Terraform Projects: Automating Your Infrastructure Journey!

This document will walk you through automating the creation of a GitHub repository and provisioning an EC2 instance in the AWS using Terraform.

Project 1: Creating a GitHub Repository with Terraform

In this project, we'll leverage Terraform's capabilities to automate the creation of a brand new repository on your GitHub account.

Prerequisites:

- **Terraform installed:** Download and install Terraform based on your operating system from the official website https://developer.hashicorp.com/terraform/install
- **GitHub Personal Access Token:** Generate a token with "repo" access permission from your GitHub settings https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/managing-your-personal-access-tokens

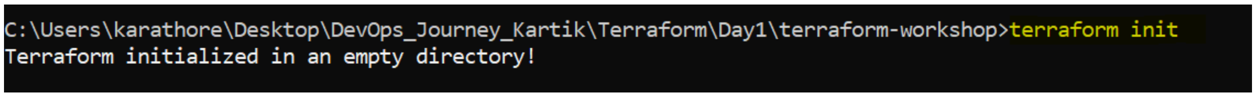
Steps:

1. **Create a Project Directory:**
   - Open your terminal and create a new directory for your project:

     mkdir terraform-workshop && cd terraform-workshop

2. **Initialize Terraform:**
   - Initialize Terraform in your project directory:

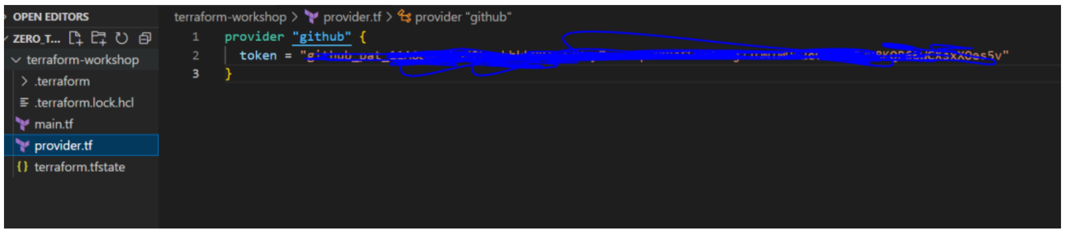
     terraform init



```
C:\Users\karathore\Desktop\DevOps_Journey_Kartik\Terraform\Day1\terraform-workshop>terraform init
Terraform initialized in an empty directory!
```

3. **Configure Terraform Provider (provider.tf):**
   - Create a file named *`provider.tf`* and add the following content, replacing `<YOUR_ACCESS_TOKEN>` with your actual GitHub personal access token:

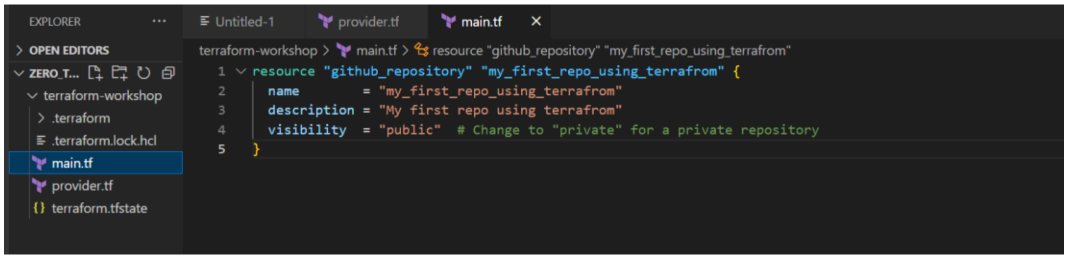
```
provider "github" {
  token = "<YOUR_ACCESS_TOKEN>"
}
```



```
provider "github" {
  token = [redacted]
}
```

4. **Define the GitHub Repository (main.tf):**
   - Create another file named `main.tf` and add the following content, replacing `<REPO_NAME>` and `<DESCRIPTION>` with your desired values:

```
resource "github_repository" "my_first_repo_using_terraform" {
  name        = "<REPO_NAME>"
  description = "<DESCRIPTION>"
  visibility  = "public"   # Change to "private" for a private repository
}
```



```
resource "github_repository" "my_first_repo_using_terrafrom" {
  name        = "my_first_repo_using_terrafrom"
  description = "My first repo using terrafrom"
  visibility  = "public"  # Change to "private" for a private repository
}
```

5. **Run Terraform Commands:**
   - **Validate:** `terraform validate` (checks for syntax errors)
   - **Preview:** `terraform plan` (shows the changes Terraform will make)
   - **Apply:** `terraform apply` (creates the GitHub repository)

Success! Head over to your GitHub account, and you'll see your newly created repository.

