CHAPTER 3

Best Practices for Reusable Terraform Modules — Google Cloud Documentation

The Case for Modules

Your fare-prediction ML pipeline runs in three environments — dev, staging, and prod. Without a module, you maintain thr

A Terraform module is any directory containing .tf files. The directory where you run terraform apply is the root module

The Four-File Module Layout

Standard Module Structure — Terraform Language prescribes four files for the minimal complete module:
modules/fare-pred-training/
■■■ main.tf # resource definitions
■■■ variables.tf # all input variable declarations
■■■ locals.tf # shared name prefixes and tag maps
■■■ outputs.tf # everything the caller can read
Add iam.tf as a fifth file when IAM resources exceed 150 lines — a threshold from AWS Prescriptive Guidance that keeps p

Authoring Input Variables

Every knob the caller needs to turn is a variable block in variables.tf. The type argument is not optional in a shared m

```
variable "environment" {
type = string
description = "Deployment tier: dev | staging | prod."
validation {
condition = contains(["dev", "staging", "prod"], var.environment)
error_message = "environment must be dev, staging, or prod."
```

Local Values — The Internal DRY Layer

Local values are the module's internal shorthand — defined once, referenced many times, and never overridable by the cal

```
name_prefix = "${var.project_name}-${var.environment}"
common_tags = merge(
{ Project = var.project_name, Environment = var.environment, ManagedBy = "terraform" },
```

local.name_prefix stamps every resource name consistently — SageMaker job, IAM role, CloudWatch log group — without repe

The distinction from input variables is fundamental: if a value should vary across instantiations, use a variable. If a

Declaring and Consuming Module Outputs

Outputs expose module internals to the caller via output blocks in outputs.tf:
output "sagemaker_role_arn" {
description = "ARN of the IAM role used by SageMaker training jobs."
value = aws_iam_role.sagemaker.arn
output "training_job_name_prefix" {
description = "Consistent name prefix applied to all training jobs in this module."
value = local.name_prefix

Calling the Module with Environment-Specific .tfvars

The root main.tf calls the module with a source pointing to its directory:

```
module "training" {
source = "./modules/fare-pred-training"
project_name = var.project_name
environment = var.environment
instance_type = var.training_instance_type
training_image_uri = var.training_image_uri
```

Keeping IAM, Tagging, and Logging Inside the Module Boundary

IAM roles, tagging, and logging configuration belong inside the module, not in the root. Co-locating them with the resou

When IAM resources inside the module exceed 150 lines, break them into a dedicated iam.tf inside the module directory. T

Terraform Validate — Pre-Plan Type Checking

Run terraform validate inside the module directory after every edit:

cd modules/fare-pred-training

terraform init -backend=false

terraform validate

terraform validate checks syntax correctness, attribute names, and type constraints against the provider schema that was

Hands-On Exercise

Task: Extract a minimal SageMaker training module and call it from a root configuration with separate dev and prod .tfva

1. Create modules/fare-pred-training/ with variables.tf, locals.tf, main.tf, outputs.tf, and iam.tf.
2. In variables.tf, declare project_name (required, string), environment (required, string with validation restricting t
3. In locals.tf, define name_prefix = "${var.project_name}-${var.environment}" and a common_tags map merging standard ke
4. In outputs.tf, export sagemaker_role_arn (referencing the IAM role in iam.tf) and training_job_name_prefix (from loca
5. Run terraform init -backend=false && terraform validate inside the module directory — confirm exit code 0.
6. Create envs/dev/terraform.tfvars with instance_type = "ml.m5.xlarge" and envs/prod/terraform.tfvars with instance_typ

Try It Next

Complete the exercises for:

Best Practices for Reusable Terraform Modules — Google Cloud Documentation

academy.kspl.tech