Code Your Cloud: Infrastructure as Code Best Practices with Terraform

Conf42 DevSecOps 2024

Madhu Kumar Yeluri
DevSecOps Consultant / Principal Cloud Architect

Fortune Technologies Ltd

IT CONSULTING

IT Consulting | Security | Cloud Migrations | DevSecOps | AI/ML

Fortune Technologies Ltd
IT CONSULTING

>_whoami

Madhu Kumar Yeluri
AWS Hero & User Group Leader Hungary
Madhu is an accomplished Principal Cloud Architect and DevSecOps Consultant with more than two decades of experience in the IT industry across different parts of the world. In addition to his professional achievements, Madhu is also passionate about giving back to the community. Open-source Software (OSS) supporter. He is humbled to serve as an AWS Hero and AWS User Group Leader, DevSecCon Chapter Leader for Hungary, DevOps Institute Brand Ambassador and Chapter Leader, and HashiCorp User Group Leader for Hungary.

Blogs: https://medium.com/@cloudgeek7 / https://dev.to/cloudgeek7
GitHub: https://github.com/cloudgeek7
Discord Container Builders: https://discord.gg/TejhNdfe26

Madhu Kumar
Principal Cloud Architect | Digital Leader | Technologist | Engineer | Pu...

Introduction

Overview: Terraform as a tool for Infrastructure as Code (IaC).

Objective: Share IaC best practices using Terraform for scalable and version-controlled cloud infrastructure.

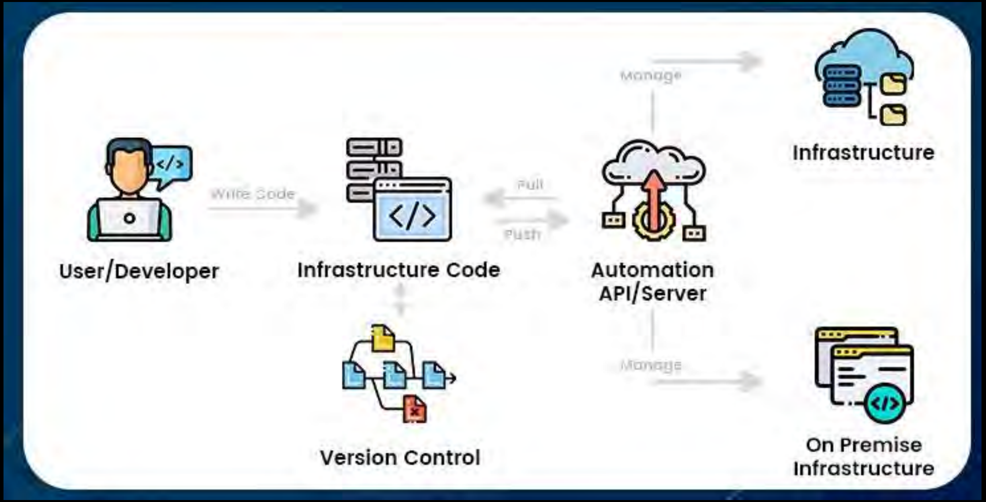
Fortune Technologies Ltd
IT CONSULTING

What is Infrastructure as Code (IaC)?

IaC: Manage and provision infrastructure via code.

Benefits of IaC:
- Consistency in deployments
  Version control for change tracking