Project 2: Provisioning an EC2 Instance with Terraform

This project demonstrates launching an EC2 instance in the cloud using Terraform's infrastructure as code capabilities.

Prerequisites:

- **Terraform installed (same as Project 1):** Make sure Terraform is installed and configured.
- **AWS Account:** Sign up for a free AWS tier account if you don't have one already https://aws.amazon.com/free/
- **AWS Credentials:** Configure your AWS credentials (access key and secret access key) for Terraform to interact with AWS

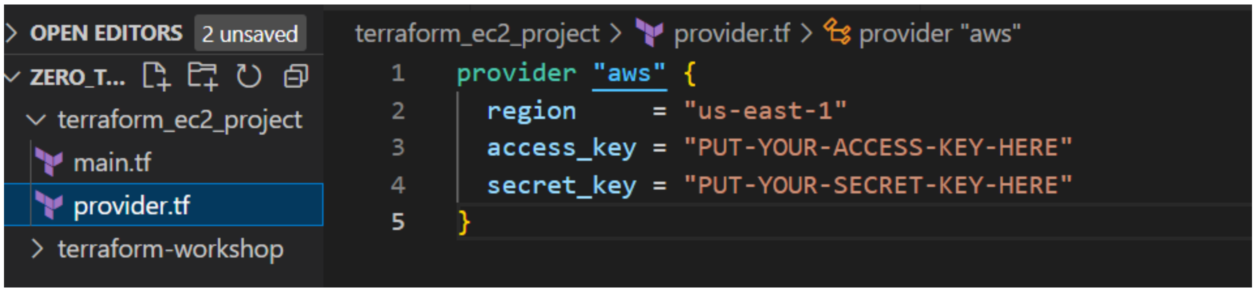
Steps:

1. **Create a Project Directory:**
   - Similar to Project 1, create a new directory for this project.
2. **Initialize Terraform:**
   - Initialize Terraform in your project directory:

     terraform init

3. **Configure Terraform Providers (provider.tf):**
   - Create a file named `provider.tf` and add the following content, replacing `<PUT-YOUR-ACCESS-KEY-HERE>` and `<PUT-YOUR-SECRET-KEY-HERE>` with your AWS credentials:

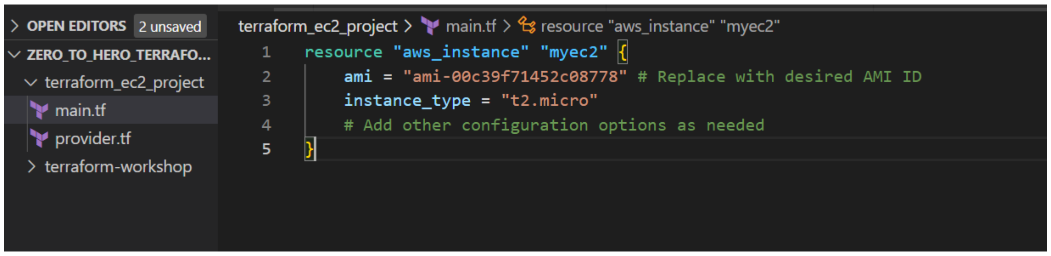
```
provider "aws" {
  region     = "us-east-1"
  access_key = "PUT-YOUR-ACCESS-KEY-HERE"
  secret_key = "PUT-YOUR-SECRET-KEY-HERE"
}
```



```
provider "aws" {
  region     = "us-east-1"
  access_key = "PUT-YOUR-ACCESS-KEY-HERE"
  secret_key = "PUT-YOUR-SECRET-KEY-HERE"
}
```

4. **Define the EC2 Instance (main.tf):**
   - Create a file named **`main.tf`** and add the following content to define a basic EC2 instance with an Amazon Linux 2 AMI:

```
resource "aws_instance" "myec2" {
    ami = "ami-00c39f71452c08778" # Replace with desired AMI ID
    instance_type = "t2.micro"
  # Add other configuration options as needed
}
```



```
resource "aws_instance" "myec2" {
    ami = "ami-00c39f71452c08778" # Replace with desired AMI ID
    instance_type = "t2.micro"
    # Add other configuration options as needed
}
```

1. **Run Terraform Commands:** (Same as Project 1)
   - **Validate:** `terraform validate`

- **Preview:** `terraform plan`
- **Apply:** `terraform apply` (creates the EC2 instance)

Important Note: Remember to terminate.