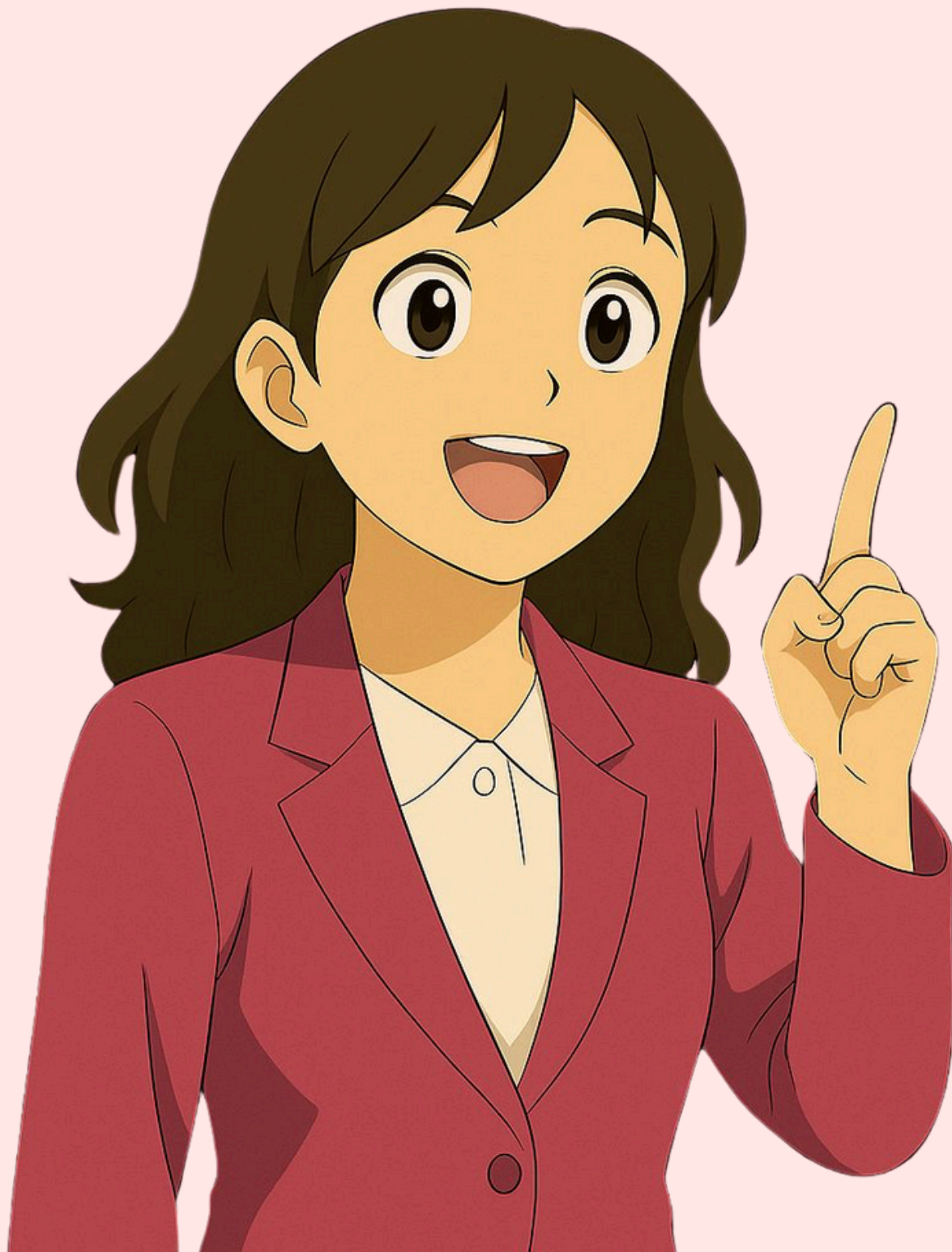# TERRAFORM
STATE FILE
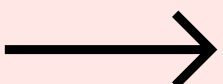## DISASTER

Nensi Ravaliya

Why 80% of DevOps Teams Skip the Lock

The state file is the **heart of Terraform.**

- It remembers everything you've built so Terraform knows what's real and what's not.
- But what if two people change it at the same time? Chaos.

Nensi Ravaliya

→ What is Terraform State?

- Terraform uses a file called **terraform.tfstate** to track all your resources.

- It's like Terraform's memory of your cloud. Without this file, Terraform is blind. it doesn't know what exists.
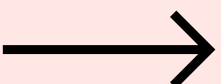
→

Nensi Ravaliya

WHY ?

- When multiple people or pipelines touch the same state file,
- Terraform can't manage who's changing what.
- That's where State Locking comes in.
- it ensures only **one operation runs at a time.**

