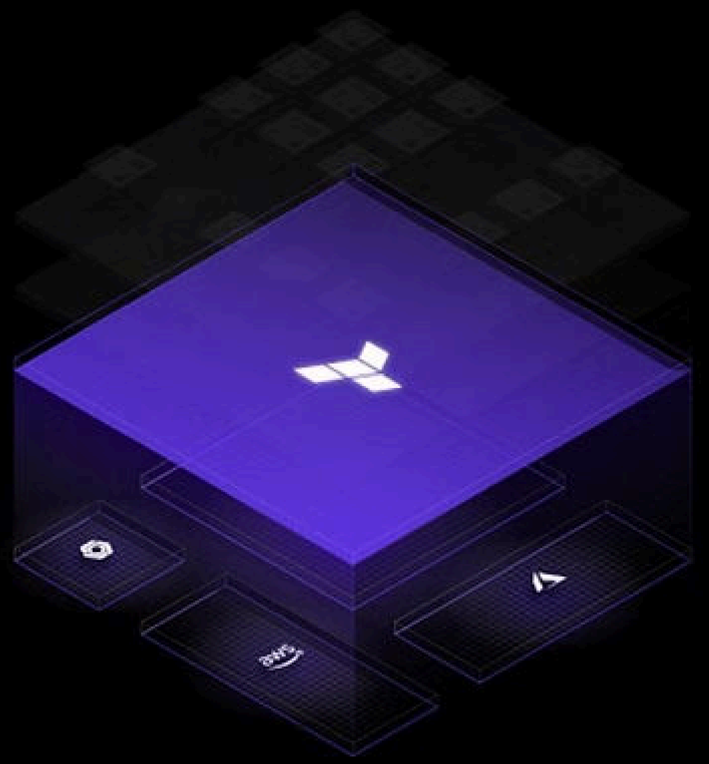


HashiCorp Terraform

What is Terraform?

Terraform is an open-source infrastructure as code software tool created by HashiCorp. It allows users to define and provision a complete data center infrastructure using a high-level configuration language known as HashiCorp Configuration Language (HCL), or optionally JSON.

TA **by thexyzcompany org**

Infrastructure as Code

1 **Efficient**

Infrastructure as code allows for efficient resource management through automation and version control.

2 **Scalable**

It enables scalability by treating infrastructure as software, making it easier to scale resources up or down as needed.

3 **Consistent**

Ensures consistent deployment of infrastructure across different environments, reducing errors and ensuring reliability.

Key Features of Terraform

Declarative Syntax

Terraform uses a declarative syntax, allowing users to express the desired end state without specifying the steps to reach it.

Resource Graph

It creates a resource graph to parallelize the creation and modification of various non-dependent resources.

Interoperability

Terraform can integrate with other tools to create a cohesive infrastructure management ecosystem.

HashiCorp Terraform

Getting Started with Terraform

1 **Installation**

Begin by installing Terraform on your local machine and ensuring it's added to the system's PATH.

2 **Configuration**

Create a new directory, define the required infrastructure in configuration files, and initialize the working directory.

3 **Execution**

Run 'terraform plan' to create an execution plan and then 'terraform apply' to apply the changes.

