UNIVERSITÀ CATTOLICA del Sacro Cuore

# Terraform

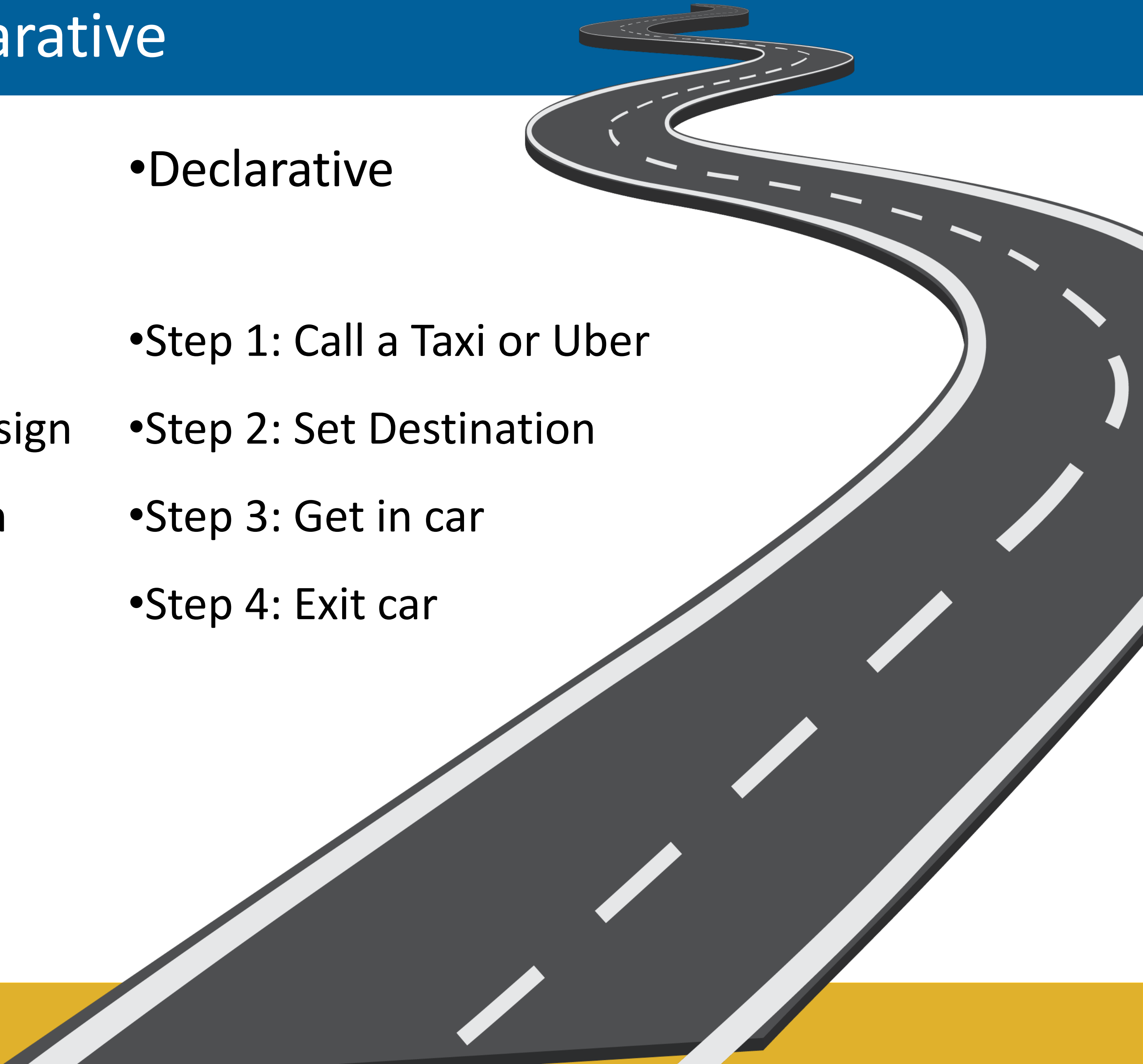
***Infrastructure as code*** *(IaC) is the process of managing and provisioning computer data centers through machine-readable definition files, rather than physical hardware configuration or interactive configuration tools*