→

Nensi Ravaliya
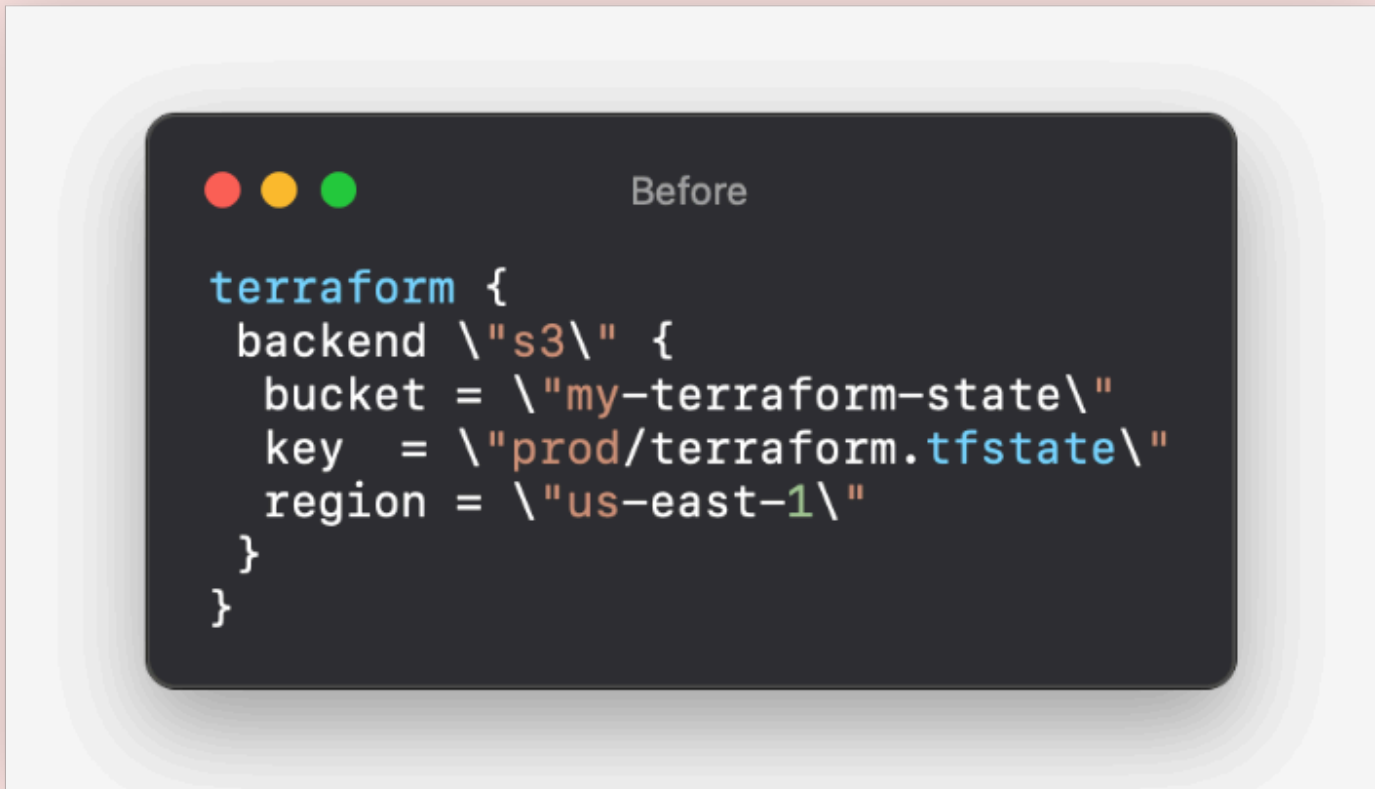
Without Locking (The Real Pain)

- **Configuration Drift:** your infra and state go out of sync.
- **Concurrent Modifications:** multiple updates overwrite each other.
- **Corrupted State:** half-applied resources, missing data, and broken infra.
- **Result?** Endless debugging and unpredictable behavior.

Nensi Ravaliya

Before: The Unprotected Setup

```
terraform {
 backend \"s3\" {
  bucket = \"my-terraform-state\"
  key  = \"prod/terraform.tfstate\"
  region = \"us-east-1\"
 }
}
```

Before

✅ Works fine for a single developer.
❌ Breaks the moment your team scales.

→

Nensi Ravaliya
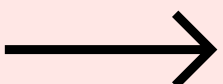
→ After: The Right Way

```
● ● ●                    After

terraform {
 backend \"s3\" {
  bucket        = \"my-terraform-state\"
  key           = \"prod/terraform.tfstate\"
  region        = \"us-east-1\"
  dynamodb_table = \"terraform-lock\"
 }
}
```

- That single dynamodb_table line creates a lock in DynamoDB.
- Terraform won't let anyone else apply until the current one finishes.

→

Nensi Ravaliya

The Impact After Locking

IMPACT

- No more **parallel** updates.
- State **consistency** across all teams.
- **Confident deployments:** no unexpected infra changes.
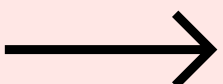- **Enterprise grade safety** with a simple setup.
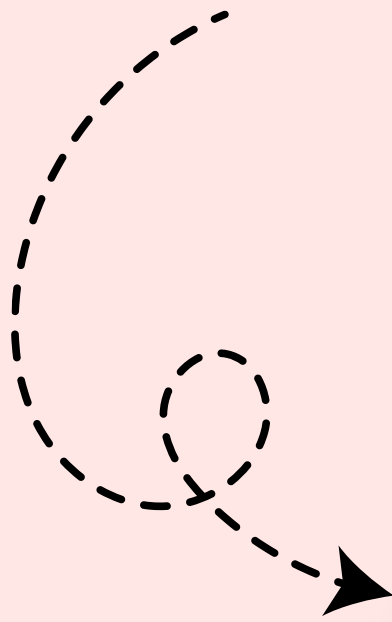
Nensi Ravaliya

EXAMPLE

- A team we worked with **wasted 60+ hours** fixing corrupted states.
- After **enabling DynamoDB** locking zero conflicts.
- Same infra. Same team. Just **smarter execution**.

Nensi Ravaliya

Repost and Follow

Nensi Ravaliya

for more content

Want to build your career in cloud?

Subscribe to Yatri Cloud Channel