10 Lesson-End Project**

**Building and Testing a Terraform Module**

**Project agenda:** To build and test a Terraform Module

**Description:** A Terraform module is a set of Terraform configuration files in a single directory. Even a simple configuration consisting of a single directory with one or more .tf files is a module. In this project, you have been asked to build and test a Terraform Module.

**Tools required:** Terraform, AWS CLI

**Prerequisites:** You must have Terraform installed in the lab to proceed. In case you do not have it please refer to Lesson 10 Demo 1 to install and set up Terraform. Also, you should have AWS CLI installed to configure AWS.

**Expected deliverables:**

Write your Terraform VPC Module code

Write your main Terraform Project code

Deploy your code and test out your Module

**Steps to be followed:**

1. Configuring the AWS CLI from the terminal
2. Creating the directory structure for the Terraform project
3. Writing your Terraform VPC Module code
4. Writing your main Terraform project code
5. Deploying your code and test out your Module

**Step 1: Configuring the AWS CLI from the terminal**

* 1. AWS CLI should be installed in your system. However, you can run the below commands to install AWS CLI:

***pip install awscli***

***sudo apt-get update***

* 1. Execute the following command to set up your AWS credentials as environment variables:

***aws configure***

**Note**: Keep your security credentials handy to authenticate to your AWS account.

* 1. Follow the prompts to input your AWS Access Key ID and Secret Access Key

**Enter your AWS access key:**

**Enter your AWS secret access key:**

**Region:** us-east-1

**Default output format:** none

* 1. The configuration process stores your credentials in a file at **~/.aws/credentials.** Navigate to the directory and add the security token of your AWS lab as we are using an AWS user account

