LIVE STREAMING

Microsoft® MVP Most Valuable Professional

Terraform Associate Exam Preparation (1)

Wednesday February 24th, 2021
18:00 to 20:00 GMT

Cloud DevOps Visions
PLAN CODE BUILD TEST PACKAGE DEPLOY CONFIGURE MONITOR
www.clouddevopsvisions.com

Mohamed Radwan
Principal DevOps Consultant
mohamedradwan.com

Microsoft Most Valuable Professional

Mohamed Radwan

Principal DevOps

Principal Cloud DevOps Consultant

With 18+ Years of Experience, Helped 50+ companies around the globe

Blog: mohamedradwan.com
Twitter: @mradwan06

Agenda

Agenda

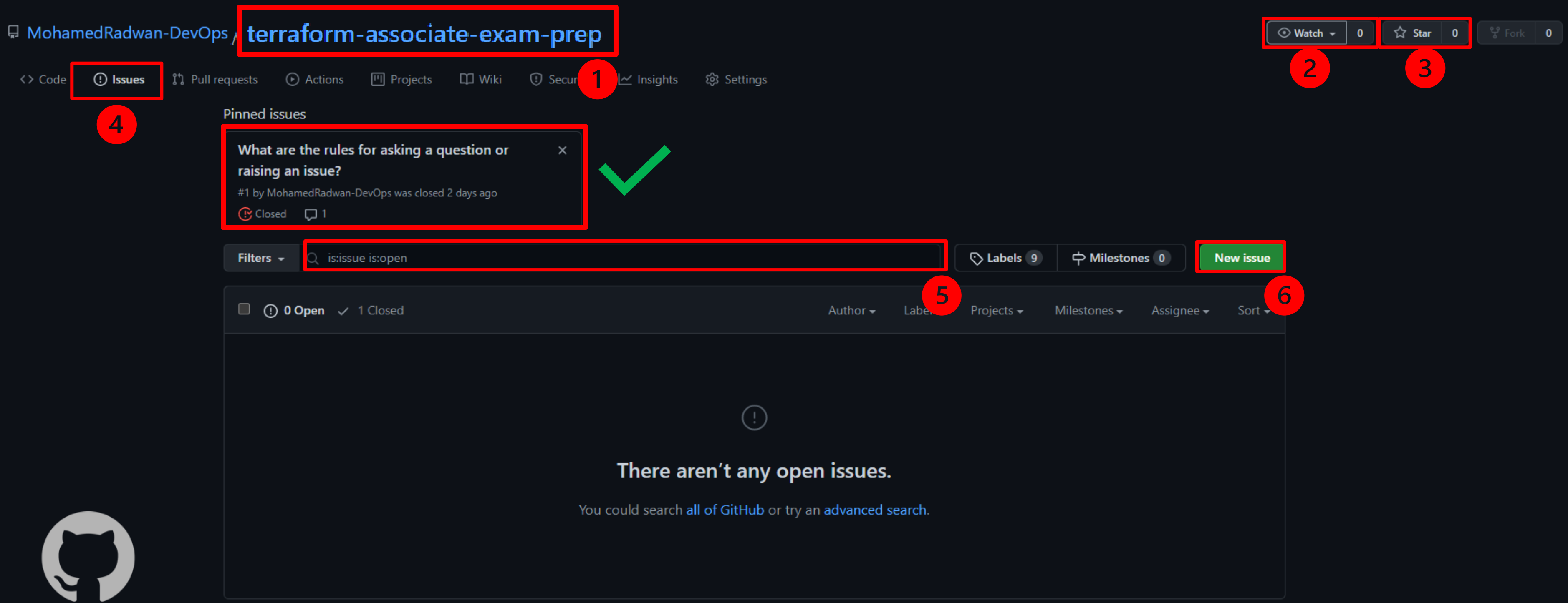
- Where to ask questions and find resources for this series (GitHub repo)
- Exam objectives
- Overview of The End-to-End Demo
- Exam Preparation Plan
- Understanding Infrastructure as Code (IaC) concepts.
- Introduction to Infrastructure as Code with Terraform
- Install Terraform (demo)
- Build infrastructure with Terraform (demo)
- Change Infrastructure with Terraform (demo)
- Destroy Infrastructure with Terraform (demo)

Continue on (P2)

Agenda (2)

- Create Resource Dependencies (demo)
- Define input variables (demo)
- Query data with output variables (demo)
- Store Remote State (demo)
- Terraform (fmt and validate) (demo)
- Taint resources (demo)
- Notes

Record your issues/questions on GitHub


MohamedRadwan-DevOps / terraform-associate-exam-prep

Pinned issues

What are the rules for asking a question or raising an issue?

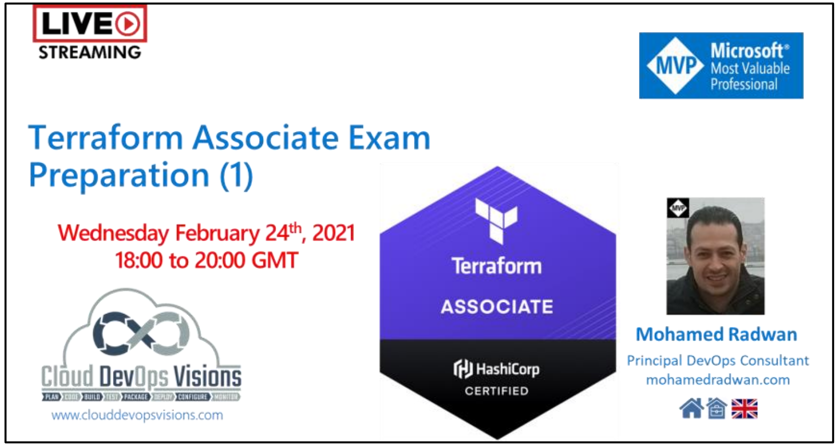
#1 by MohamedRadwan-DevOps was closed 2 days ago

Closed 1

There aren't any open issues.

You could search all of GitHub or try an advanced search.

Feedback is very important to continue


LIVE STREAMING

Microsoft® Most Valuable Professional MVP

Terraform Associate Exam Preparation (1)

Wednesday February 24th, 2021
18:00 to 20:00 GMT

Cloud DevOps Visions
www.clouddevopsvisions.com

Terraform ASSOCIATE HashiCorp CERTIFIED

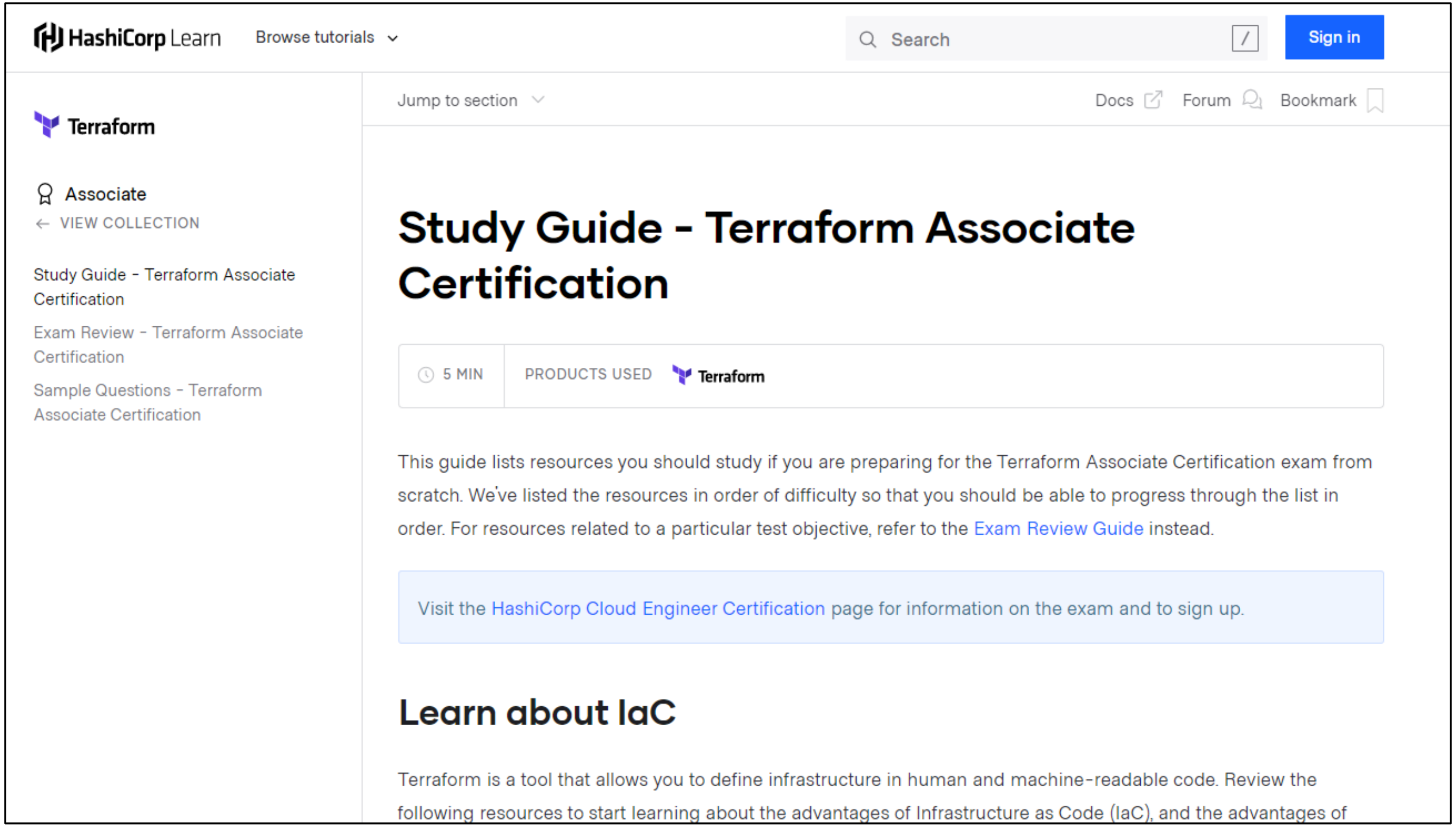
Mohamed Radwan
Principal DevOps Consultant
mohamedradwan.com

https://youtu.be/og76ViVI4ow

Feedback and questions

FEEDBACK

Main source HashiCorp Learn



Study Guide - Terraform Associate Certification

Exam Review - Terraform Associate Certification

Sample Questions - Terraform Associate Certification

Study Guide - Terraform Associate Certification

5 MIN | PRODUCTS USED Terraform

This guide lists resources you should study if you are preparing for the Terraform Associate Certification exam from scratch. We've listed the resources in order of difficulty so that you should be able to progress through the list in order. For resources related to a particular test objective, refer to the Exam Review Guide instead.