- Scalability for faster provisioning



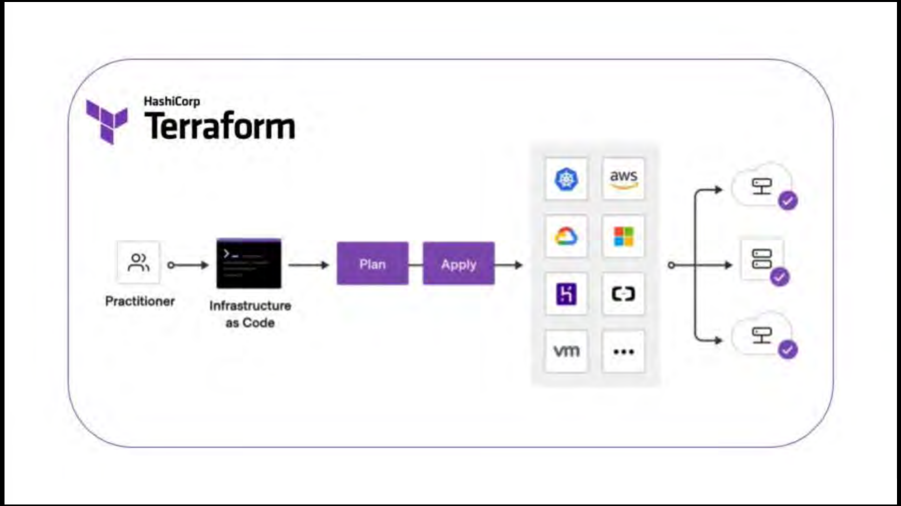
Fortune Technologies Ltd
IT CONSULTING

Why Terraform?

Provider-Agnostic: Works with AWS, Azure, GCP, and more.

Declarative Language: Focus on end-state, not process.

Strong Community & Ecosystem: Numerous modules and support.



Best Practices Overview

Organising Code Structure

Modules for Reusability

State Management

Environment Segmentation

Security Practices

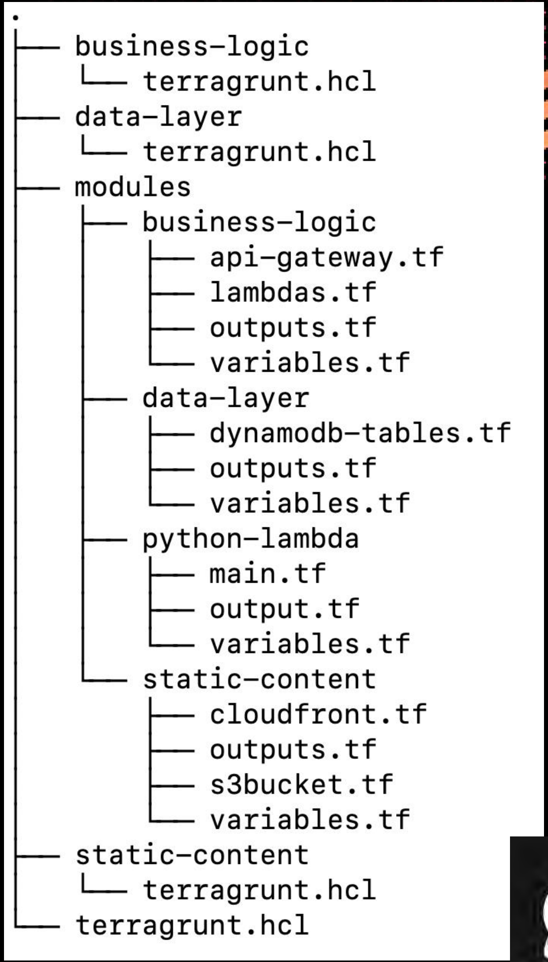
Automating and Testing Infrastructure

Organising Code Structure

Separate by Service or Component (e.g., network, database, application)

Standardise File Naming (e.g., main.tf, variables.tf, outputs.tf)

Example: Show a sample folder structure for an app



```
.
├── business-logic
│   └── terragrunt.hcl
├── data-layer
│   └── terragrunt.hcl
├── modules
│   ├── business-logic
│   │   ├── api-gateway.tf
│   │   ├── lambdas.tf
│   │   ├── outputs.tf
│   │   └── variables.tf
│   ├── data-layer
│   │   ├── dynamodb-tables.tf
│   │   ├── outputs.tf
│   │   └── variables.tf
│   ├── python-lambda
│   │   ├── main.tf
│   │   ├── output.tf
│   │   └── variables.tf
│   └── static-content
│       ├── cloudfront.tf
│       ├── outputs.tf
│       ├── s3bucket.tf
│       └── variables.tf
├── static-content
│   └── terragrunt.hcl
└── terragrunt.hcl
```