***cd .aws***

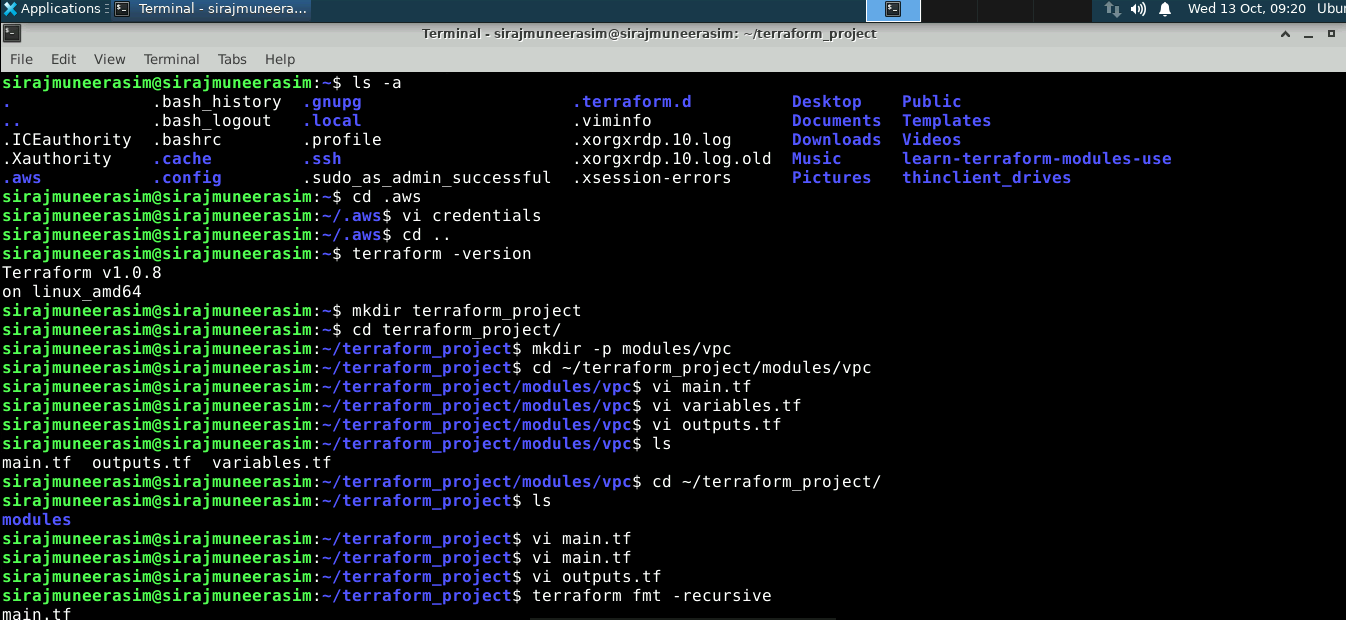
***vi credentials***

**To save and exit:** Press **Esc** to enter Command mode, and then type **:wq** to save and quit the file

**Step 2: Creating the directory structure for the Terraform project**

2.1 Check the installation and functioning of Terraform using the below command:

***terraform --version***



2.2 Create a new directory to house your Terraform code called **terraform\_project**:

***mkdir terraform\_project***

2.3 In the main project directory, create a custom directory called modules and a directory inside it called vpc:

***cd terraform\_project***

***mkdir -p module/vpc***

**Step 3: Writing your Terraform VPC Module code**

3.1 In the vpc directory, create a new file called **main.tf** and add the provided code:

**cd** ~/***terraform\_project/module/vpc***

***vi main.tf***

**Code:**

provider "aws" {

region = var.region

}

resource "aws\_vpc" "this" {

cidr\_block = "10.0.0.0/16"

}

resource "aws\_subnet" "this" {

vpc\_id = aws\_vpc.this.id

cidr\_block = "10.0.1.0/24"

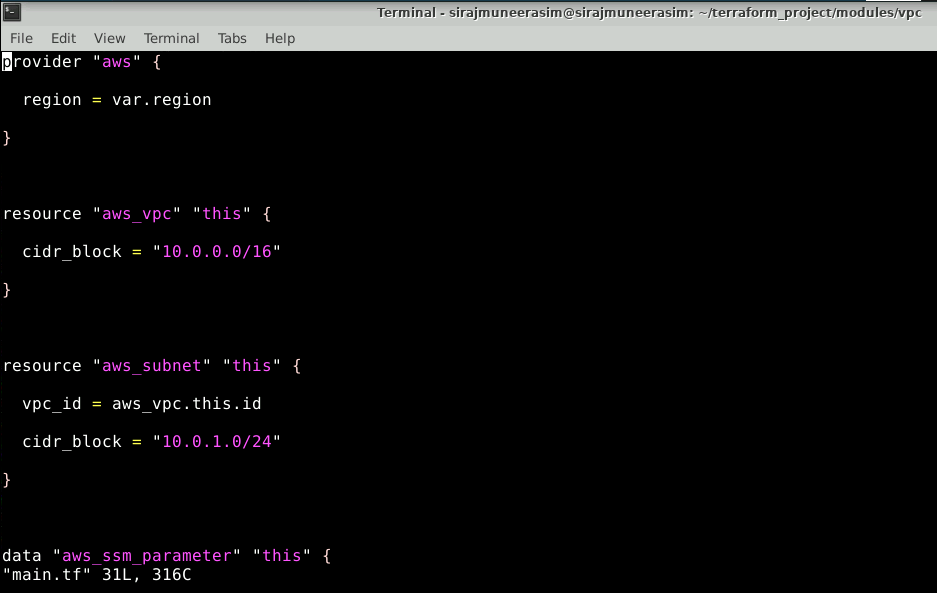
}

data "aws\_ssm\_parameter" "this" {

name = "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86\_64-gp2"