Visit the HashiCorp Cloud Engineer Certification page for information on the exam and to sign up.

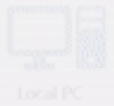
Learn about IaC

Terraform is a tool that allows you to define infrastructure in human and machine-readable code. Review the following resources to start learning about the advantages of Infrastructure as Code (IaC), and the advantages of

Exam Objectives

1. Understanding Infrastructure as Code (IaC) concepts.
2. Understanding Terraform's purpose (vs other IaC).
3. Understanding Terraform basics.
4. Demonstrating that you can use the Terraform CLI (outside of core workflow).
5. Demonstrating that you can interact with Terraform modules.
6. Demonstrating that you can navigate the Terraform workflow.
7. Demonstrating that you can implement and maintain state.
8. Demonstrating that you can read, generate, and modify Terraform configuration.
9. Demonstrating that you understand Terraform Cloud and Enterprise capabilities.

Overview of The End-to-End Demo



Exam Preparation Plan

Day 1 (Session 1)
3 objectives

1. Infrastructure as Code (IaC).
2. Terraform vs other IaC.
3. Terraform basics.

Day 2 (Session 2)
3 objectives

4. Terraform CLI outside the workflow.
5. Terraform modules.
6. Terraform workflow.

Day 3 (Session 3)
3 objectives

7. Implement and maintain state.
8. Terraform configuration.
9. Terraform Cloud & Enterprise capabilities.

- **End-to-End Demo**

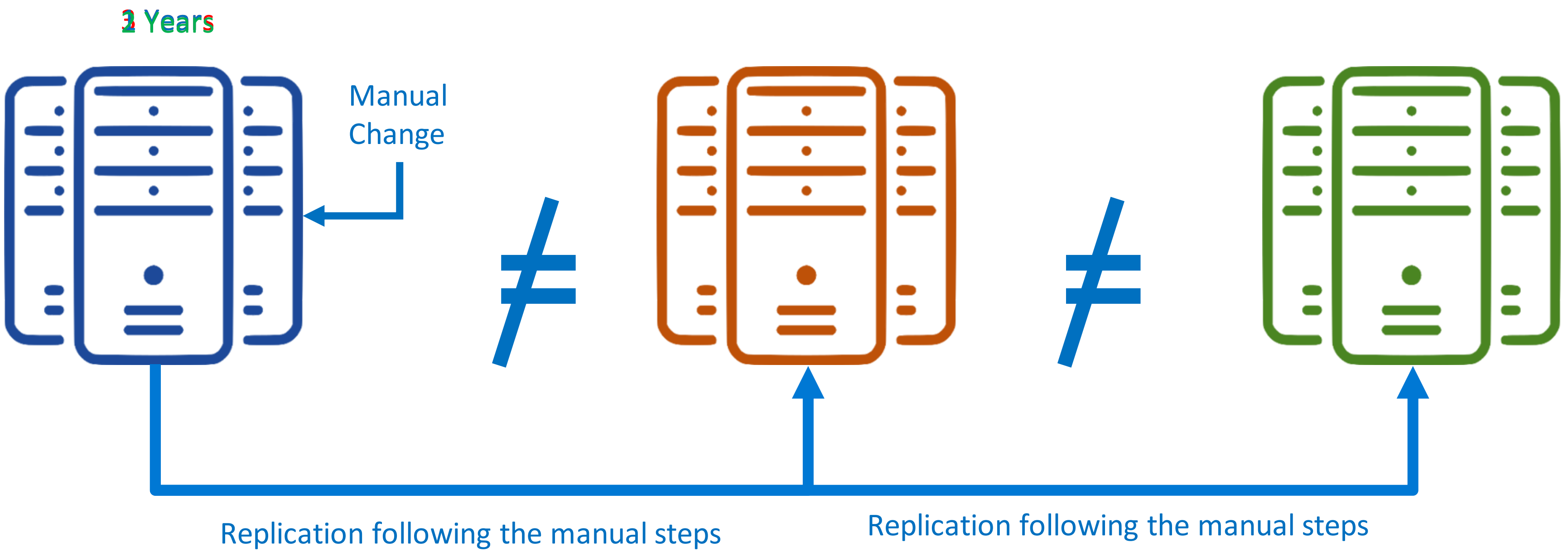
1 - Understanding Infrastructure as Code (IaC) concepts.