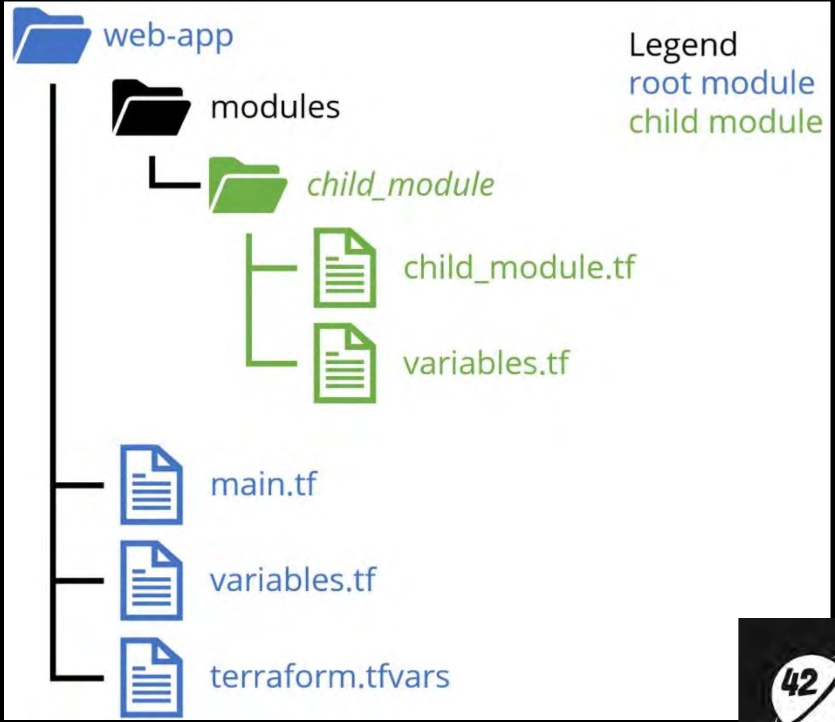
Fortune Technologies Ltd
IT CONSULTING

Modules for Reusability

Why Modules? Reusable and maintainable code blocks

Module Design: Use clear inputs/outputs and single-purpose modules

Example: AWS VPC module structure



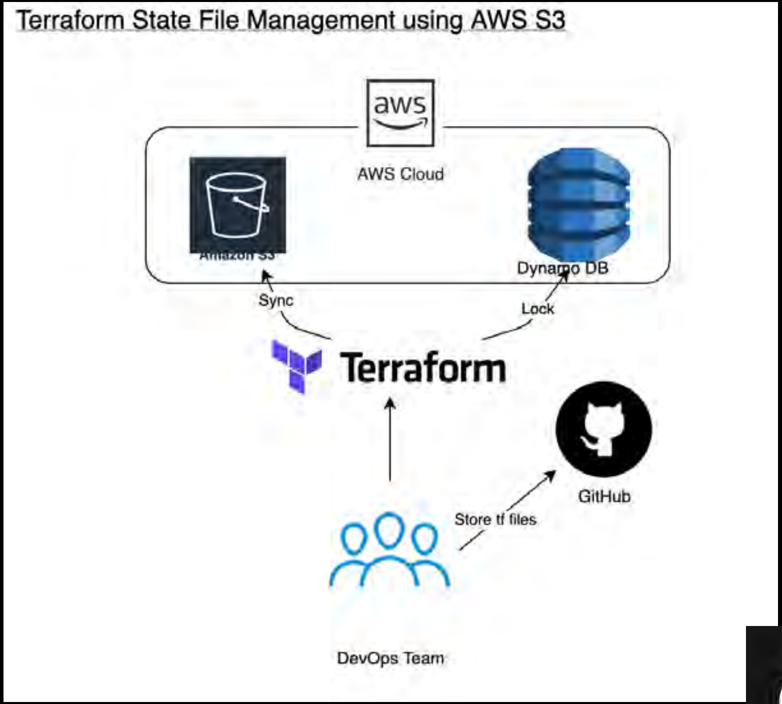
Fortune Technologies Ltd
IT CONSULTING

State Management Best Practices

Remote State Storage: Use S3, with DynamoDB for locking

Secure State Files: Enable encryption as they contain sensitive information

State Versioning: Enable versioning for change tracking

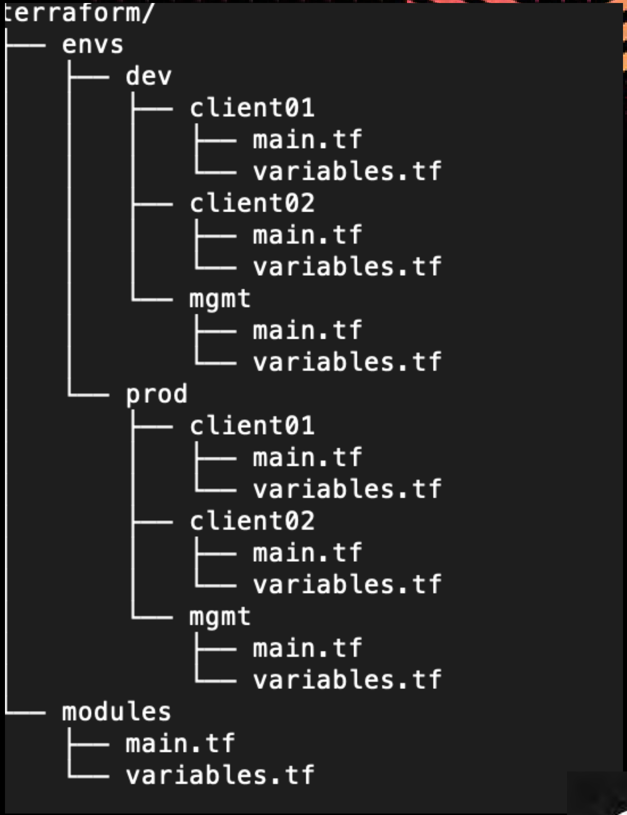


Environment Segmentation

Use Workspaces or Directories to separate Dev, Staging, Prod environments.

Maintain Isolation and Consistency across environments.

Example: Project directory with segmented environments.



```
erraform/
├── envs
│   ├── dev
│   │   ├── client01
│   │   │   ├── main.tf
│   │   │   └── variables.tf
