**Lab Manual- Manage Azure App Service Using Terraform**

Table of Contents

[1 OBJECTIVE 2](#_Toc113463963)

[2 What is Terraform 3](#_Toc113463964)

[3 PRE-REQUISISTE 4](#_Toc113463965)

[4 Create Azure APP Service with Terraform 4](#_Toc113463966)

# OBJECTIVE

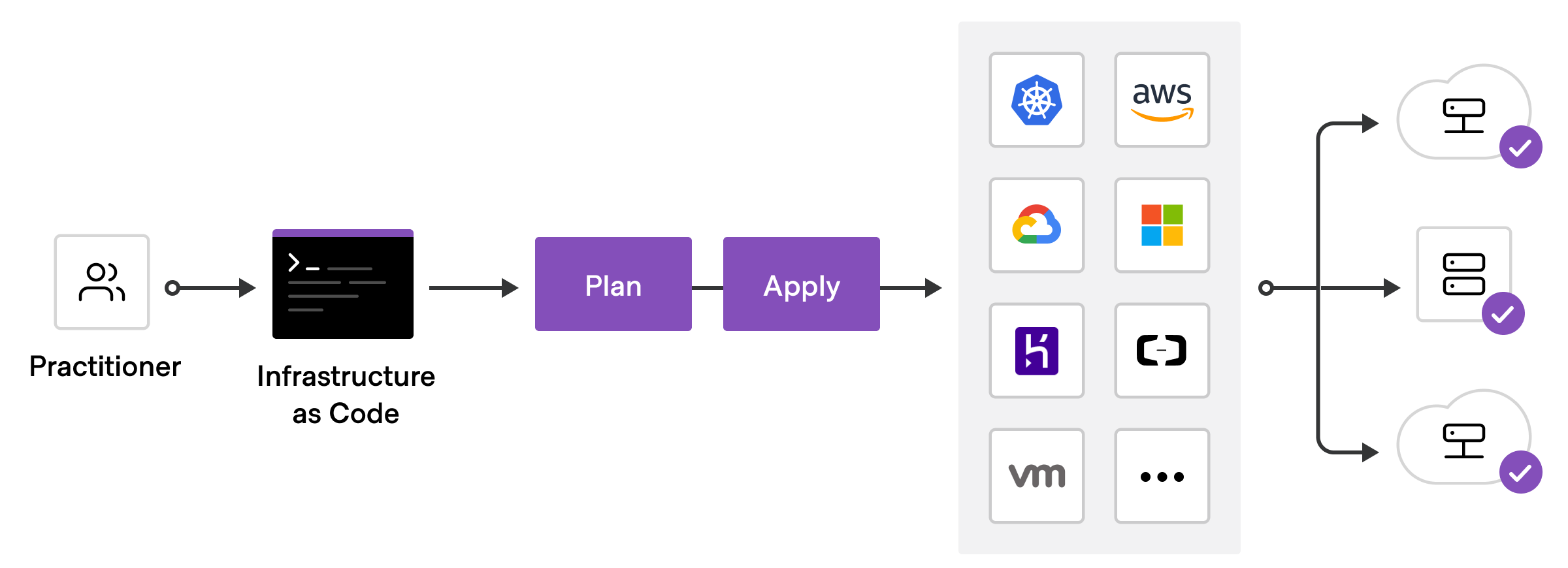
Terraform Azure providers enable you to manage all of your Azure infrastructure using the same declarative syntax and tooling. Using these providers you can:

* Provision core platform capabilities such as management groups, policies, users, groups, and policies. For more information, see [Terraform implementation of Cloud Adoption Framework Enterprise-scale](https://github.com/Azure/terraform-azurerm-caf-enterprise-scale#readme).
* Provision Azure DevOps Projects and pipelines to automate regular infrastructure and application deployments.
* Provision Azure resources required by your applications.