A. Explain what IaC is

B. Describe advantages of IaC patterns

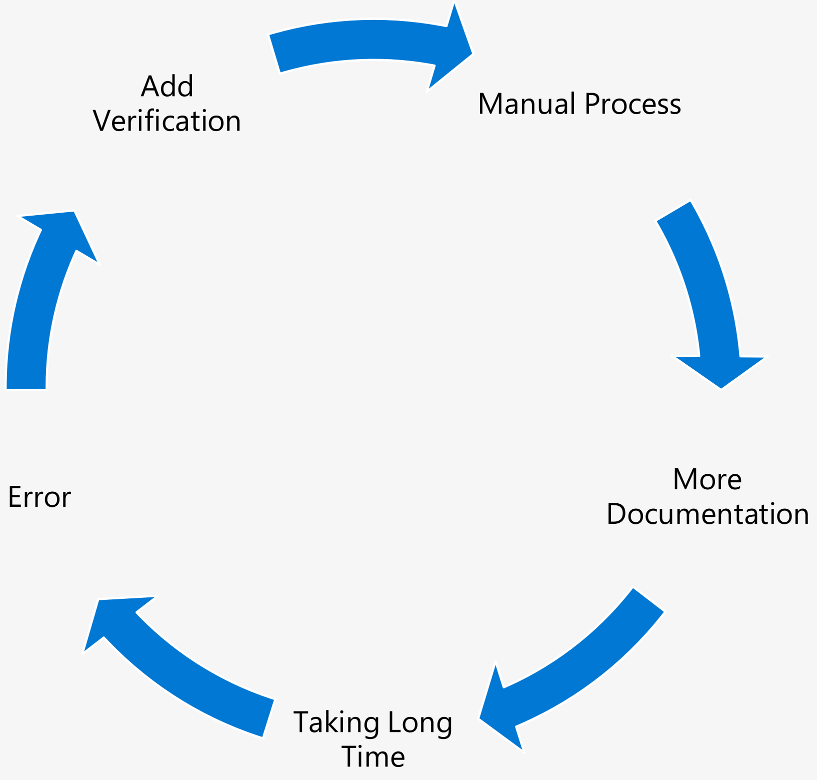
A. Explain what IaC is

Challenges With Traditional Infrastructure


1 Years
Manual Change
≠
≠
Replication following the manual steps
Replication following the manual steps

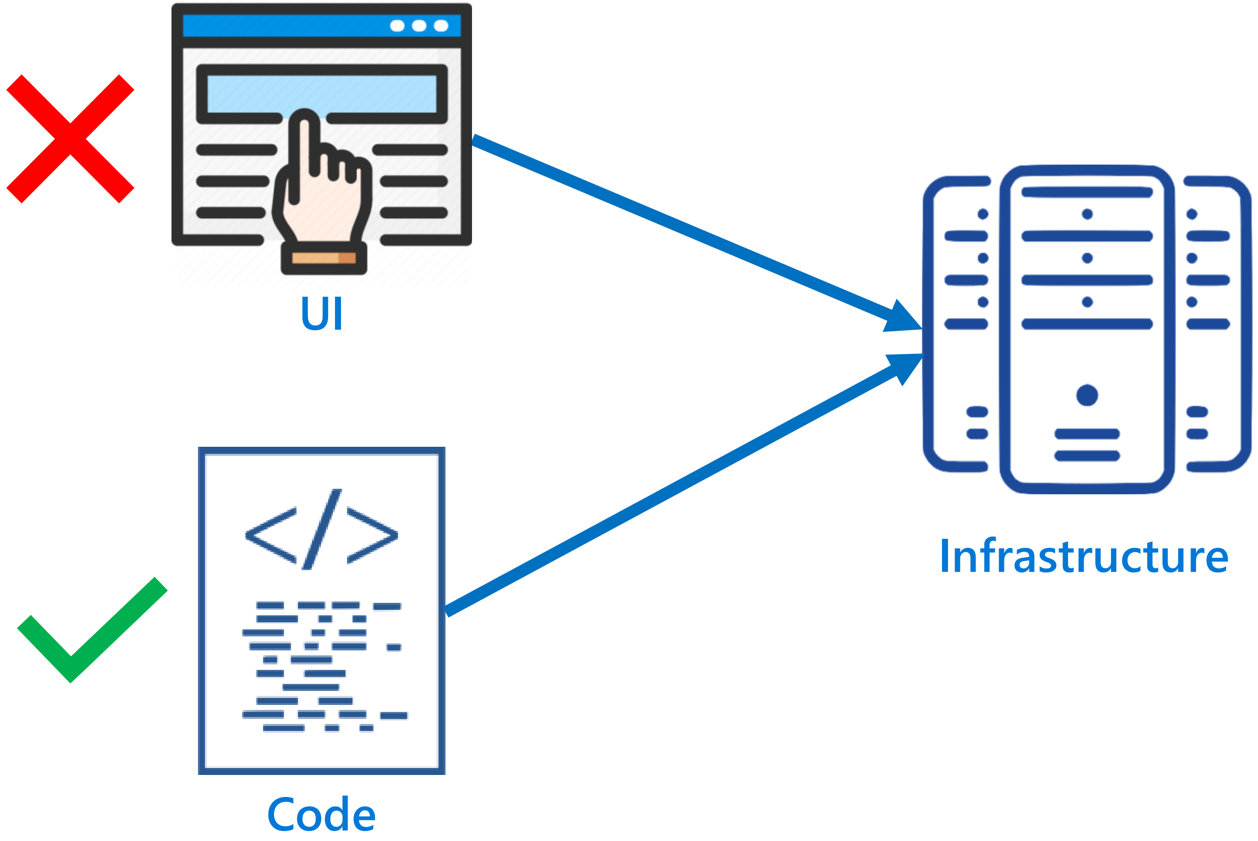
Scale up required downtime

Vicious Circle of Manual Process



Pets vs. Cattle

Infrastructure & Configuration as Code



Declarative vs Imperative Infrastructure as Code

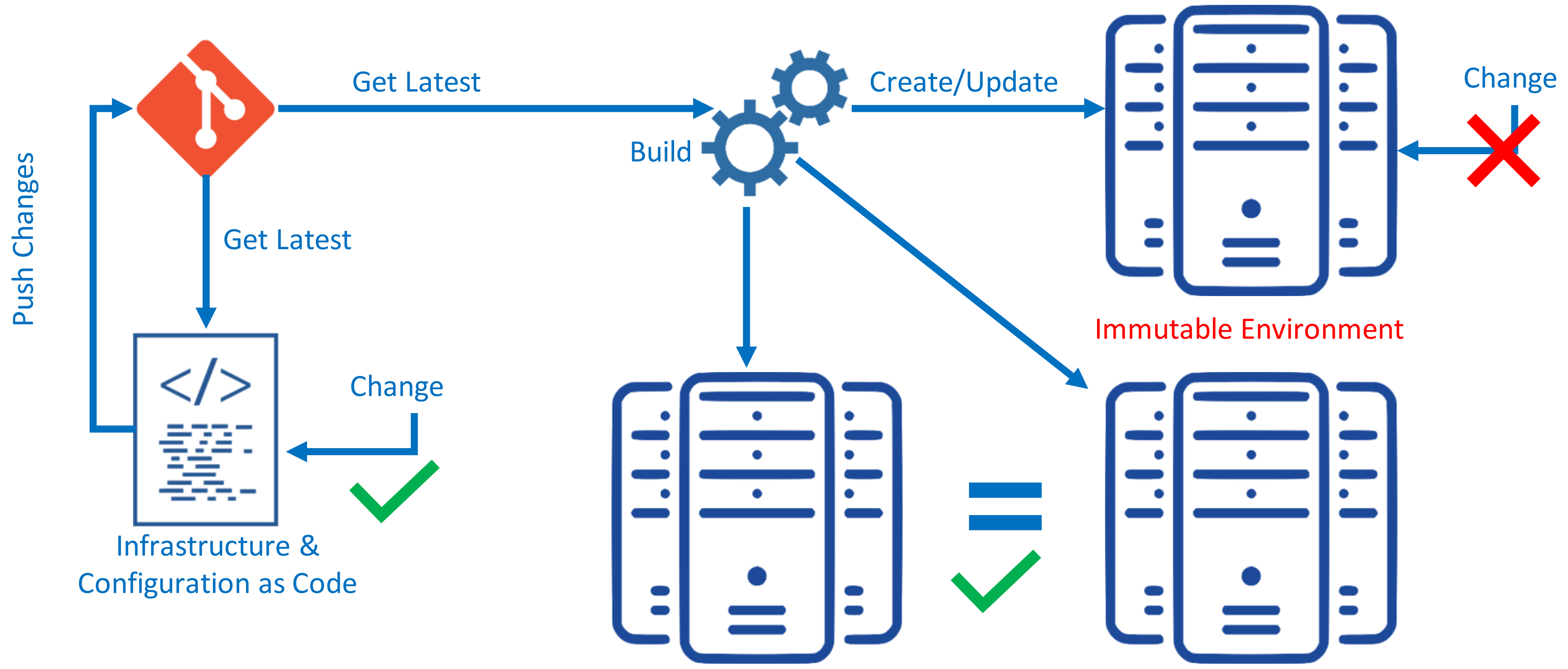
Imperative

- What and how
- Workflow
- Procedure
- Reach final state by executing ordered steps.

Declarative

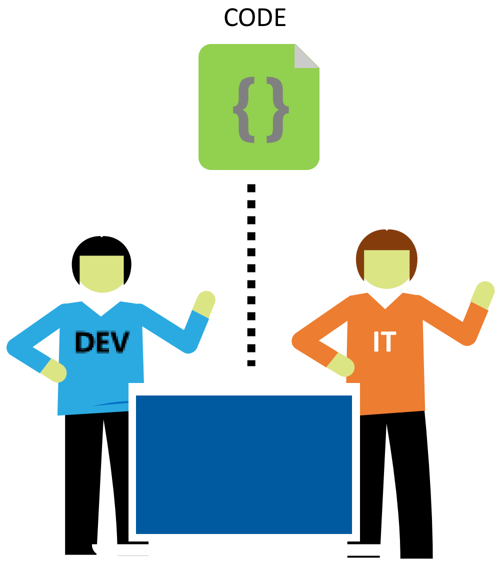
- What not how
- Logic no workflow
- Functional
- Reach final state by executing the content with no order

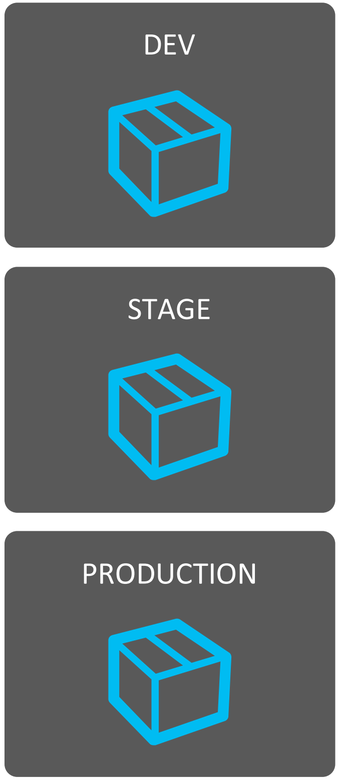
Infrastructure as Code workflow



B. Describe advantages of IaC patterns

Infrastructure & Configuration as Code





Value

- Deployment Optimization
- Accelerate Delivery

Measure

- Deployment Rate
- MTTR

Advantages of Infrastructure as Code

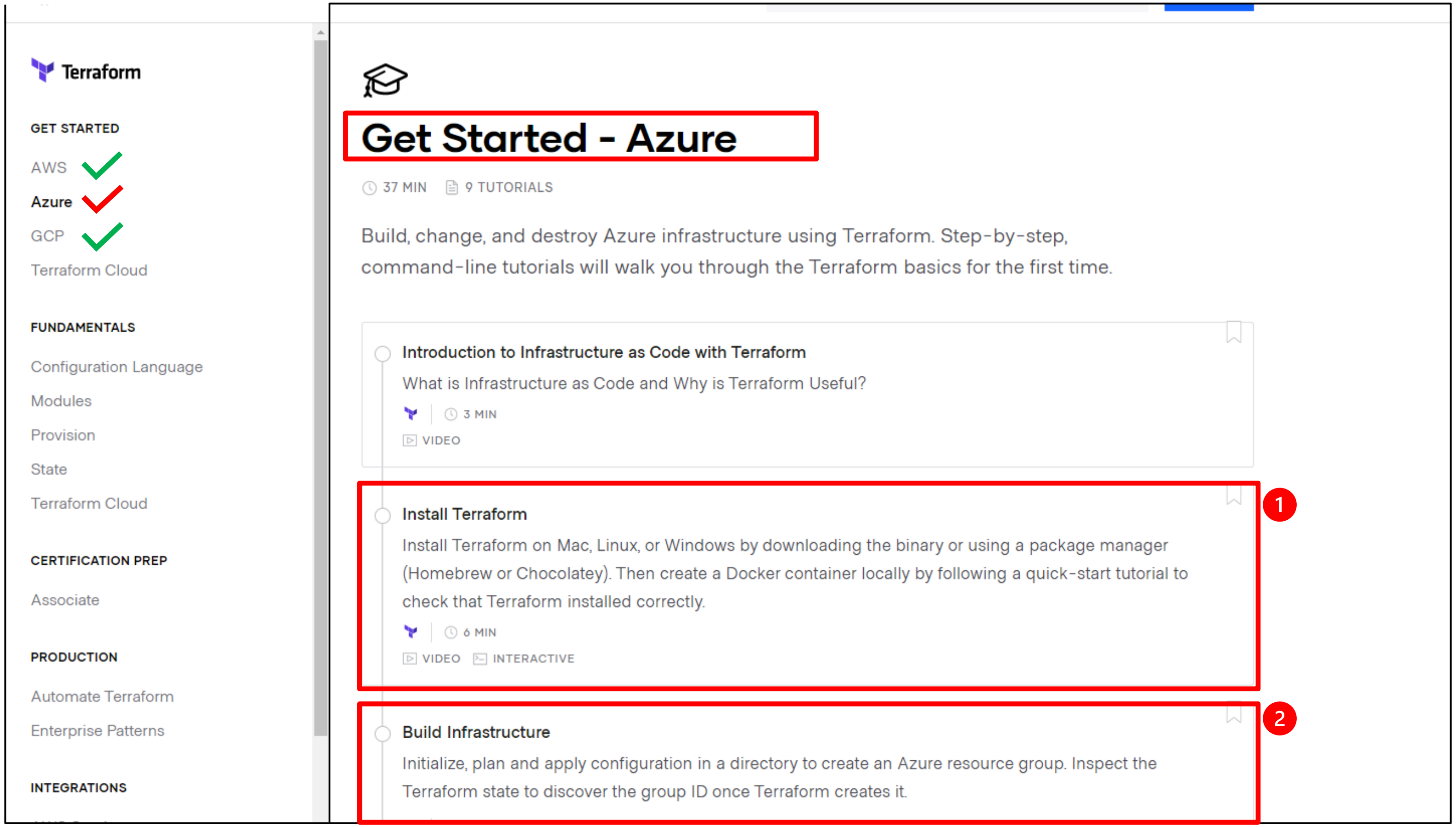
- Easily Repeatable
- Easily Readable
- Operational certainty with “terraform plan”
- Standardized environments builds
- Quickly provisioned development environments
- Disaster Recovery