Writing Terraform Configuration Files

Declarative

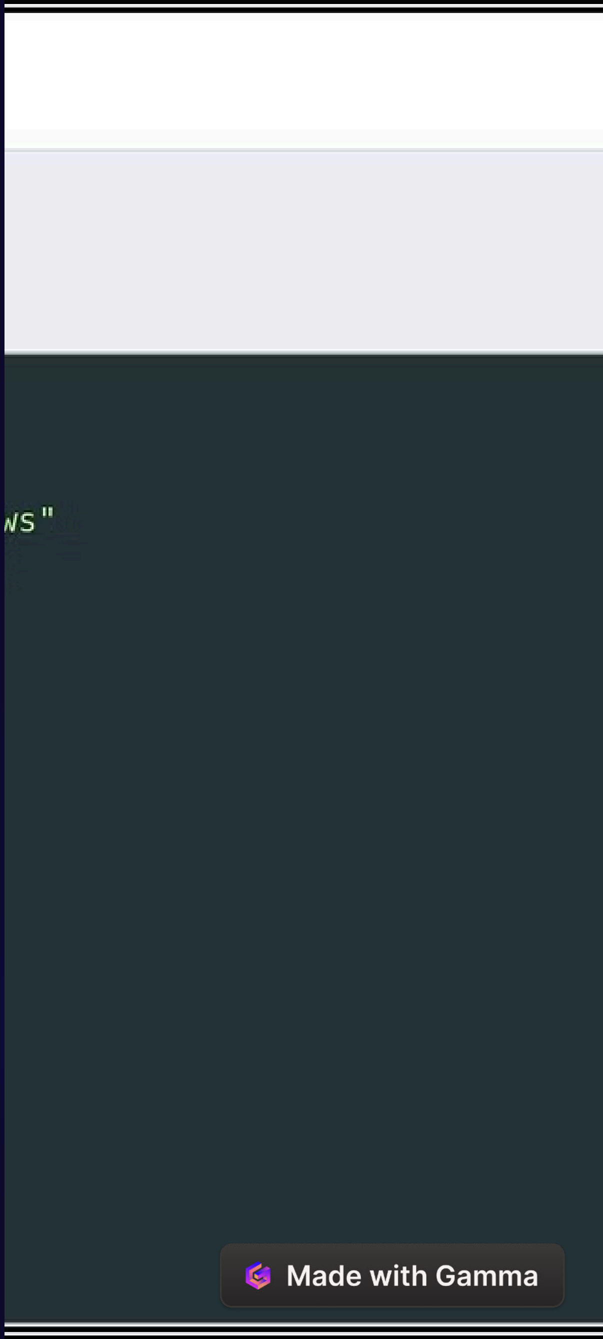
Configuration files represent infrastructure in a declarative manner, specifying the desired end state.

Modular

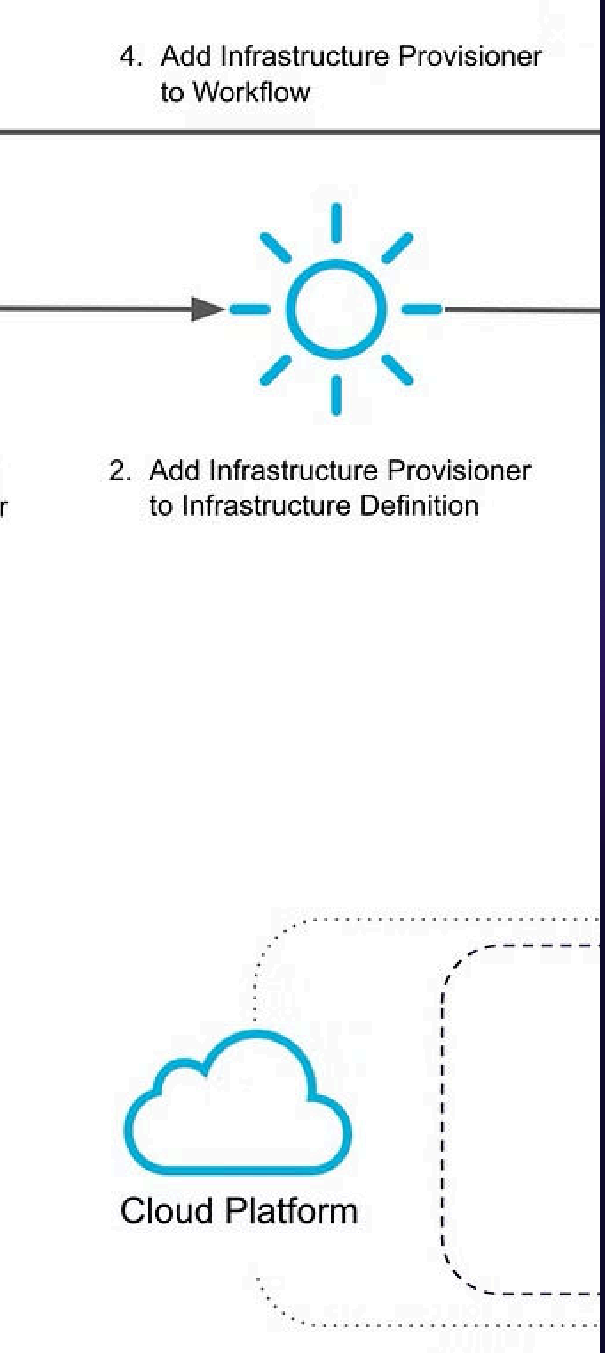
This modular approach allows for reusable components, making configurations more maintainable and scalable.