# What is Terraform

Terraform is HashiCorp's infrastructure as code tool. It lets you define resources and infrastructure in human-readable, declarative configuration files, and manages your infrastructure's lifecycle. Using Terraform has several advantages over manually managing your infrastructure:

* Terraform can manage infrastructure on multiple cloud platforms.
* The human-readable configuration language helps you write infrastructure code quickly.
* Terraform's state allows you to track resource changes throughout your deployments.
* You can commit your configurations to version control to safely collaborate on infrastructure.
* To deploy infrastructure with Terraform:
  + **Scope** - Identify the infrastructure for your project.
  + **Author** - Write the configuration for your infrastructure.
  + **Initialize** - Install the plugins Terraform needs to manage the infrastructure.
  + **Plan** - Preview the changes Terraform will make to match your configuration.
  + **Apply** - Make the planned changes.

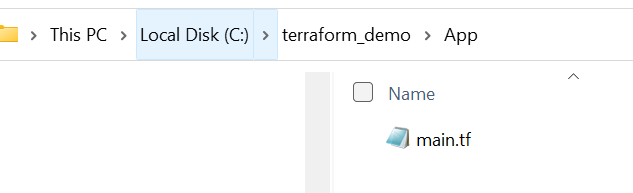


# PRE-REQUISISTE

* Accounts in Azure
* A local Computer with 4 CPU, 16 GB RAM, 200 GB disk space
* An Azure tenant and access to a subscription, like **Owner** or **Contributor** rights.
* VS Code or other IDE. However, VS Code has a [Terraform extension](https://docs.microsoft.com/en-us/azure/developer/terraform/configure-vs-code-extension-for-terraform) to improve the authoring process.
* Terraform open-source command-line interface
* Azure CLI ([download](https://docs.microsoft.com/en-us/cli/azure/install-azure-cli)). This tutorial uses version 2.32.0.

# Create Azure APP Service with Terraform

1. Create Another Folder and Create **Main.tf** file inside it.



1. Update the Content of the Main.tf with below code

terraform {

required\_providers {

azurerm = {

source = "hashicorp/azurerm"

version = "~>2.0"

}

}

}

provider "azurerm" {

features {}

subscription\_id = "c49a4614-f368-4f5b-b72a-88bf82d12229"

tenant\_id = "be04fbd5-6b00-412c-a86c-ca105b5cce90"

client\_id = "0b381472-3197-49d4-a324-f1a96a23c8a7"

client\_secret = "QYd8Q~0fCdFMQW\_AtMZoNaiXHlGSbdGOq1fbTcui"

}

resource "azurerm\_resource\_group" "example" {

name = "example-RG864"

location = "eastus"

}

resource "azurerm\_app\_service\_plan" "example" {

name = "ctrlsplan-appserviceplan"

location = azurerm\_resource\_group.example.location

resource\_group\_name = azurerm\_resource\_group.example.name

sku {

tier = "Standard"

size = "S2"

}

}

resource "azurerm\_app\_service" "example" {

name = "ctrl67y8777kr4"

location = azurerm\_resource\_group.example.location

resource\_group\_name = azurerm\_resource\_group.example.name

app\_service\_plan\_id = azurerm\_app\_service\_plan.example.id

site\_config {

dotnet\_framework\_version = "v4.0"

scm\_type = "LocalGit"

}

app\_settings = {

"SOME\_KEY" = "some-value"

}

connection\_string {

name = "Database"

type = "SQLServer"

value = "Server=some-server.mydomain.com;Integrated Security=SSPI"

}

}

1. Run [terraform init](https://www.terraform.io/docs/commands/init.html) to initialize the Terraform deployment. This command downloads the Azure modules required to manage your Azure resources.

**terraform init**

Text

Description automatically generated

1. Run [terraform plan](https://www.terraform.io/docs/commands/plan.html) to create an execution plan.

**terraform plan**

Text

Description automatically generated

1. Run [terraform apply](https://www.terraform.io/docs/commands/apply.html) to apply the execution plan to your cloud infrastructure.

**terraform Apply**

Text

Description automatically generated

Text

Description automatically generated

1. Go to Azuree Portal and Verify

Graphical user interface, text, application

Description automatically generated