}

**To save and exit:** Press **Esc** to enter Command mode, and then type **:wq** to save and quit the file



3.2 Create a new file called **variables.tf** and add the provided code:

***vi variables.tf***

**Code:**

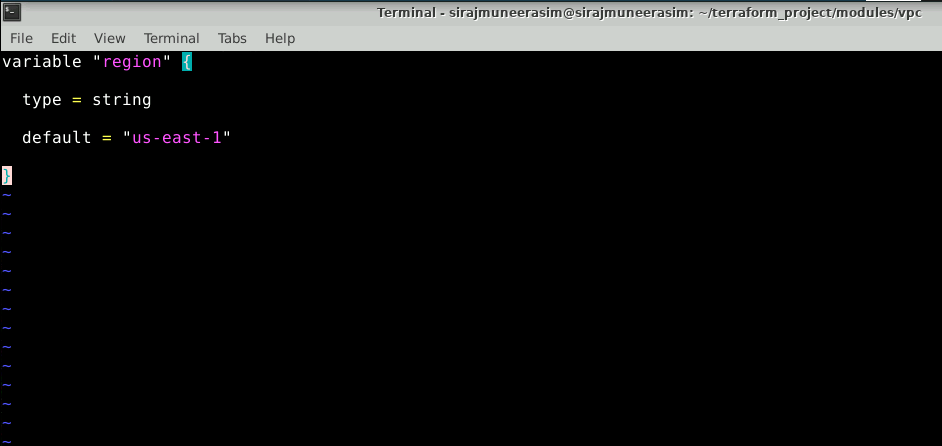
variable "region" {

type = string

default = "us-east-1"

}

**To save and exit**: Press **Esc** to enter Command mode, and then type **:wq** to write and quit the file



3.3 Create a new file called **outputs.tf** and add the provided code:

***vi outputs.tf***

**Code:**

output "subnet\_id" {

value = aws\_subnet.this.id

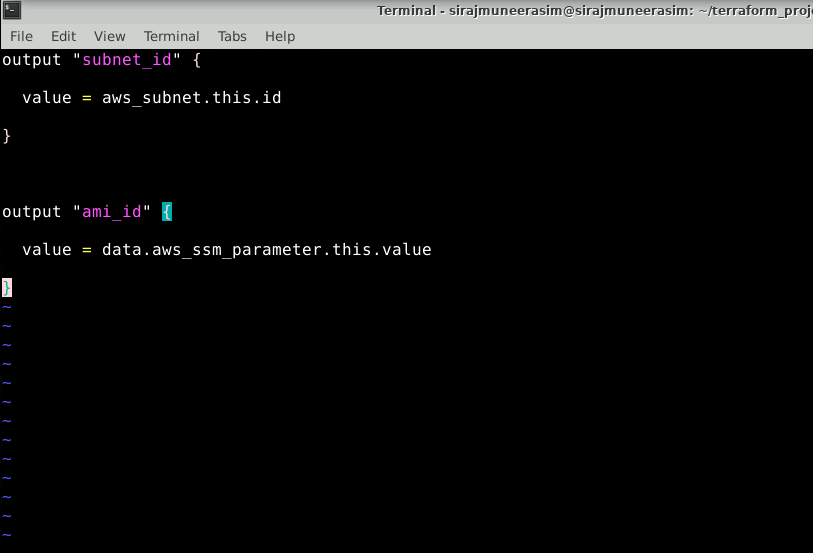
}

output "ami\_id" {

value = data.aws\_ssm\_parameter.this.value

}

**To save and exit**: Press Esc to enter Command mode, and then type **:wq** to save and quit the file



**Step 4: Writing your main Terraform project code**

4.1 In the **terraform\_project** directory, create a new file called **main.tf** and add the provided code which invokes the VPC module created earlier:

**cd ~/*terraform\_project/***

***vi main.tf***

**Code:**

variable "main\_region" {

type = string

default = "us-east-1"

}

provider "aws" {

region = var.main\_region

}

module "vpc" {

source = "./modules/vpc"

