Terraform Course Outline

## Module 1: What is Terraform?

We go over what Terraform is and why it’s a highly sought-after skillset. We’ll also go over how to set up CloudShell so we can diveright into developing

* What is Terraform?
* How to setup your projects to follow along

## Module 2: Terraform Configurations

Learn the fundamentals of Terraform by creating a configuration file and deploying resources into Azure. We’ll also explore the most common ways to authenticate with Azure using the Azure Provider and discuss creating resource dependencies within the Terraform configuration.

* Terraform Configuration Files
* Authenticating with the Azure Provider
* Terraform Init, Plan, and Apply
* Using Interpolation within Terraform Configurations

## Module 3: Terraform State

Understanding how Terraform state works is a critical component for building Terraform solutions, which is why we’ll be going deep into how the state file keeps track of changes made with Terraform.

* Why does Terraform need to keep track of state
* Examine the inter-workings of the state file
* Storing the state file centrally using remote state
* Retrieve output from Terraform state

## Module 4: Variables

Variables allow Terraform modules to take in input making configuration files dynamic. Infrastructure code can be templatized and re-used throughout different environments.

* Create variable blocks in Terraform configurations
* Input values into Terraform configurations using several methods
* Learn about the common variable types

## Module 5: Terraform Modules

Terraform modules split up infrastructure into small, testable components that make it easy for infrastructure developers to introduce changes in small amounts. Learn how to easily create modules and use them with source control.

* Create a module
* Pass data between modules
* Learn how to use modules from GitHub
* Learn about the Terraform Registry

## Module 6: Advanced HCL

HCL is a functional language that allows infrastructure developers to create loops and conditional logic within Terraform code. This allows a high levels of abstraction to be created for deploying infrastructure.

* Functions, conditions, loops and expressions
* Dynamic blocks within Terraform resource blocks
* Creating resources that scale with count