│   │   ├── client02
│   │   │   ├── main.tf
│   │   │   └── variables.tf
│   │   └── mgmt
│   │       ├── main.tf
│   │       └── variables.tf
│   └── prod
│       ├── client01
│       │   ├── main.tf
│       │   └── variables.tf
│       ├── client02
│       │   ├── main.tf
│       │   └── variables.tf
│       └── mgmt
│           ├── main.tf
│           └── variables.tf
└── modules
    ├── main.tf
    └── variables.tf
```

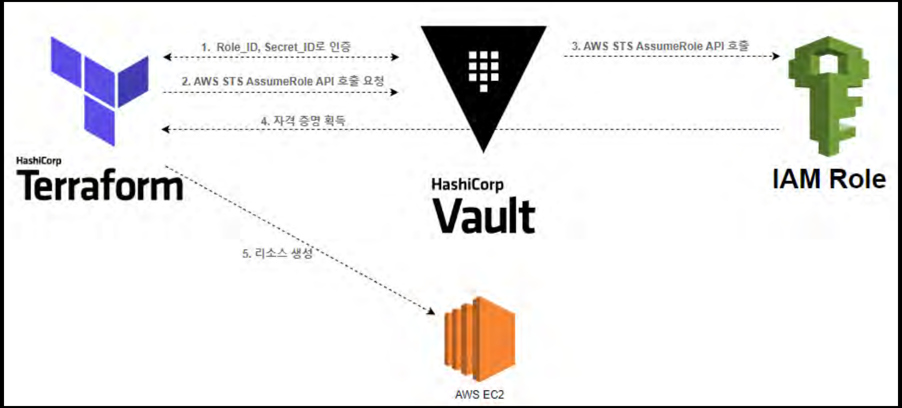
Fortune Technologies Ltd
IT CONSULTING

Security Practices

Secret Management: Use Vault, AWS Secrets Manager, etc., avoid hardcoding

Role-Based Access: Grant least-privilege access

Enable Auditing for compliance and activity tracking

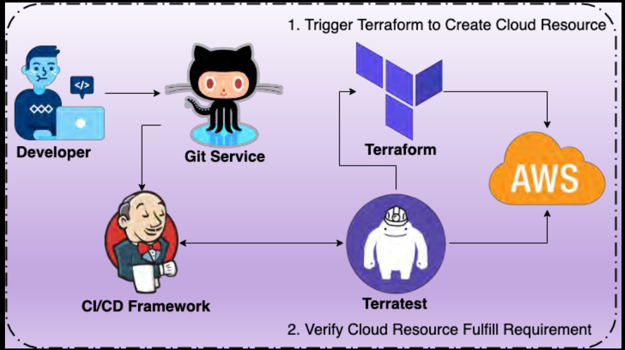

1. Role_ID, Secret_ID로 인증
2. AWS STS AssumeRole API 호출 요청
3. AWS STS AssumeRole API 호출
4. 자격 증명 획득
5. 리소스 생성
HashiCorp
Terraform
HashiCorp
Vault
IAM Role
AWS EC2

