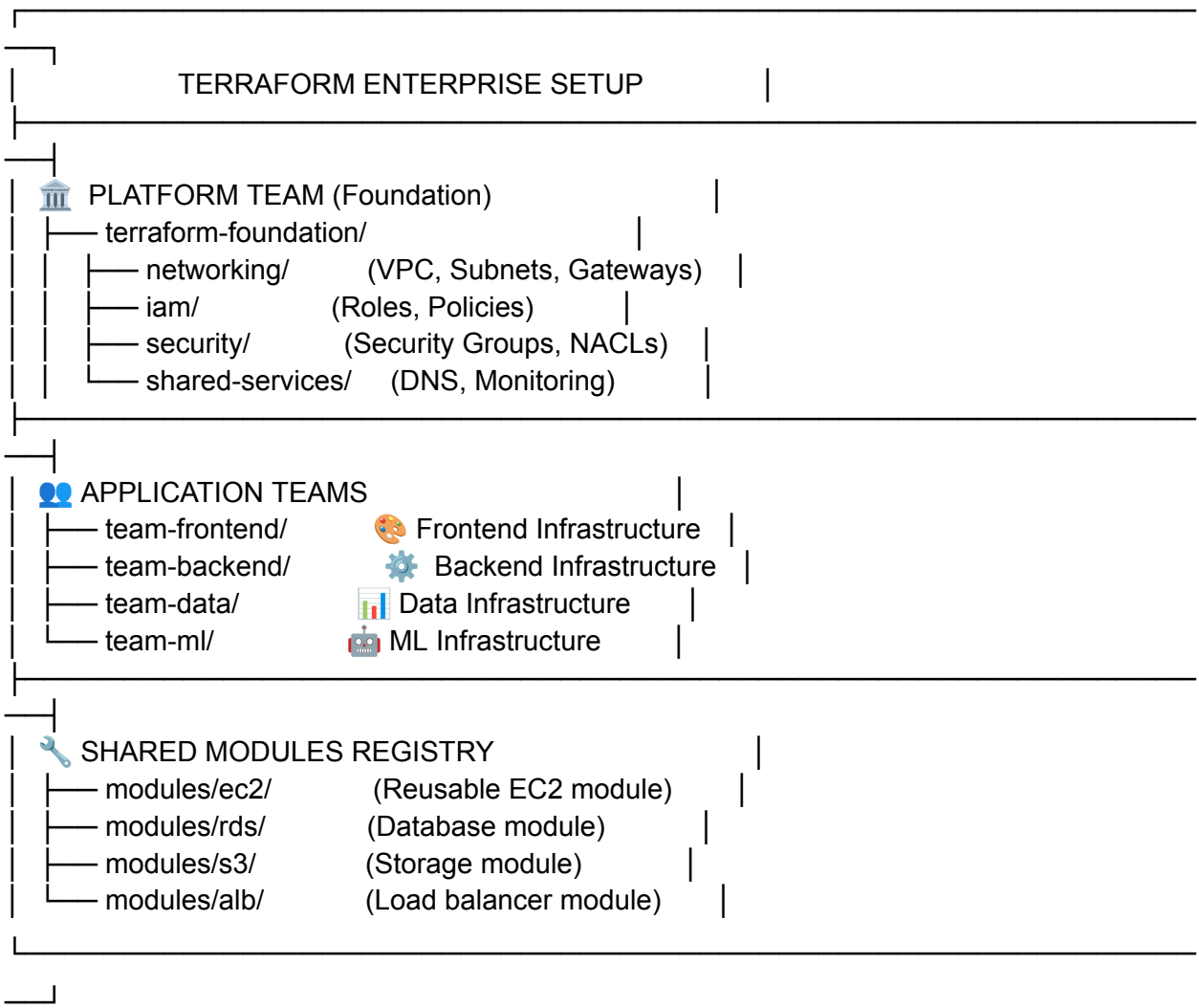
📊 The Architecture Diagram:



🎯 Real-World Example: Netflix/Uber Style Structure

1. Platform Team (The Foundation Builders) 🏗️

hcl

```hcl
# terraform-foundation/networking/main.tf
module "vpc" {
  source = "../../modules/vpc"

  vpc_cidr = "10.0.0.0/16"
```

```
  environment = var.environment

  # Output for other teams to consume
}

# Outputs shared across organization
output "vpc_id" {
  value = module.vpc.vpc_id
}
```

2. Application Teams (The Feature Builders) 👨‍💻

hcl

```
# team-backend/main.tf
data "terraform_remote_state" "foundation" {
  backend = "s3"
  config = {
    bucket = "company-terraform-state"
    key    = "foundation/networking/terraform.tfstate"
  }
}

module "backend_infrastructure" {
  source = "../modules/ec2"

  # Using foundation team's outputs
  vpc_id    = data.terraform_remote_state.foundation.outputs.vpc_id
  subnet_id = data.terraform_remote_state.foundation.outputs.private_subnet_id
}
```

🔑 Key Strategies That Impressed Them:

✅ 1. Remote State Management

- Each team has separate state files
- Foundation team provides "golden" outputs
- No state file conflicts!

✅ 2. Module Registry Pattern

```
modules/
├── ec2/          ← Standard company EC2 setup
├── rds/          ← Pre-configured database
```

```
├── alb/          ← Load balancer with security
└── monitoring/   ← Standard observability
```

✅ 3. Workspace Separation

bash

```
# Platform Team
terraform workspace select foundation-prod
terraform workspace select foundation-staging

# Application Teams
terraform workspace select backend-prod
terraform workspace select frontend-staging
```

✅ 4. CI/CD Pipeline Governance

yaml

```
# .github/workflows/terraform.yml
- name: Terraform Plan
  if: github.event_name == 'pull_request'

- name: Terraform Apply
  if: github.ref == 'refs/heads/main'
  # Only platform team can modify foundation
```

💡 The Secret Sauce:

🔐 RBAC (Role-Based Access Control):

- Platform Team: Full access to foundation
- App Teams: Only their resources + read foundation outputs
- Security Team: Policy enforcement across all

📋 Standards & Governance:

- Mandatory module usage for security compliance
- Standardized tagging and naming conventions
- Automated policy checks with Sentinel/OPA

🎉 Interview Result:

"This is exactly how we structure our Terraform at our organization. You clearly understand enterprise-scale infrastructure!"