**Source: https://en.wikipedia.org/wiki/Infrastructure_as_Code**

## Infrastructure as Code Benefits

- Automation drives a net reduction in costs
- Avoids configuration drift
- Improves security/compliance
- Aligns with cloud DevOps best CI/CD practices

## Imperative vs Declarative



- Imperative
- Step 1: Get in your car
- Step 2: Take a left turn at the Stop sign
- Step 3: Get on highway going north
- Step 4: Take Exit 274
- Step 5: Take a left into driveway
- Step 6: ……
- ……

- Declarative
- Step 1: Call a Taxi or Uber
- Step 2: Set Destination
- Step 3: Get in car
- Step 4: Exit car

## What are Declarative and Imperative Programming?

- ***Imperative programming*** is a programming paradigm that uses statements that change a program's state. (runbook)
- ***Declarative programming*** is a programming paradigm  that expresses the logic of a computation without describing its control flow. (no / less runbook needs)

# What are Declarative and Imperative Programming?

Many of us are used to imperative  languages used for setting up an environment.

Kubernetes, OpenShift and many (public) cloud environment can instead be addressed in a declarative manner.

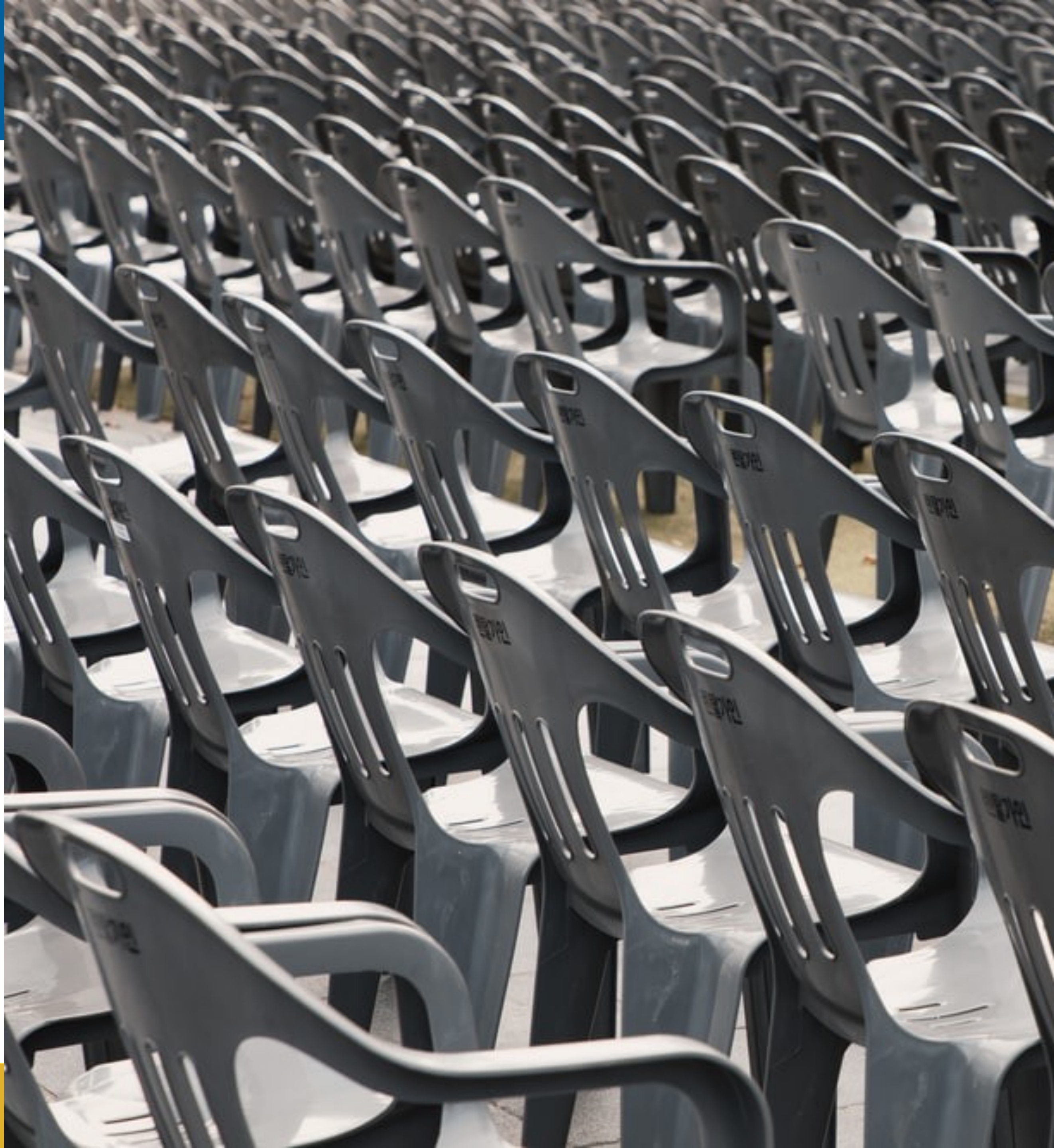
Declarative architectures allow for self-optimizing and self-healing systems that are easier to manage than imperative architectures.

