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**.

***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.
* Terraforms 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.

Terraform providers manage resources by communicating between Terraform and target APIs.

When multiple users or automation tools run the same Terraform configuration, they should all use the same versions of their required providers. There are two ways for you to manage provider versions in your configuration.

1. Specify provider version constraints in your configuration's terraform block.
2. Use the [dependency lock file](https://www.terraform.io/docs/language/dependency-lock.html)