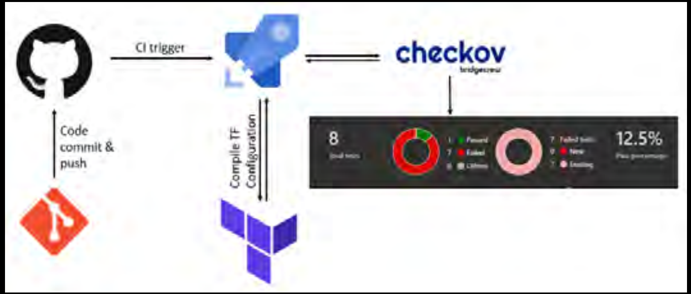
Automating and Testing Infrastructure

Integrate with CI/CD pipelines for testing, validation, and deployment

Automated Testing: Use tools like Terratest or Checkov

Example Workflow: CI/CD pipeline with Terraform linting and apply stages


1. Trigger Terraform to Create Cloud Resource
Developer
Git Service
Terraform
AWS
CI/CD Framework
Terratest
2. Verify Cloud Resource Fulfill Requirement



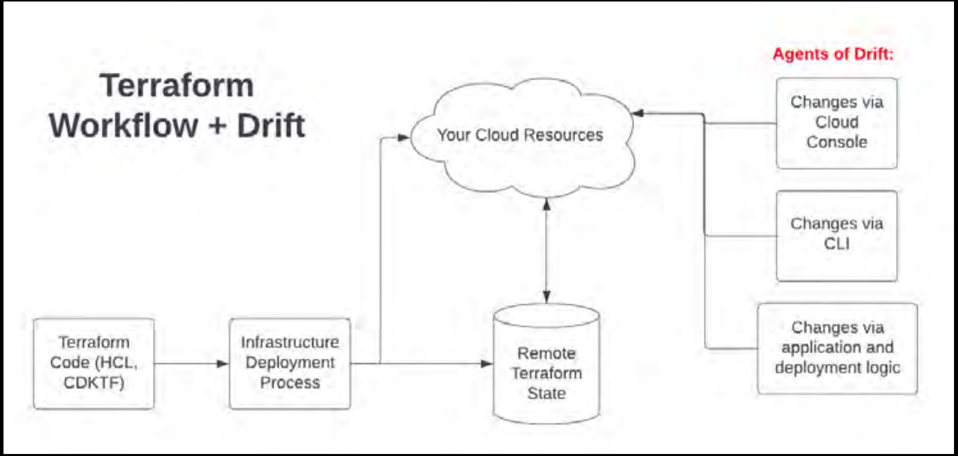
Fortune Technologies Ltd
IT CONSULTING

Managing Drift and Consistency

Understanding Drift: Identify changes made outside Terraform

Drift Detection: Regularly run `terraform plan` and automate checks

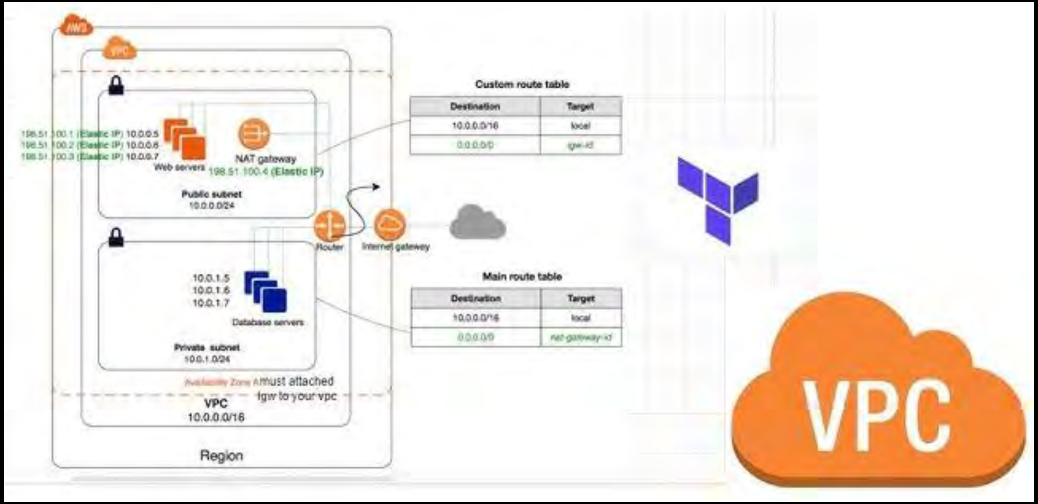
Example: Drift detection in a DevOps pipeline