This is a key driver (to avoid spending to much time on making runbooks and needed exception handling ) for Terraform, Helm, Kubernetes, OpenShift…. And IBM Schematics

## Idempotency

- *"Property of an operation whereby it can applied multiple times without changing the result beyond the initial application"*

- Running a script once or 100 times should result in the same thing.
- Enables you to effectively manage your infrastructure automation scripts
- Avoid configuration drift



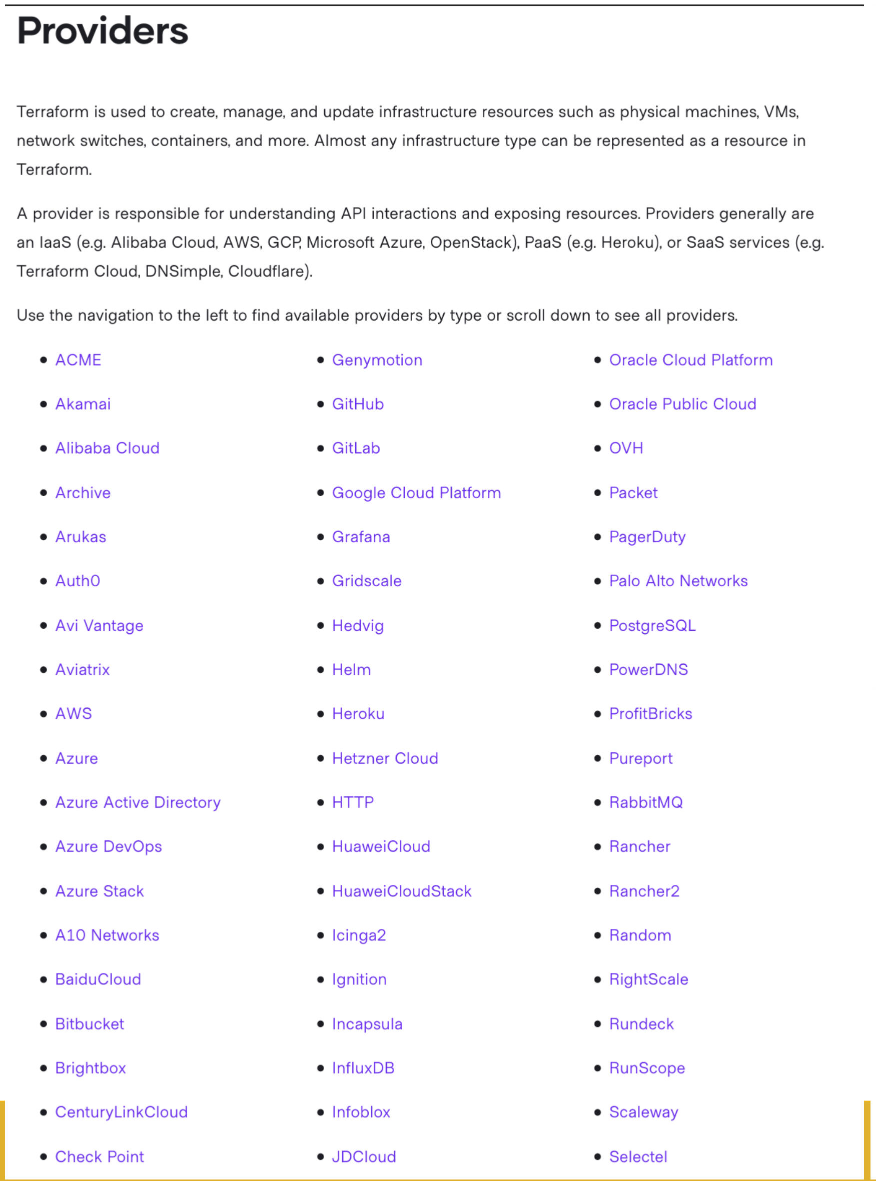
# Terraform

- Open Source (MPL 2.0)
- Created by HashiCorp in 2014
- Written in Golang
- Pluggable Architecture

- **Develop**
  - Write your template
- **Plan**
  - Verify resources to be provisioned
- **Apply**
  - Create resources
- **Update**
  - Make changes to infrastructure

## Terraform Providers

- Abstraction layers for cloud providers
- Expose resources that can be configured in Terraform
- Maintained by HashiCorp or community