Versioned

Configuration files can be versioned, providing an audit trail and enabling collaboration among teams.



```
ws"
```


4. Add Infrastructure Provisioner to Workflow
2. Add Infrastructure Provisioner to Infrastructure Definition
Cloud Platform

Provisioning Infrastructure with Terraform

1 **Resource Creation**

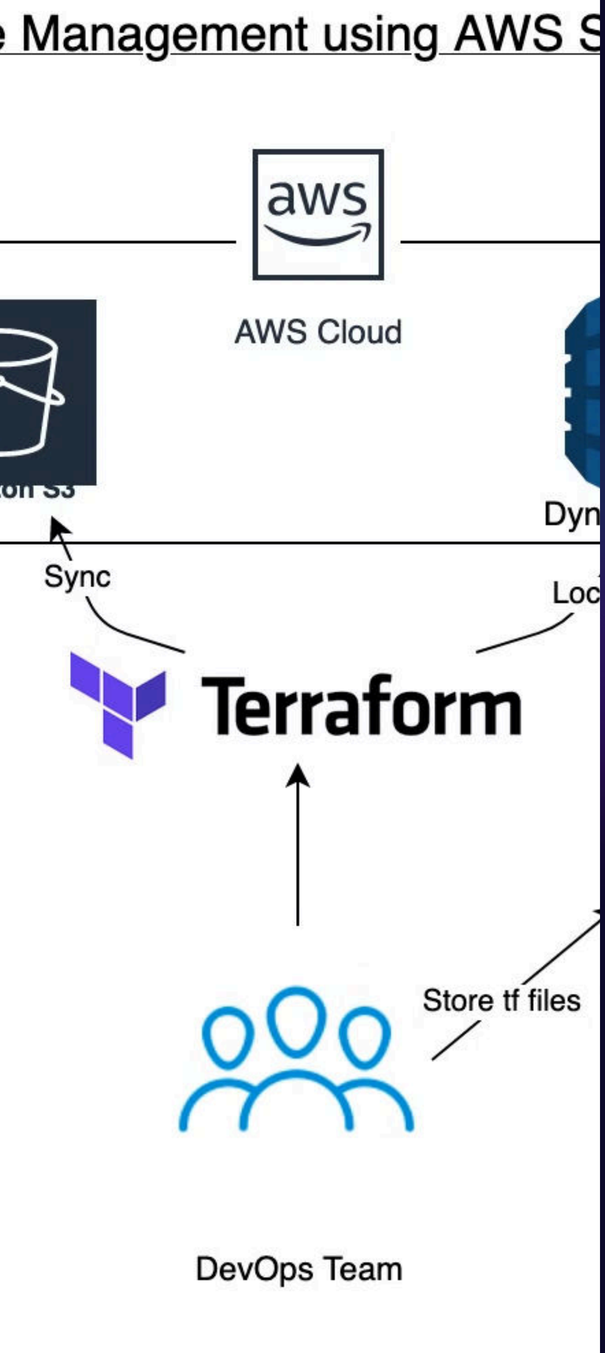
Terraform provisions infrastructure resources based on the defined configuration files and executes the plan.

2 **Dependency Management**

It handles dependencies between resources, ensuring proper sequencing during infrastructure provisioning.

3 **State Management**

Terraform keeps track of the infrastructure's state to accurately determine required changes.



Managing Terraform State

1

Remote Backend

Manage the state file remotely with a compatible storage service using the backend block configuration.

2

State Locking

Enable locking to prevent concurrent executions and ensure consistency when making changes to the infrastructure.

3

State Versioning

Utilize state versioning for auditability and to revert to earlier versions if necessary.

Best Practices for Using Terraform

1

Use Modules

Organize and encapsulate configurations using modules to promote reusability and maintainability.

2

Apply Code Review

Implement code review processes to ensure high-quality configurations and foster knowledge sharing.

3

Use Variables

Leverage variables to parameterize configurations and simplify environment-specific changes.

4

Implement Testing

Establish automated testing to validate configurations and prevent potential issues in the infrastructure.