Advantages of Terraform

- Platform Agnostic
- State Management
- Operator Confidence

Introduction to Infrastructure as Code with Terraform

HashiCorp Learn



Terraform

GET STARTED

AWS ✓

Azure ✓

GCP ✓

Terraform Cloud

FUNDAMENTALS

Configuration Language

Modules

Provision

State

Terraform Cloud

CERTIFICATION PREP

Associate

PRODUCTION

Automate Terraform

Enterprise Patterns

INTEGRATIONS

Get Started - Azure

37 MIN 9 TUTORIALS

Build, change, and destroy Azure infrastructure using Terraform. Step-by-step, command-line tutorials will walk you through the Terraform basics for the first time.

Introduction to Infrastructure as Code with Terraform

What is Infrastructure as Code and Why is Terraform Useful?

3 MIN

VIDEO

1

Install Terraform

Install Terraform on Mac, Linux, or Windows by downloading the binary or using a package manager (Homebrew or Chocolatey). Then create a Docker container locally by following a quick-start tutorial to check that Terraform installed correctly.

6 MIN

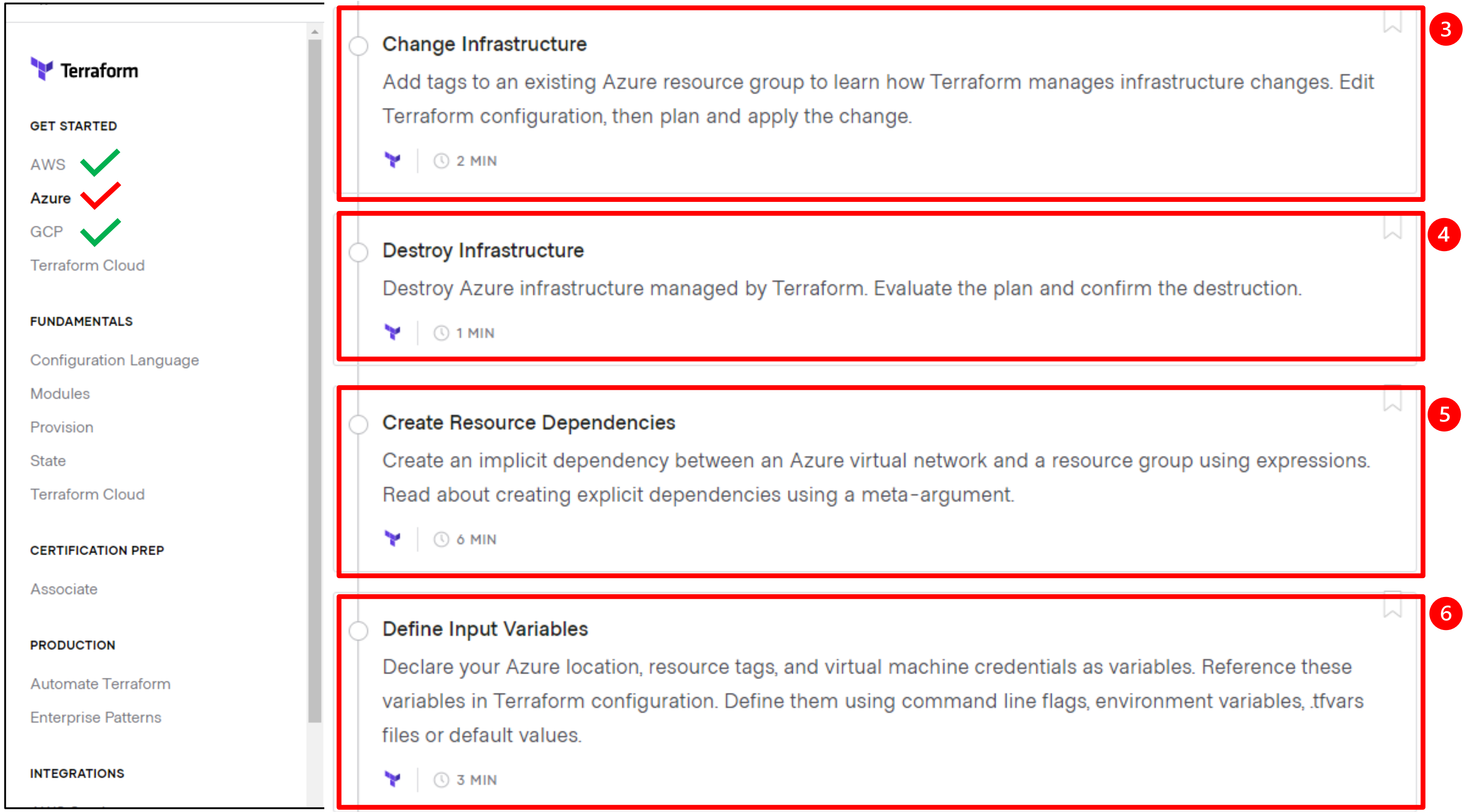
VIDEO INTERACTIVE

2

Build Infrastructure

Initialize, plan and apply configuration in a directory to create an Azure resource group. Inspect the Terraform state to discover the group ID once Terraform creates it.

HashiCorp Learn



Terraform

GET STARTED

AWS ✓

Azure ✓

GCP ✓

Terraform Cloud

FUNDAMENTALS

Configuration Language

Modules

Provision

State

Terraform Cloud

CERTIFICATION PREP

Associate

PRODUCTION

Automate Terraform

Enterprise Patterns

INTEGRATIONS

3

Change Infrastructure

Add tags to an existing Azure resource group to learn how Terraform manages infrastructure changes. Edit Terraform configuration, then plan and apply the change.

2 MIN

4

Destroy Infrastructure

Destroy Azure infrastructure managed by Terraform. Evaluate the plan and confirm the destruction.

1 MIN

5

Create Resource Dependencies

Create an implicit dependency between an Azure virtual network and a resource group using expressions. Read about creating explicit dependencies using a meta-argument.

