Terraform: Code Modules

- How we can make the Infrastructure Code Reusable?
- Problem with Terraform Config Structure -

```
+-- environments
¦   +-- dev
¦   ¦+-- main.tf
¦   +-- production
¦   ¦+-- main.tf
¦   +-- staging
¦       +-- main.tf
+-- main.tf
+-- provider.tf
```

- How User will add the New Resource Like Elastic Cache in above Structure?

- **Terraform Modules :** Terraform Modules provides re-usable code.
- With Terraform, user can put code inside of a *Terraform module* and reuse that module in multiple places.
- Modules are the key ingredient to writing reusable, maintainable, and testable Terraform code.
- Terraform's way of creating modules is very simple: create a directory that holds a bunch of .tf files.
- Similar to functions in programming languages, module is reusable code that can be invoked multiple times with different inputs.

```
+-- elasticache
¦   +-- main.tf
+-- environments
¦   +-- dev
¦   ¦ +-- main.tf
¦   +-- production
¦   ¦ +-- main.tf
¦   +-- staging
¦       +-- main.tf
+-- main.tf
```

➤ Below Syntax can be used to add elasticache in other Envs.

```
module "dev-elasticache" {
   source = "../../elasticache"
}
```

- **Configurable Terraform Modules :**
- Now that user have our reusable module in place, user will hit another problem: each environment might have its own requirement from a certain resource.
- Eg. In dev we might need just one cache.m3.medium node in our Elasticache cluster, but in production, we might need 3 cache.m3.large nodes in the cluster.
- Above Issue can be solved by making the module configurable using Input varaibles.

```
+-- elasticache
¦   +-- main.tf
¦   +-- variables.tf
+-- environments
¦   +-- dev
¦   ¦ +-- main.tf
¦   +-- production
¦   ¦ +-- main.tf
¦   +-- staging
¦       +-- main.tf
+-- main.tf
```

- variables.tf file will hold the variables that configure the module.

➤ Sample Variable file.

```
variable "environment" {}
variable "node_count" {}
variable "node_type" {}
variable "availability_zones" { type = "list" }
```

➤ Sample Module main.tf file.

```
resource "aws_elasticache_group" "elasticache-cluster" {
  availability_zones          = ["${var.availability_zones}"]
  replication_group_id        = "tf-${var.environment}-group"
  replication_group_description = “${var.environment} group"
  node_type                   = "${var.node_type}"
  number_cache_clusters       = "${var.node_count}"
  parameter_group_name        = “default.redis3.1"
  port                        = 6379
```

- Module call in Dev Env-

```
module "dev-elasticache" {
  source              = "../../elasticache"
  environment         = "dev"
  node_count          = 1
  node_type           = “cache.m3.small"
  availability_zones = ["us-east-1a", "us-east-1b"]
}
```

- Module call in prod Env -

```
module "production-elasticache" {
  source              = "../../elasticache"
  environment         = “prod"
  node_count          = 3
  node_type           = "cache.m3.large"
  availability_zones = ["us-east-1a", "us-east-1b"]
}
```

➤ Source of Modules :

GITHUB

REGISTRY

LOCAL FILE PATH

Will see you in Next Lecture…

See you in next lecture …