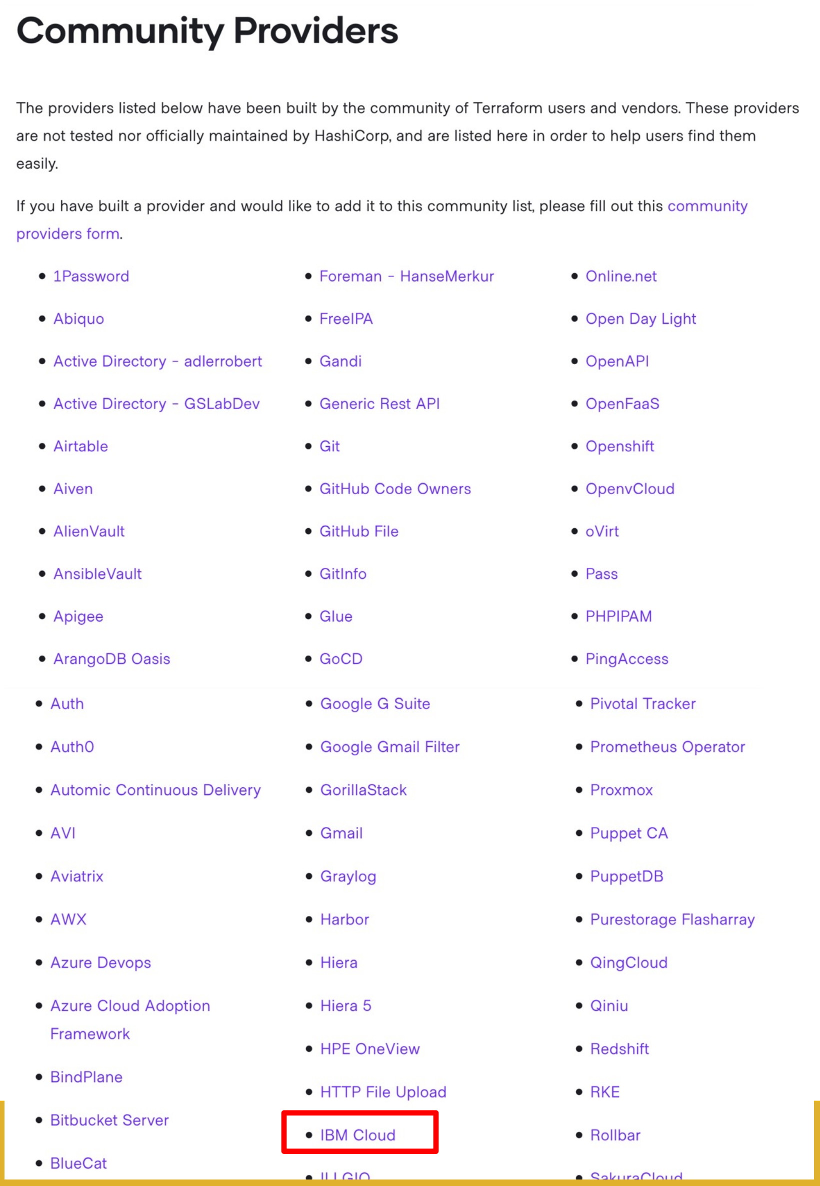
### Providers

Terraform is used to create, manage, and update infrastructure resources such as physical machines, VMs, network switches, containers, and more. Almost any infrastructure type can be represented as a resource in Terraform.

A provider is responsible for understanding API interactions and exposing resources. Providers generally are an IaaS (e.g. Alibaba Cloud, AWS, GCP, Microsoft Azure, OpenStack), PaaS (e.g. Heroku), or SaaS services (e.g. Terraform Cloud, DNSimple, Cloudflare).

Use the navigation to the left to find available providers by type or scroll down to see all providers.

- ACME
- Akamai
- Alibaba Cloud
- Archive
- Arukas
- Auth0
- Avi Vantage
- Aviatrix
- AWS
- Azure
- Azure Active Directory
- Azure DevOps
- Azure Stack
- A10 Networks
- BaiduCloud
- Bitbucket
- Brightbox
- CenturyLinkCloud
- Check Point
- Genymotion
- GitHub
- GitLab
- Google Cloud Platform
- Grafana
- Gridscale
- Hedvig
- Helm
- Heroku
- Hetzner Cloud
- HTTP
- HuaweiCloud
- HuaweiCloudStack
- Icinga2
- Ignition
- Incapsula
- InfluxDB
- Infoblox
- JDCloud
- Oracle Cloud Platform
- Oracle Public Cloud
- OVH
- Packet
- PagerDuty
- Palo Alto Networks
- PostgreSQL
- PowerDNS
- ProfitBricks
- Pureport
- RabbitMQ
- Rancher
- Rancher2
