MÓDULO 13: TERRAFORM MODULAR

Infraestructura como Código para MLOps

Guía MLOps v5.0: Senior Edition | DuqueOM | Noviembre 2025

🏗️ MÓDULO 13: Terraform Modular

Infraestructura Reproducible y Versionada

"Si no puedo recrear tu infra con un comando, no es reproducible."

Duración	Teoría	Práctica
5-6 horas	30%	70%

Lo Que Lograrás

1. **Estructurar** código Terraform modular
2. **Gestionar** state de forma segura
3. **Implementar** ambientes (dev/staging/prod)
4. **Aplicar** patrones de IaC para ML

13.1 Estructura Modular

```
infra/
├── terraform/
│   ├── modules/                    # Módulos reutilizables
│   │   ├── networking/
│   │   │   ├── main.tf
│   │   │   ├── variables.tf
│   │   │   └── outputs.tf
│   │   ├── ecs-service/
│   │   │   ├── main.tf
│   │   │   ├── variables.tf
│   │   │   └── outputs.tf
│   │   └── mlflow-server/
│   │       ├── main.tf
│   │       ├── variables.tf
│   │       └── outputs.tf
│   │
│   ├── environments/               # Configuración por ambiente
│   │   ├── dev/
│   │   │   ├── main.tf
│   │   │   ├── variables.tf
│   │   │   ├── terraform.tfvars
│   │   │   └── backend.tf
│   │   ├── staging/
│   │   └── prod/
│   │
│   └── shared/                     # Recursos compartidos
│       └── ecr/
│           └── main.tf
```

13.2 Backend Remoto (State Management)

```
# environments/dev/backend.tf
terraform {
  backend "s3" {
    bucket         = "my-terraform-state"
    key            = "bankchurn/dev/terraform.tfstate"
    region         = "us-east-1"
    encrypt        = true
    dynamodb_table = "terraform-locks"
  }

  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}
```

13.3 Módulo ECS Service

```
# modules/ecs-service/variables.tf
variable "name" {
  description = "Nombre del servicio"
  type        = string
}

variable "container_image" {
  description = "URI de la imagen Docker"
  type        = string
}

variable "container_port" {
  description = "Puerto del contenedor"
  type        = number
  default     = 8000
}

variable "cpu" {
  description = "CPU units"
  type        = number
  default     = 256
}

variable "memory" {
  description = "Memory in MB"
  type        = number
  default     = 512
}

variable "desired_count" {
  description = "Número de tasks"
  type        = number
  default     = 2
}

variable "environment_variables" {
  description = "Variables de entorno"
  type        = map(string)
  default     = {}
}
```

```
# modules/ecs-service/main.tf
resource "aws_ecs_task_definition" "this" {
  family                   = var.name
  network_mode             = "awsvpc"
  requires_compatibilities = ["FARGATE"]
  cpu                      = var.cpu
  memory                   = var.memory
  execution_role_arn       = aws_iam_role.execution.arn
  task_role_arn           = aws_iam_role.task.arn

  container_definitions = jsonencode([
    {
      name      = var.name
      image     = var.container_image
      essential = true

      portMappings = [
        {
          containerPort = var.container_port
          hostPort      = var.container_port
          protocol      = "tcp"
        }
      ]

      environment = [
        for k, v in var.environment_variables : {
          name  = k
          value = v
        }
      ]

      healthCheck = {
        command     = ["CMD-SHELL", "curl -f http://localhost:${var.container_port}/health || exit 1"]
        interval    = 30
        timeout     = 5
        retries     = 3
        startPeriod = 60
      }

      logConfiguration = {
        logDriver = "awslogs"
        options = {
          awslogs-group         = aws_cloudwatch_log_group.this.name
          awslogs-region        = data.aws_region.current.name
          awslogs-stream-prefix = var.name
        }
      }
    }
  ])
}

resource "aws_ecs_service" "this" {
  name            = var.name
  cluster         = var.cluster_id
  task_definition = aws_ecs_task_definition.this.arn
  desired_count   = var.desired_count
  launch_type     = "FARGATE"

  network_configuration {
    subnets          = var.subnet_ids
    security_groups  = [aws_security_group.this.id]
    assign_public_ip = false
  }

  load_balancer {
    target_group_arn = aws_lb_target_group.this.arn
    container_name   = var.name
    container_port   = var.container_port
  }
}
```

13.4 Uso de Módulos

```
# environments/dev/main.tf
provider "aws" {
  region = var.aws_region
}

module "networking" {
  source = "../../modules/networking"

  name        = "bankchurn-${var.environment}"
  cidr_block  = "10.0.0.0/16"
  environment = var.environment
}

module "bankchurn_api" {
  source = "../../modules/ecs-service"

  name            = "bankchurn-api-${var.environment}"
  container_image = "${var.ecr_repo}:${var.image_tag}"
  container_port  = 8000

  cpu    = var.environment == "prod" ? 512 : 256
  memory = var.environment == "prod" ? 1024 : 512

  desired_count = var.environment == "prod" ? 3 : 1

  cluster_id = module.networking.ecs_cluster_id
  subnet_ids = module.networking.private_subnet_ids

  environment_variables = {
    LOG_LEVEL          = var.environment == "prod" ? "INFO" : "DEBUG"
    MLFLOW_TRACKING_URI = module.mlflow.tracking_uri
  }
}

module "mlflow" {
  source = "../../modules/mlflow-server"

  name        = "mlflow-${var.environment}"
  environment = var.environment
  vpc_id      = module.networking.vpc_id
  subnet_ids  = module.networking.private_subnet_ids
}
```

13.5 Workspaces para Multi-Ambiente

```
        # Crear workspaces
        terraform workspace new dev
        terraform workspace new staging
        terraform workspace new prod

        # Cambiar workspace
        terraform workspace select dev

        # Ver workspace actual
        terraform workspace show

        # Listar todos
        terraform workspace list
```

```
# Usar workspace en configuración
locals {
  environment = terraform.workspace

  config = {
    dev = {
      instance_count = 1
      instance_type  = "t3.small"
    }
    staging = {
      instance_count = 2
      instance_type  = "t3.medium"
    }
    prod = {
      instance_count = 3
      instance_type  = "t3.large"
    }
  }
}

resource "aws_instance" "this" {
  count         = local.config[local.environment].instance_count
  instance_type = local.config[local.environment].instance_type
}
```

13.6 Comandos Esenciales

```
# Inicializar
terraform init

# Validar sintaxis
terraform validate

# Plan (ver cambios)
terraform plan -out=tfplan

# Aplicar
terraform apply tfplan

# Destruir
terraform destroy

# Formatear código
terraform fmt -recursive

# Importar recurso existente
terraform import aws_s3_bucket.this my-bucket-name
```

13.7 Ejercicio: Infraestructura BankChurn

Checklist

```
ESTRUCTURA:
[ ] Módulos separados por recurso
[ ] Environments configurados
[ ] Backend remoto (S3 + DynamoDB)

SEGURIDAD:
[ ] No secrets en código
[ ] IAM roles mínimos
[ ] Security groups restrictivos

OPERACIONES:
[ ] terraform plan sin errores
[ ] terraform apply funcional
[ ] Outputs documentados
```

Siguiente Paso

Con infra como código, es hora de orquestar con **Kubernetes**.

Ir a Módulo 14: Kubernetes para ML →

Módulo 13 completado. Tu infraestructura ahora es código.