region = var.main\_region

}

resource "aws\_instance" "my-instance" {

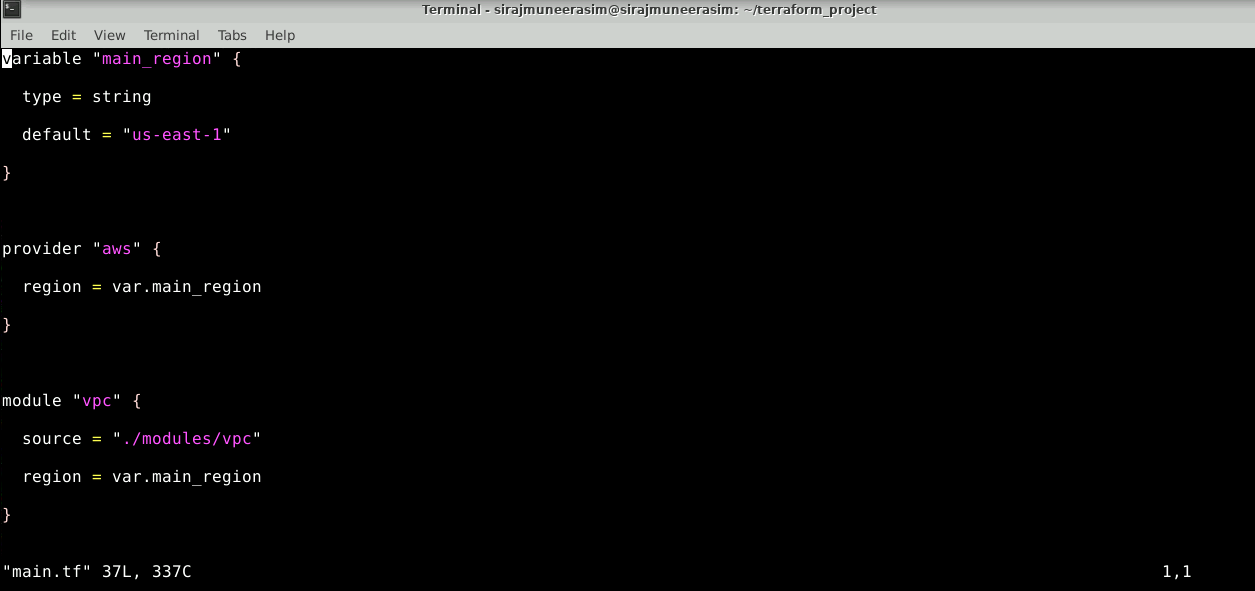
ami = module.vpc.ami\_id

subnet\_id = module.vpc.subnet\_id

instance\_type = "t2.micro"

}

**To save and exit**: Press Esc to enter Command mode, and then type **:wq** to save and quit the file



4.2 Create a new file called **outputs.tf** and add the provided code:

***vi outputs.tf***

**Code:**

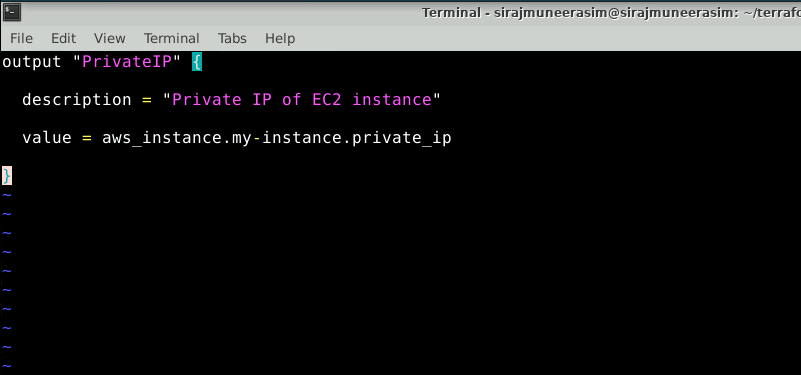
output "PrivateIP" {

description = "Private IP of EC2 instance"