6 MIN

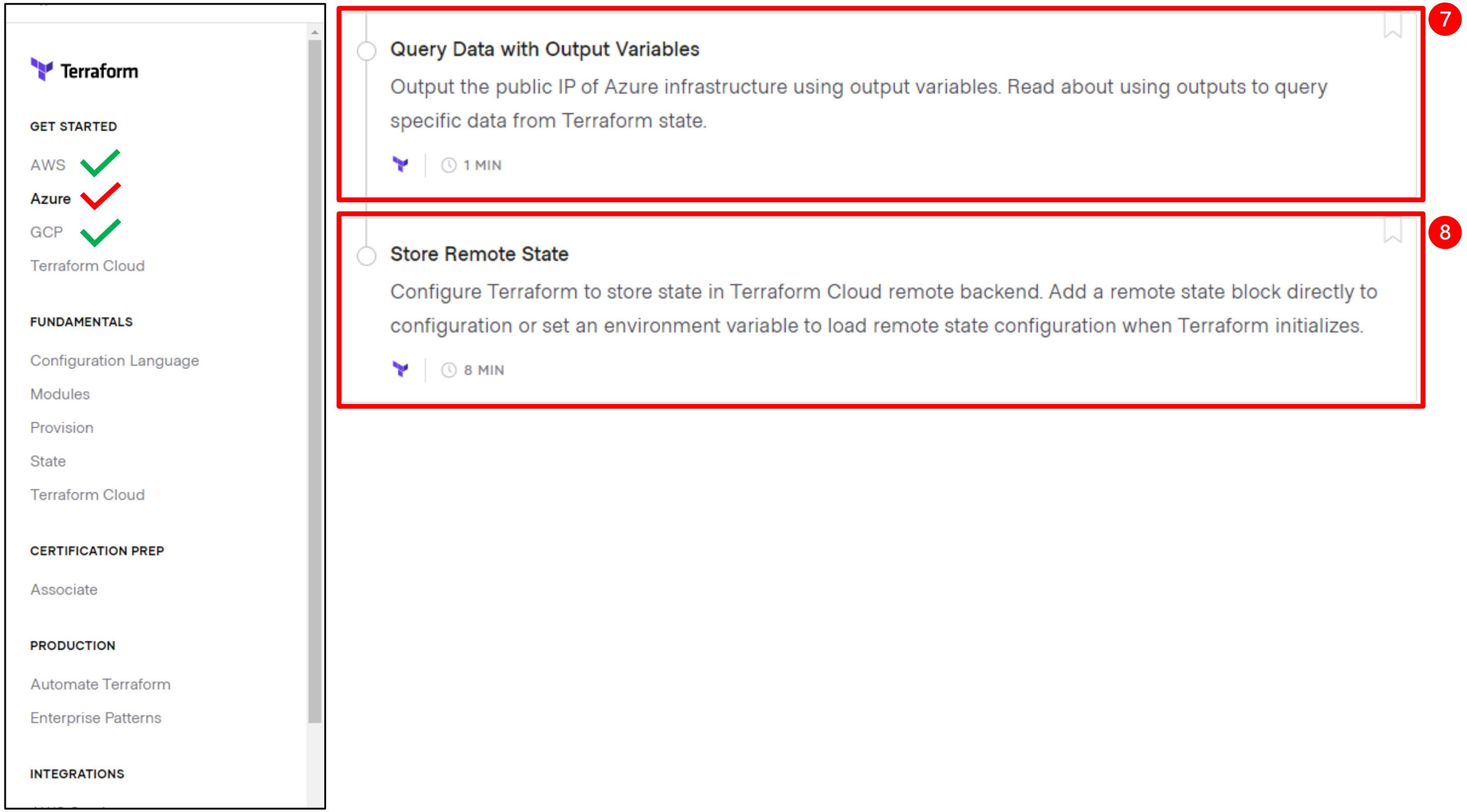
6

Define Input Variables

Declare your Azure location, resource tags, and virtual machine credentials as variables. Reference these variables in Terraform configuration. Define them using command line flags, environment variables, .tfvars files or default values.

3 MIN

HashiCorp Learn



Terraform

GET STARTED

AWS ✓

Azure ✓

GCP ✓

Terraform Cloud

FUNDAMENTALS

Configuration Language

Modules

Provision

State

Terraform Cloud

CERTIFICATION PREP

Associate

PRODUCTION

Automate Terraform

Enterprise Patterns

INTEGRATIONS

7

Query Data with Output Variables

Output the public IP of Azure infrastructure using output variables. Read about using outputs to query specific data from Terraform state.

1 MIN

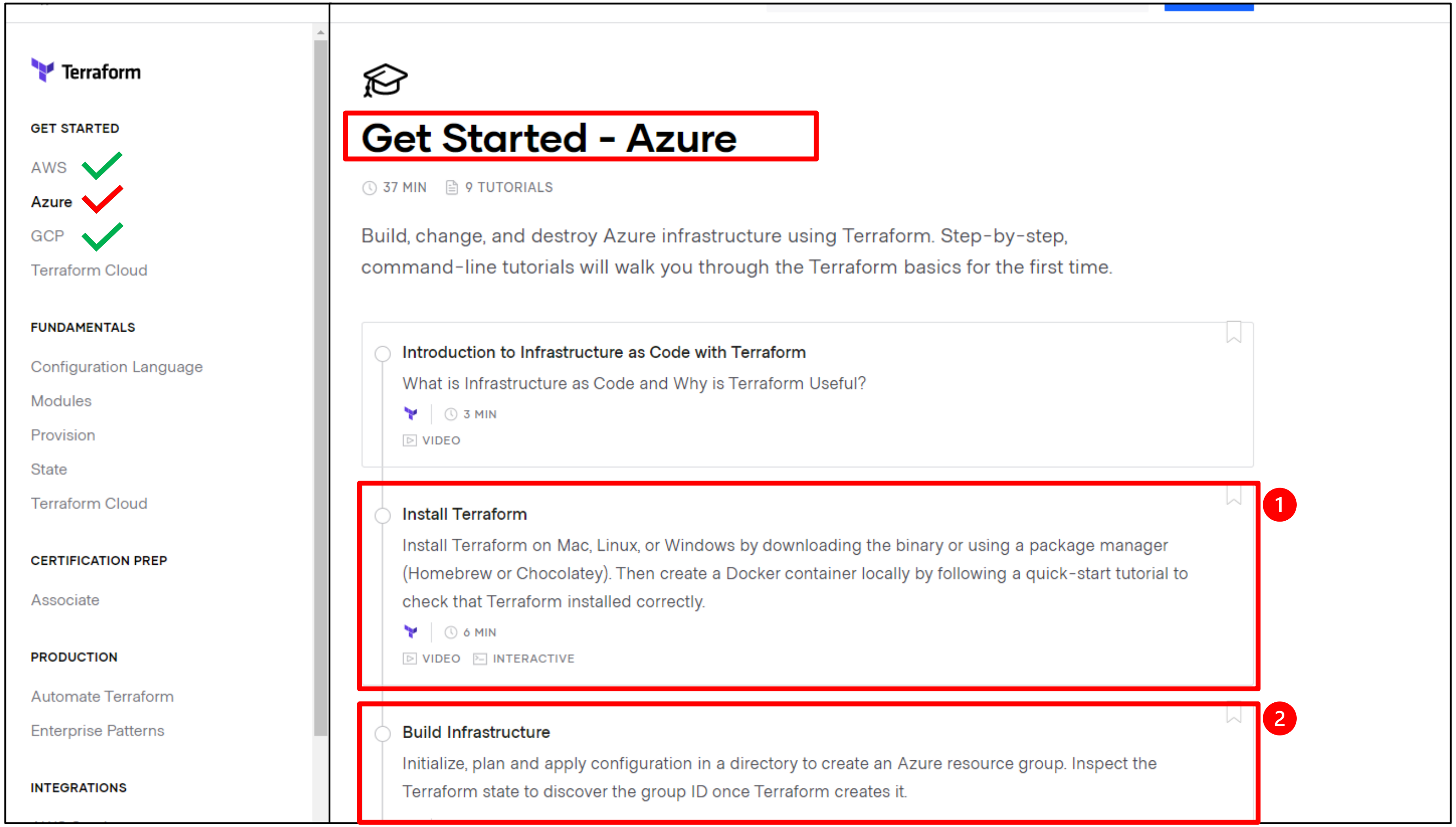
8

Store Remote State

Configure Terraform to store state in Terraform Cloud remote backend. Add a remote state block directly to configuration or set an environment variable to load remote state configuration when Terraform initializes.

8 MIN

HashiCorp Learn



Terraform

GET STARTED

AWS ✓

Azure ✓

GCP ✓

Terraform Cloud

FUNDAMENTALS

Configuration Language

Modules

Provision

State

Terraform Cloud

CERTIFICATION PREP

Associate

PRODUCTION

Automate Terraform

Enterprise Patterns

INTEGRATIONS

Get Started - Azure

37 MIN 9 TUTORIALS

Build, change, and destroy Azure infrastructure using Terraform. Step-by-step, command-line tutorials will walk you through the Terraform basics for the first time.

Introduction to Infrastructure as Code with Terraform

What is Infrastructure as Code and Why is Terraform Useful?

3 MIN

VIDEO

Install Terraform 1

Install Terraform on Mac, Linux, or Windows by downloading the binary or using a package manager (Homebrew or Chocolatey). Then create a Docker container locally by following a quick-start tutorial to check that Terraform installed correctly.

6 MIN

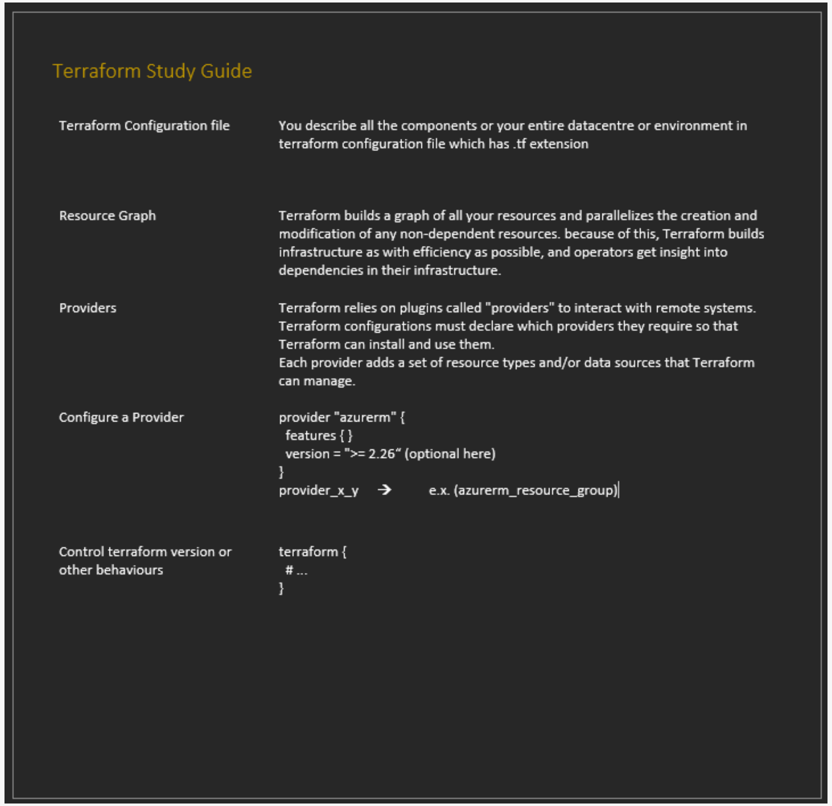
VIDEO INTERACTIVE

Build Infrastructure 2

Initialize, plan and apply configuration in a directory to create an Azure resource group. Inspect the Terraform state to discover the group ID once Terraform creates it.