- Random
- RightScale
- Rundeck
- RunScope
- Scaleway
- Selectel



### Community Providers

The providers listed below have been built by the community of Terraform users and vendors. These providers are not tested nor officially maintained by HashiCorp, and are listed here in order to help users find them easily.

If you have built a provider and would like to add it to this community list, please fill out this community providers form.

- 1Password
- Abiquo
- Active Directory - adlerrobert
- Active Directory - GSLabDev
- Airtable
- Aiven
- AlienVault
- AnsibleVault
- Apigee
- ArangoDB Oasis
- Auth
- Auth0
- Automic Continuous Delivery
- AVI
- Aviatrix
- AWX
- Azure Devops
- Azure Cloud Adoption Framework
- BindPlane
- Bitbucket Server
- BlueCat
- Foreman - HanseMerkur
- FreeIPA
- Gandi
- Generic Rest API
- Git
- GitHub Code Owners
- GitHub File
- GitInfo
- Glue
- GoCD
- Google G Suite
- Google Gmail Filter
- GorillaStack
- Gmail
- Graylog
- Harbor
- Hiera
- Hiera 5
- HPE OneView
- HTTP File Upload
- IBM Cloud
- Online.net
- Open Day Light
- OpenAPI
- OpenFaaS
- Openshift
- OpenvCloud
- oVirt
- Pass
- PHPIPAM
- PingAccess
- Pivotal Tracker
- Prometheus Operator
- Proxmox
- Puppet CA
- PuppetDB
- Purestorage Flasharray
- QingCloud
- Qiniu
- Redshift
- RKE
- Rollbar