Terraform Workflow + Drift
Agents of Drift:
Your Cloud Resources
Changes via Cloud Console
Changes via CLI
Changes via application and deployment logic
Terraform Code (HCL, CDKTF)
Infrastructure Deployment Process
Remote Terraform State

Case Study: Example Workflow

Walkthrough: AWS VPC deployment with subnets using Terraform

Applied Best Practices:
- Module Reuse
- Remote State Management
- CI/CD Automation



Custom route table

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	igw-id

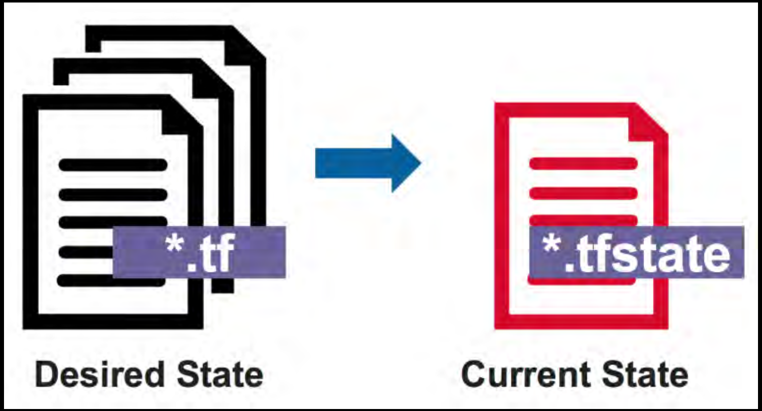
Main route table

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	nat-gateway-id

Challenges and Solutions

Common Challenges: State conflicts, sensitive data, drift management

Solutions: Use workspaces, remote backends, and automated CI/CD pipelines



Additional Tools & Resources

Terraform Cloud/Enterprise: Collaboration and policy management

Terragrunt: Enhances Terraform workflows for larger teams

Community Resources: Terraform Registry, GitHub.

Fortune Technologies Ltd
IT CONSULTING

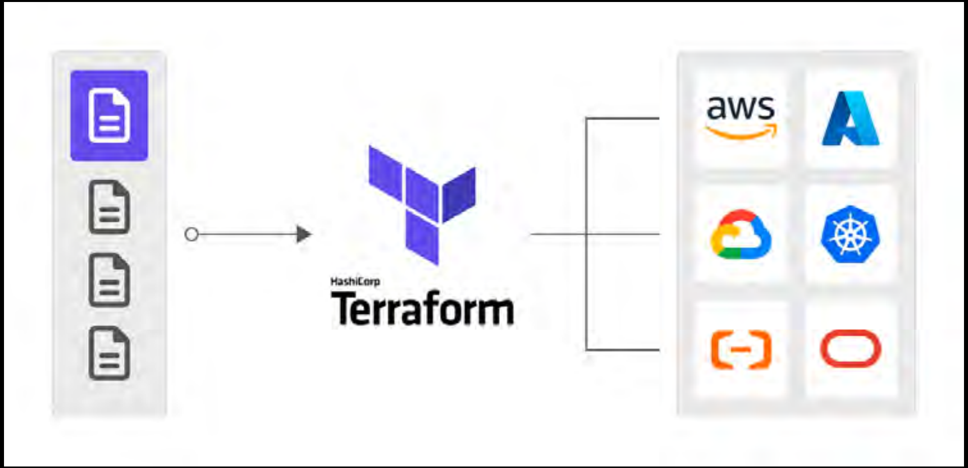
Key Takeaways

Consistency: Structure code for maintainability

Modularity: Reuse with modules

Security: Prioritize state and secret management

Automation: Use CI/CD for reliable infrastructure



HashiCorp User Group Budapest, Hungary

Fortune Technologies Limited
IT Consulting | Security | Cloud Migrations | DevSecOps | AI/ML

Thank You

Fortune Technologies Limited
IT Consulting | Security | Cloud Migrations | DevSecOps | AI/ML