Take-aways Knowledge

All resources will be on GitHub and YouTube



Terraform Study Guide

Terraform Configuration file	You describe all the components or your entire datacentre or environment in terraform configuration file which has .tf extension
Resource Graph	Terraform builds a graph of all your resources and parallelizes the creation and modification of any non-dependent resources. because of this, Terraform builds infrastructure as with efficiency as possible, and operators get insight into dependencies in their infrastructure.
Providers	Terraform relies on plugins called "providers" to interact with remote systems. Terraform configurations must declare which providers they require so that Terraform can install and use them. Each provider adds a set of resource types and/or data sources that Terraform can manage.
Configure a Provider	provider "azurerm" { features { } version = ">= 2.26" (optional here) } provider_x_y ➔ e.x. (azurerm_resource_group)
Control terraform version or other behaviours	terraform { # ... }

mohamedradwan.com

Terraform Configuration file

File with.tf extension

Resource Graph

- Parallelizes the creation
- Modification of any non-dependent resources.
- Efficiency as possible

Providers

- Plugins called "providers"
- Must declare which providers
- Each provider adds a set of resource

Configure a Provider

```
provider "azurerm" {
  features {}
  version = ">= 2.26“ (optional here)
}
provider_x_y        →      e.x. (azurerm_resource_group)
```

Control terraform version or other behaviours

```
terraform {
  # …
}
```

Constrain a provider version

```
terraform {
  required_providers {
    azurerm  = {
      source = "hashicorp/azurerm"
      version = ">= 2.26"
    }  } }
```

Alias

Distinguish several providers
Use desired provider using this syntax inside the resource block {}
Provider_value.alias_value
e.g. provider → aws.east

Alias

```
# The default provider configuration
provider "aws" {
  region = " us-west-2"
}

# Additional provider configuration for east region
provider "aws" {
  alias  = “east"
  region = " us-east-1"
}
```

e.g aws.east

Terraform init (cmd)

- Downloads plugin(s)
- Re-running init
- Can’t install third-party plugins

Upgrade provider

terraform init --upgrade

Terraform.lock.hcl (file)

1st time init will generate a new .terraform.lock.hcl
Include the lock file in your version control
Belongs to the configuration as a whole
Automatically creates or updates with init

Terraform plan (cmd)

Generate an execution plan

Terraform apply (cmd)

- Apply the configuration in the execution plan.
- Output shows the execution plan
- I can specify which using -out switch

Terraform.tfstate (file)

- It has the IDs and properties of the resources real-world infrastructure .
- Sensitive values in plaintext
- Meta-data and stores them in this file
- For teams or larger projects, state remotely.

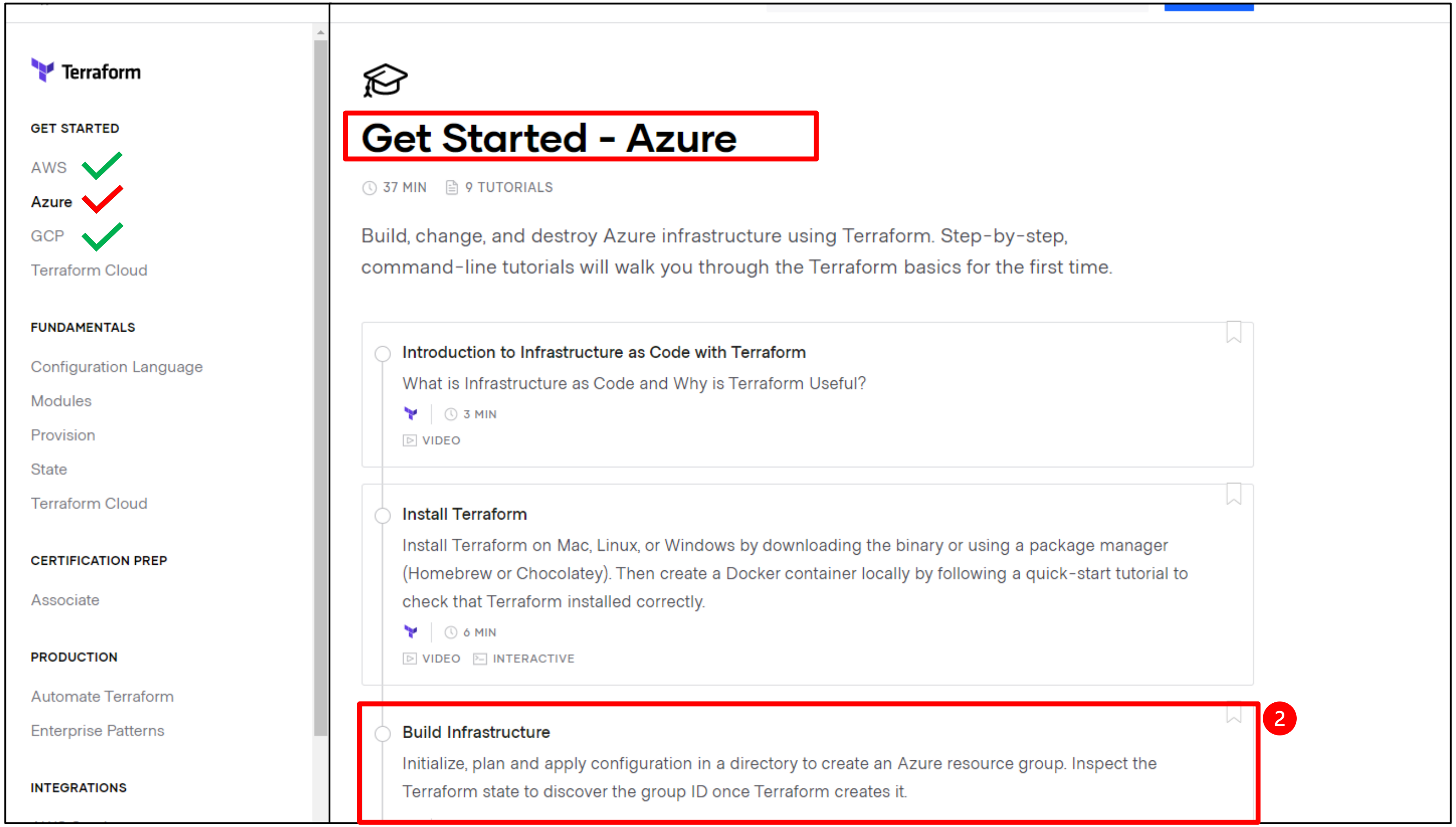
Terraform show (cmd)

Inspect or show all the current state which exists in the state file

Terraform state (cmd)

- Slice and dice the state file
- Many switches

HashiCorp Learn



Terraform

GET STARTED

AWS ✓

Azure ✓

GCP ✓

Terraform Cloud

FUNDAMENTALS

Configuration Language

Modules

Provision

State

Terraform Cloud

CERTIFICATION PREP

Associate

PRODUCTION

Automate Terraform

Enterprise Patterns

INTEGRATIONS

Get Started - Azure

37 MIN · 9 TUTORIALS

Build, change, and destroy Azure infrastructure using Terraform. Step-by-step, command-line tutorials will walk you through the Terraform basics for the first time.

Introduction to Infrastructure as Code with Terraform

What is Infrastructure as Code and Why is Terraform Useful?

3 MIN

VIDEO

Install Terraform

Install Terraform on Mac, Linux, or Windows by downloading the binary or using a package manager (Homebrew or Chocolatey). Then create a Docker container locally by following a quick-start tutorial to check that Terraform installed correctly.

6 MIN

VIDEO · INTERACTIVE

Build Infrastructure

Initialize, plan and apply configuration in a directory to create an Azure resource group. Inspect the Terraform state to discover the group ID once Terraform creates it.

2

Take-aways Knowledge

Terraform refresh

- Reconcile the state
- Via its state file
- Detect any drift from the last-known state