# Configuration language (HCL)

Template

```
provider ”ibm" {
  generation = ”2”
}

variable "name" {
  default = "cluster"
}

variable "worker_count" {
  default = "1"
}

variable "flavor" {
  default = "bx2.2x8"
}

data "ibm_container_cluster_config" "cluster_config" {
  cluster_name_id = ibm_container_vpc_cluster.cluster.id
}

data "ibm_resource_group" "resource_group" {
  name = var.resource_group
}
```

```
resource "ibm_container_vpc_cluster" "cluster" {
  name              = "${var.cluster_name}
                       ${random_id.name1.hex}"
  vpc_id            = ibm_is_vpc.vpc1.id
  kube_version      = var.kube_version
  flavor            = var.flavor
  worker_count      = var.worker_count
  resource_group_id = data.ibm_resource_group.
                              resource_group.id

  zones {
    subnet_id = ibm_is_subnet.subnet1.id
    name      = local.ZONE1
  }
}

output "cluster_config_file_path" {
  value =
         data.ibm_container_cluster_config
             .cluster_config.config_file_path
}
```

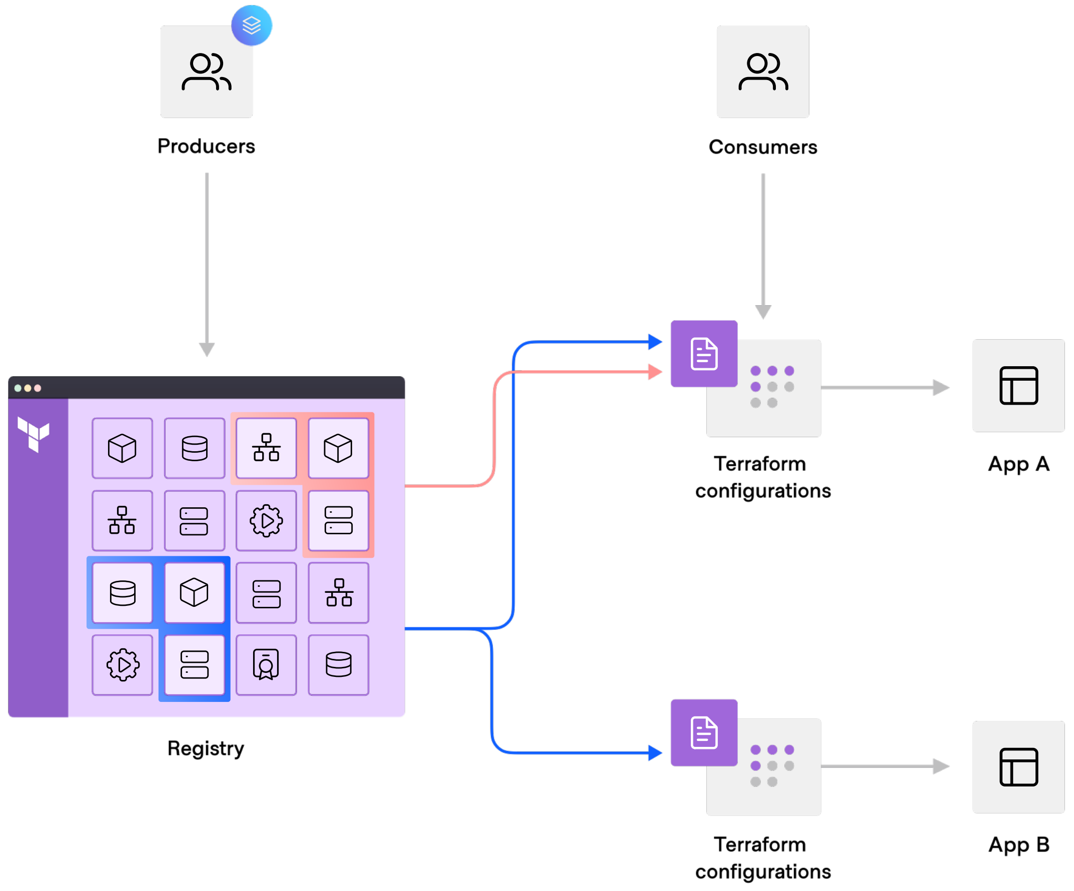
# Architecture

## Compose, collaborate, reuse

It Promote best practices across the organization with reusable, pre-approved modules and manage the module lifecycle from end to end.