value = aws\_instance.my-instance.private\_ip

}

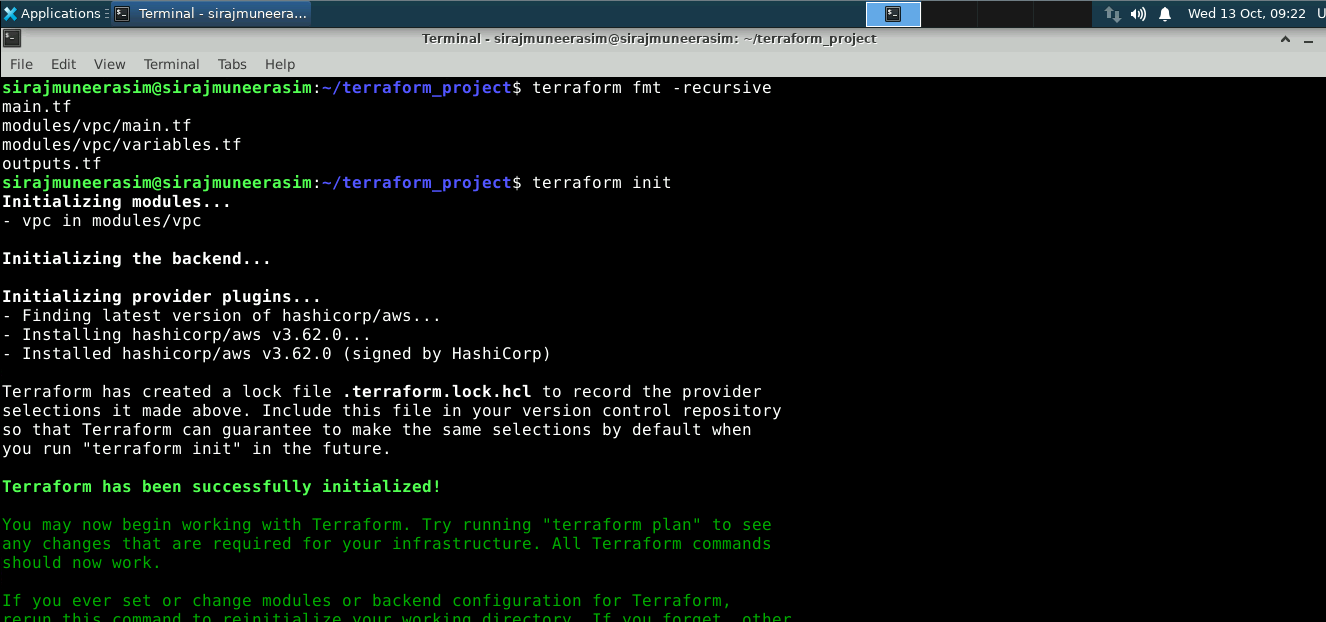
**To save and exit**: Press Esc to enter Command mode, and then type **:wq** to save and quit the file



**Step 5: Deploying your code and test out your Module**

5.1 Format the code in all the files using the below command:

***terraform fmt -recursive***

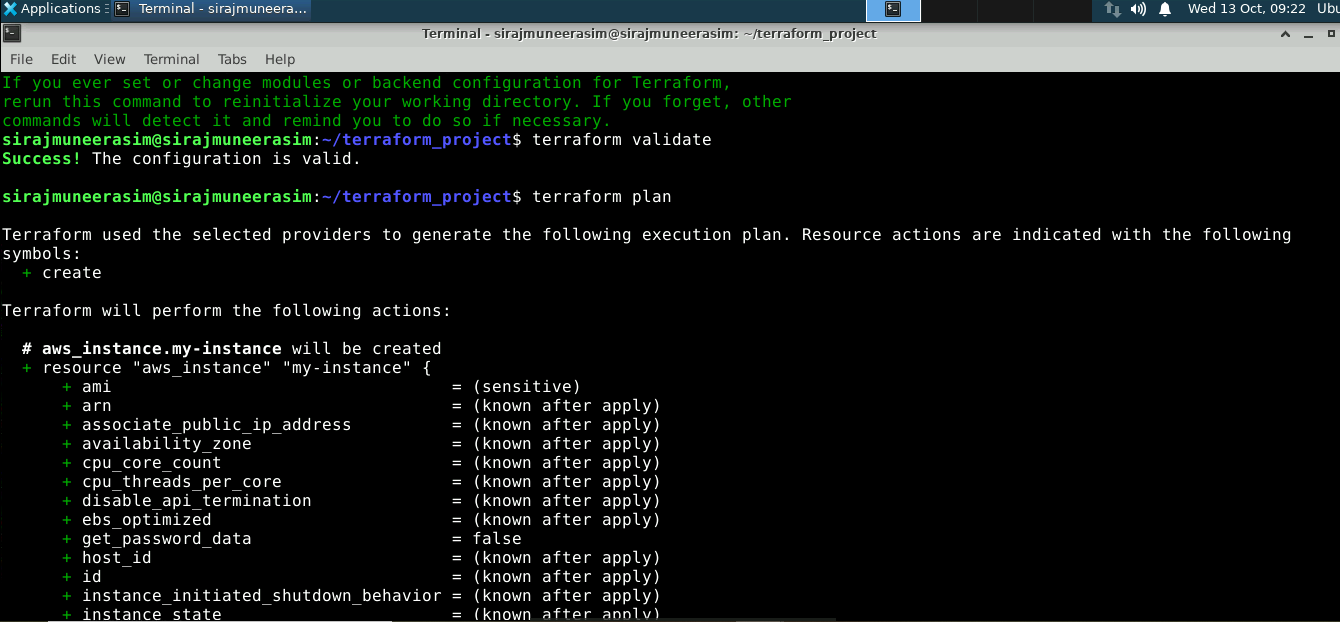


5.2 Initialize the Terraform configuration to fetch any required providers and get the code referenced in the module block with the below command:

***terraform init***

5.3 Validate the code using the below command:

***terraform validate***

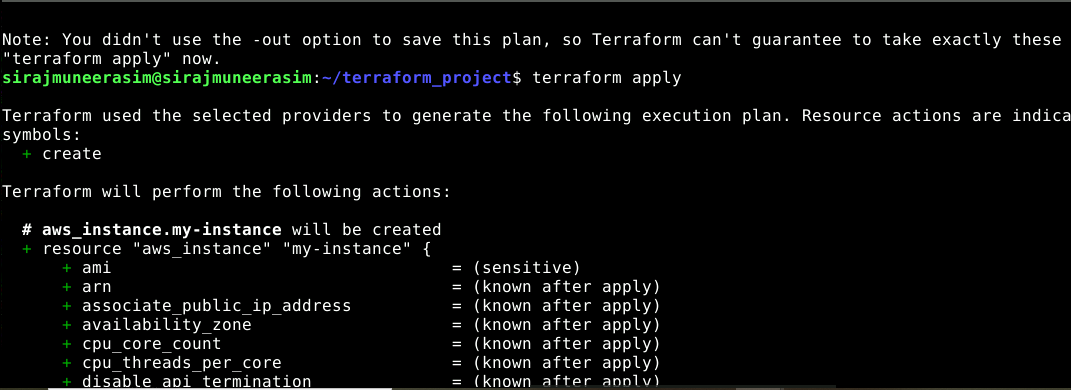


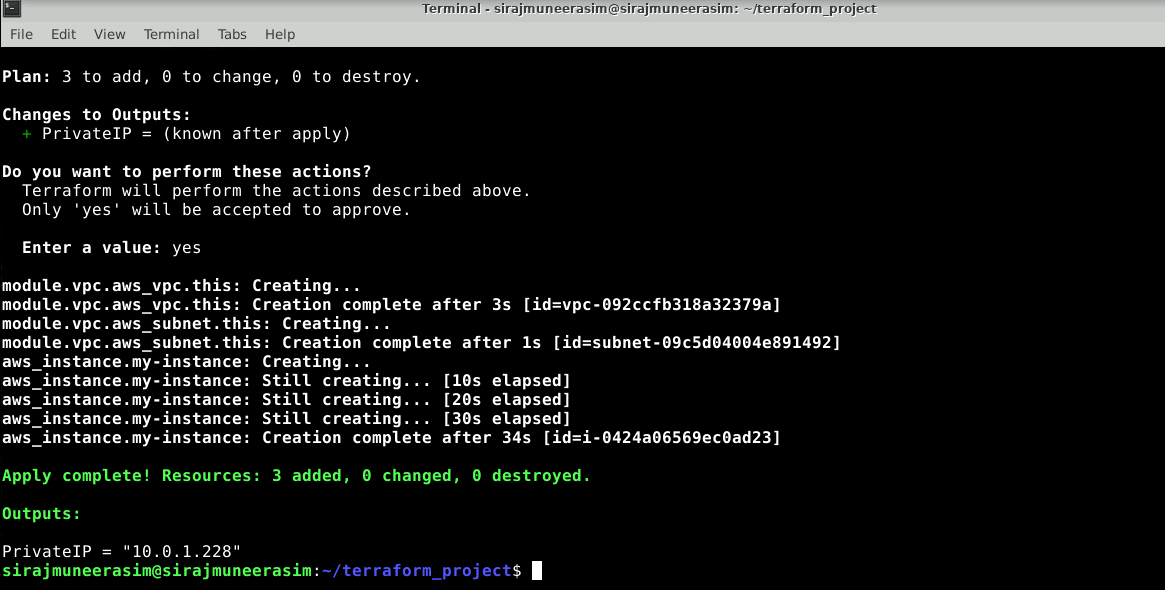
5.4 Review the actions that will be performed when you deploy the code using the below command:

***terraform plan***

5.5 Deploy the code with the below command:

***terraform apply --auto-approve***





5.6 You can view the resources that are created using the below command:

***terraform state***