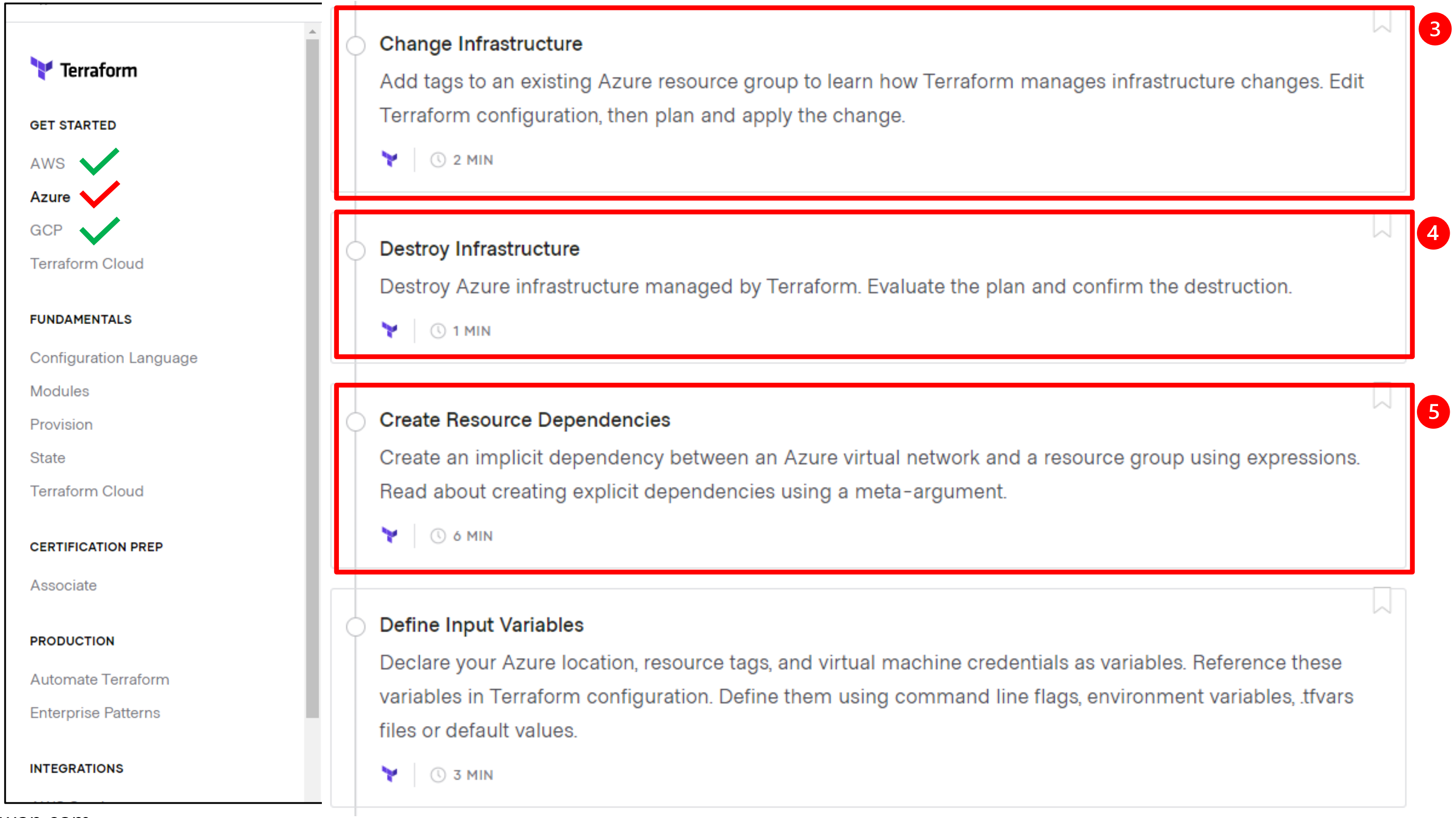
Terraform plan -out=<myplan> (cmd)

Ensures your terraform apply operation runs the exact plan

Symbols

Symbol	Meaning
~ (normal colour)	It means change resource
+ (green colour)	It means add resource
- (red colour)	It means destroy resource
-/+ (red &green colour)	It means replace resource as it was tainted

HashiCorp Learn



Terraform

GET STARTED

AWS ✓

Azure ✓

GCP ✓

Terraform Cloud

FUNDAMENTALS

Configuration Language

Modules

Provision

State

Terraform Cloud

CERTIFICATION PREP

Associate

PRODUCTION

Automate Terraform

Enterprise Patterns

INTEGRATIONS

Change Infrastructure 3

Add tags to an existing Azure resource group to learn how Terraform manages infrastructure changes. Edit Terraform configuration, then plan and apply the change.

2 MIN

Destroy Infrastructure 4

Destroy Azure infrastructure managed by Terraform. Evaluate the plan and confirm the destruction.

1 MIN

Create Resource Dependencies 5

Create an implicit dependency between an Azure virtual network and a resource group using expressions. Read about creating explicit dependencies using a meta-argument.

6 MIN

Define Input Variables

Declare your Azure location, resource tags, and virtual machine credentials as variables. Reference these variables in Terraform configuration. Define them using command line flags, environment variables, .tfvars files or default values.

3 MIN

Take-aways Knowledge

Terraform destroy

- Remove infrastructure, as with apply
- Waits for approval

Value [""]

- Define lists.
- Argument accepts more than one value.

Create the required resources for VM

Azure requires the following underlying resources before you can deploy a virtual machine:

- Resource group
- Virtual network
- Subnet
- Network security group
- Network interface
- Public IP address as well as an explicitly opened port 22 for SSH access.

Resource Dependencies

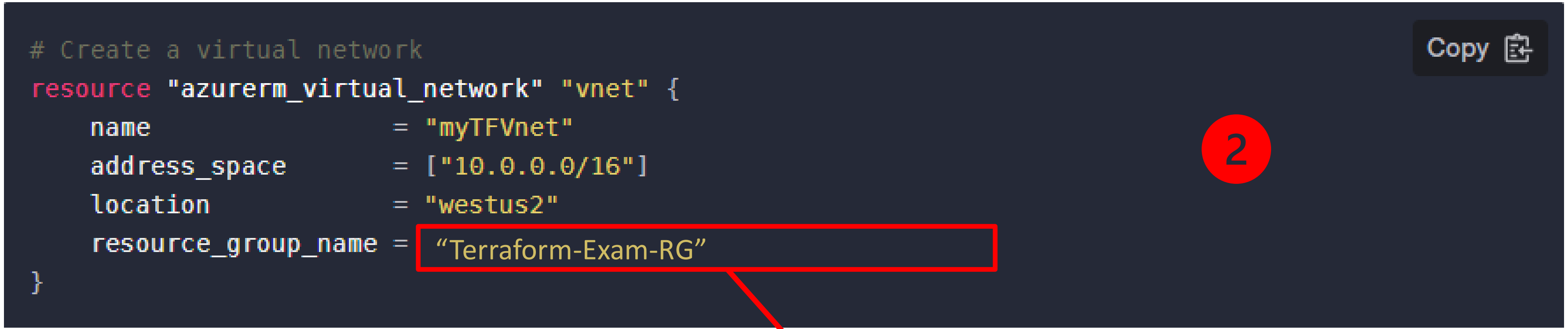
Implicit dependencies:

Determine automatically based on the configuration.

Explicit dependencies:

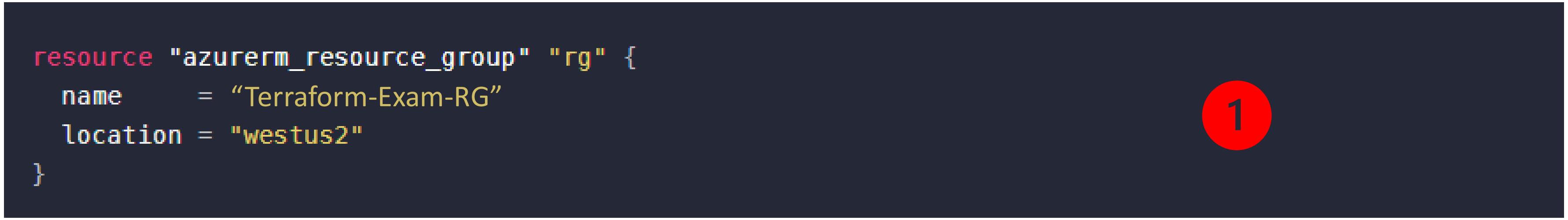
Using the depends_on

Creates a virtual network for your virtual machine.



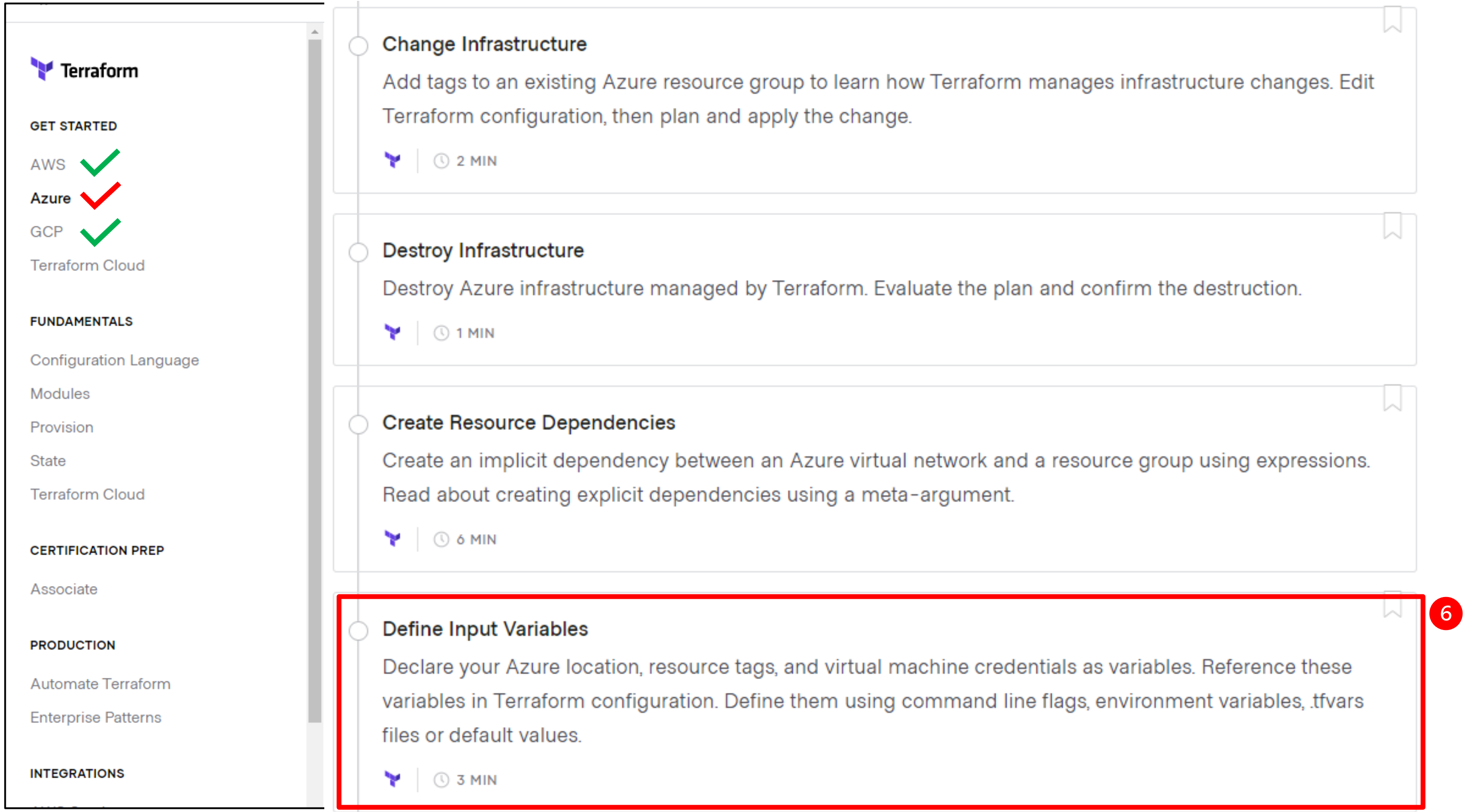
```
# Create a virtual network
resource "azurerm_virtual_network" "vnet" {
    name                = "myTFVnet"
    address_space       = ["10.0.0.0/16"]
    location            = "westus2"
    resource_group_name = “Terraform-Exam-RG”
}
```