## Standardize the provisioning process

Help teams collaborate smoothly and securely with a unified, standardized workflow.



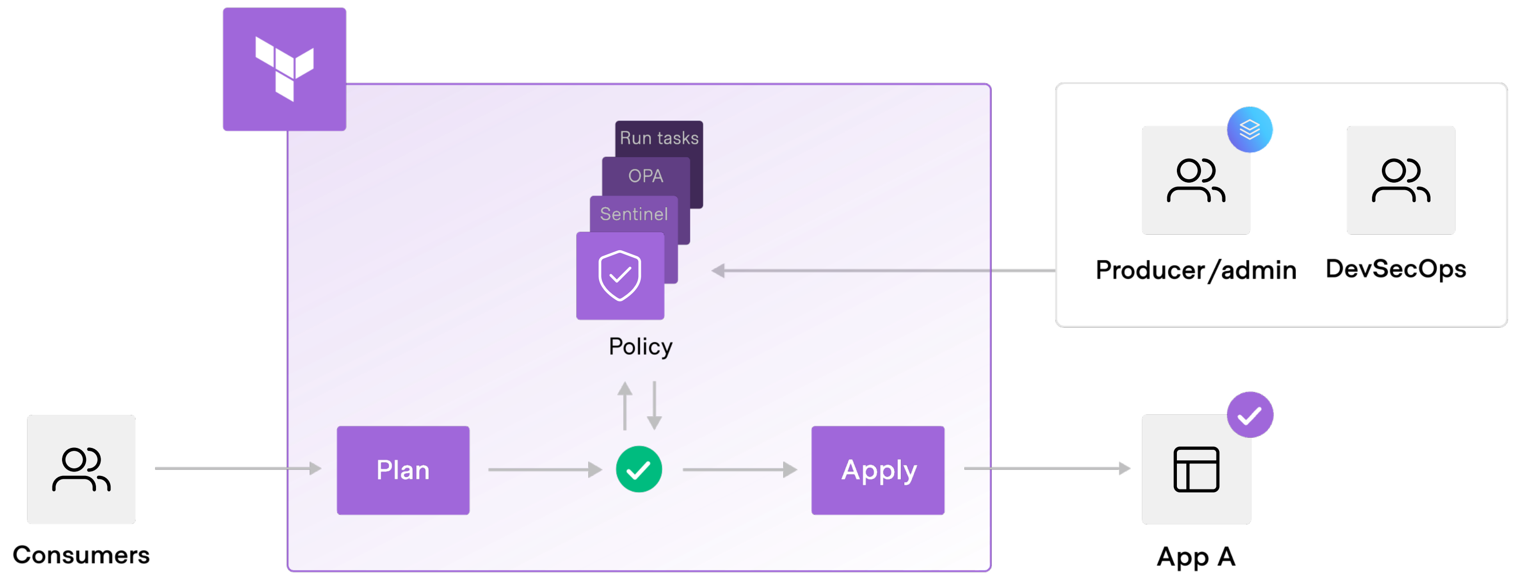
# Architecture

## Ensure security

It limits risk and maintains efficiency with automated policy enforcement for compliance when provisioning.

## Integrate third-party services

Leverage run tasks that cover security and code scanning, cost control, and regulatory compliance checks for enhanced validation.