Implicit dependencies <_type_>.<_name_>.<_property_>



```
resource "azurerm_resource_group" "rg" {
  name     = “Terraform-Exam-RG”
  location = "westus2"
}
```

HashiCorp Learn



Change Infrastructure

Add tags to an existing Azure resource group to learn how Terraform manages infrastructure changes. Edit Terraform configuration, then plan and apply the change.

2 MIN

Destroy Infrastructure

Destroy Azure infrastructure managed by Terraform. Evaluate the plan and confirm the destruction.

1 MIN

Create Resource Dependencies

Create an implicit dependency between an Azure virtual network and a resource group using expressions. Read about creating explicit dependencies using a meta-argument.

6 MIN

Define Input Variables

Declare your Azure location, resource tags, and virtual machine credentials as variables. Reference these variables in Terraform configuration. Define them using command line flags, environment variables, .tfvars files or default values.

3 MIN

Take-aways Knowledge

Variables.tf (file)

- Declare all variables
- var.<var name>

Terraform.tfvars (file)

- Values for all variables
- Automatically loads them
- *.auto.tfvars

Map (data type)

- Collection of string values
- Explicitly or implicitly

Object (data type)

Group different kinds of values, for example strings, bool values, and/or numbers

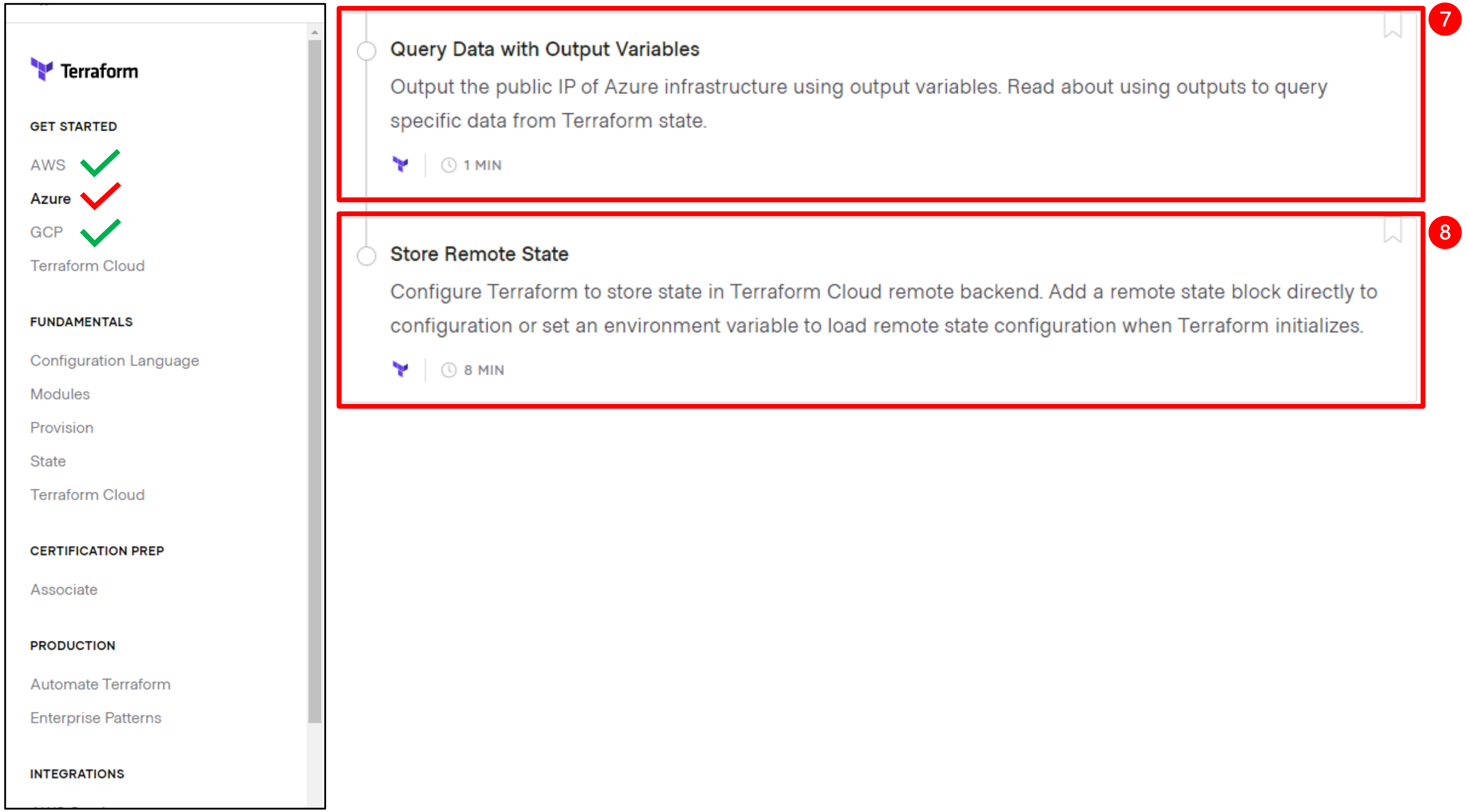
Lookup (function)

- Dynamic lookup in a map (data type)
- lookup (var.map_var1, var.map_var2)

-var (flag)

- Avoid being prompted for sensitive variable
- Terraform apply -var 'var1=val1' -var var2=val2'

HashiCorp Learn



Terraform

GET STARTED

AWS ✓

Azure ✓

GCP ✓

Terraform Cloud

FUNDAMENTALS

Configuration Language

Modules

Provision

State

Terraform Cloud

CERTIFICATION PREP

Associate

PRODUCTION

Automate Terraform

Enterprise Patterns

INTEGRATIONS

7

Query Data with Output Variables

Output the public IP of Azure infrastructure using output variables. Read about using outputs to query specific data from Terraform state.

1 MIN

8

Store Remote State

Configure Terraform to store state in Terraform Cloud remote backend. Add a remote state block directly to configuration or set an environment variable to load remote state configuration when Terraform initializes.

8 MIN

Take-aways Knowledge

Output

- Outputs after an apply operation
- Query these values

Remote backends

- Shared storage space for state data
- Migration of local state, reinitialize your project.

Locking

- Most of remote backends support locking.
- May wait until the current run is finished.
- You can run apply with -lock-timeout <time>

Amazon S3

State locking which can be enabled by setting the dynamodb_table

Demo (fmt, valideate)

Take-aways Knowledge

Terraform fmt (cmd)

- Rewrite Terraform configuration files
- Encourage consistency of style

Terraform validate

Validate configuration in modules, attribute names, and value types.

Provisioners

- Behaviors that can't be directly represented
- Model specific actions

Types of Provisioners

- local-exec Provisioner
- remote-exec Provisioner

Local-exec Provisioner

Invokes a local executable running Terraform, not on the resource.

Remote-exec Provisioner

Invokes a script on a remote resource

How to use Local Provisioners

```
resource "aws_instance" "web" {
  provisioner "local-exec" {
    command = "echo The server's IP address is ${self.private_ip}"
  }}
```

How to use Remote Provisioners

```
resource "aws_instance" "web" {
  provisioner "remote-exec" {
    inline = [
      "puppet apply",
      "consul join ${aws_instance.web.private_ip}",
    ] } }
```

Provisioners as a last resort.

Provisioners should only be used as a last resort.

Provisioners must connect

To remote system using SSH or WinRM.

Add provisioner block inside resource block

```
resource "aws_instance" "web" {
  # ...
  provisioner "local-exec" {
    command = "echo The server's IP address is ${self.private_ip}"
  } }
```

Demo (taint)

Take-aways Knowledge

Terraform taint (cmd)

- Marks a Terraform-managed resource as tainted
- Forcing it to be destroyed
- Terraform show

Auto tainted

Failed resources during provisioning

Symbol version = “~> 1.0”

Any version more than 1.0 and less than 2.0

Feedback and questions

YouTube

GitHub

FEEDBACK

LIVE STREAMING

Microsoft® MVP Most Valuable Professional

Thank you!

Cloud DevOps Visions

PLAN CODE BUILD TEST PACKAGE DEPLOY CONFIGURE MONITOR

www.clouddevopsvisions.com

Terraform ASSOCIATE HashiCorp CERTIFIED

Mohamed Radwan

Principal DevOps Consultant

mohamedradwan.com