# Architecture

## Detect and alert

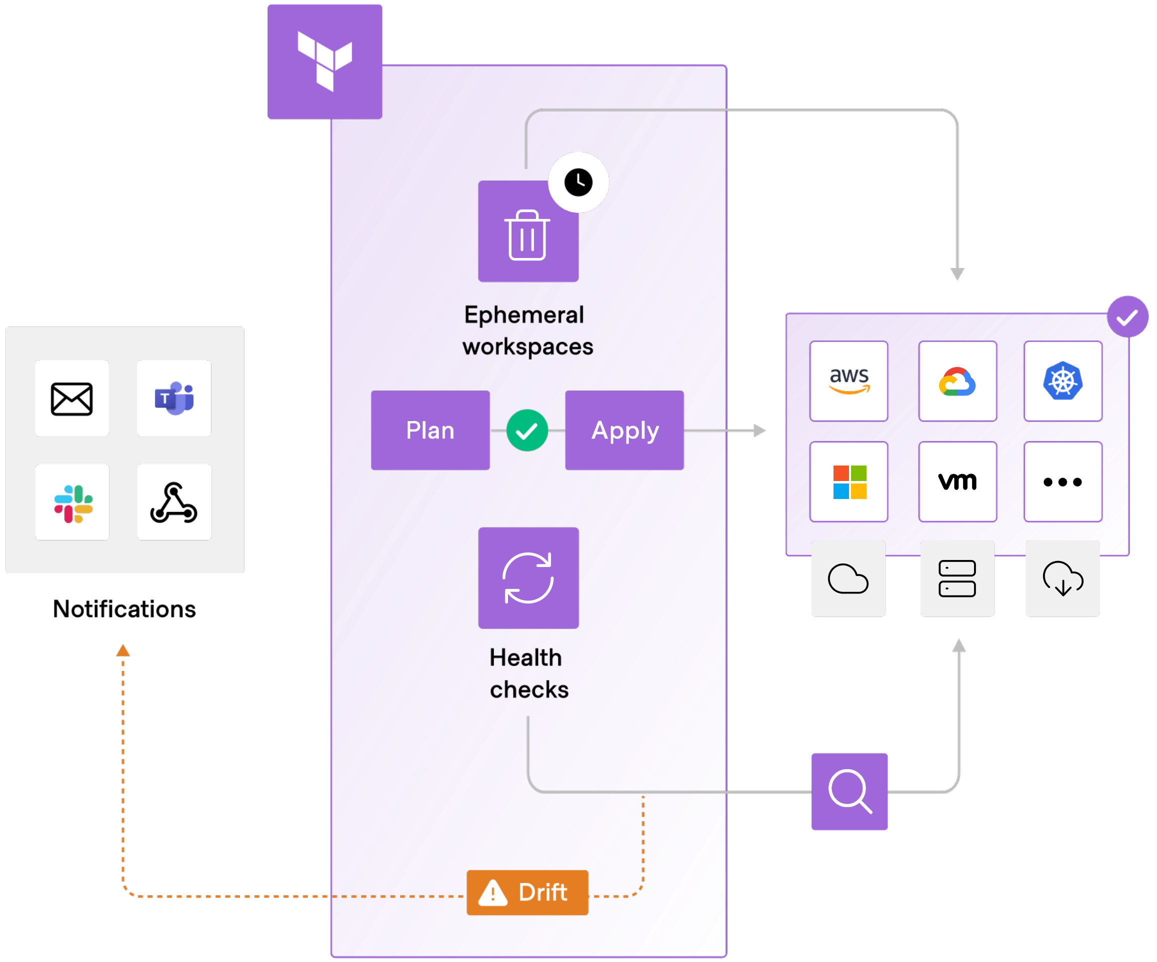
It can create alerts when unexpected infrastructure changes happen

## Validate deployed workloads

Monitor if workloads are operating as expected to respond to image revocation, close security gaps, control budgets, deal with certificate expiration, and more.

## Ephemeral workspaces

Enable automatic destruction of resources without manual cleanup, reduce infrastructure costs, and streamline workspace management.



## Q&A

# Thanks

for your attention!

Questions?

Federico Accetta federico_accetta@it.ibm.com