Day 13 Internship Report

Learning and Implementing Terraform Modules on Contact Manager

Intern Name:	Arsalan Sharief
Company:	Signiance Technology
Role:	DevOps Intern
Day:	13

Repository Link:

https://github.com/arsalan-signiance/internship-day10-11-12

Objective of the Day

Today’s objective was to understand and implement Terraform modules to refactor the existing infrastructure code of the Contact Manager project into a clean, reusable, and production style modular structure.

The goal was to improve:

- Code organization
- Reusability
- Maintainability
- Scalability
- Separation of concerns

Concepts Learned

1. What are Terraform Modules?

A Terraform module is a container for multiple resources that are used together. Modules allow:

- Grouping related resources
- Reusing infrastructure code
- Maintaining clean structure
- Creating environment-based deployments (dev, prod)

Every Terraform configuration has:

- Root Module: main working directory
- Child Modules: reusable infrastructure components

2. Why Modules Were Needed in This Project

Initially, all infrastructure resources were written in a single directory. As the project grew (VPC, ECS, AMP, SNS, ALB, SSM, etc.), the configuration became complex.

Using modules helped to:

- Separate networking from compute
- Isolate monitoring configuration
- Manage IAM roles independently
- Keep environment-specific values clean

This follows production DevOps best practices.

Modules Created and Their Responsibilities

🔹 VPC Module

- Responsible for: Creating custom VPC, Public & Private Subnets, Internet Gateway, NAT

Gateway, Route Tables, Security Groups.

  - Purpose: Provides secure networking foundation for ECS and ALB.

- ### ECS Module

  - Responsible for: ECS Cluster, Task Definition (API + ADOT sidecar containers), ECS Service, CloudWatch Log Groups.
  - Includes: Fargate configuration, Multi-container task setup, IAM task roles, SSM Parameter integration.

- ### AMP Module

  - Responsible for: Creating AMP workspace, Configuring alert rule groups, Alert Manager integration.
  - Purpose: Enable metrics storage and alert evaluation.

- ### SNS Module

  - Responsible for: SNS Topic, Email subscription, Topic policy for AMP access.
  - Purpose: Send alert notifications via email.

- ### IAM Module

  - Responsible for: ECS Task Execution Role, ECS Task Role, AMP permissions, SSM access permissions.

- Purpose: Secure access between AWS services.

Root Module (Environment Layer)

The `dev` environment acts as the root module. It calls all child modules, passes required variables, defines environment specific values, and manages provider configuration.

Example usage in `main.tf`:

```
module "vpc" {

  source = "../../modules/vpc"

}
```

This separates reusable infrastructure from environment configuration.

Key Terraform Concepts Practiced:

- Variable declaration and passing between modules
- Output values for inter module communication
- Multi container ECS task definition
- Sensitive variables handling

- Terraform init, validate, plan, apply workflow
- Correct HCL block formatting
- Module source referencing
- Environment-based structure

Challenges Faced

Issues Encountered:

- HCL syntax errors (single-line block formatting issues)
- Incorrect use of semicolons and commas
- Variable referencing between modules
- Handling sensitive variables properly

Resolutions:

- Refactoring into multi-line block syntax
- Correct module input/output mapping
- Validating configuration before apply

Result:

- The project was successfully converted into a modular Terraform structure.
- Infrastructure became clean, organized, and reusable.
- The setup now follows real-world DevOps best practices.
- The project is ready for version control and scalable deployments.

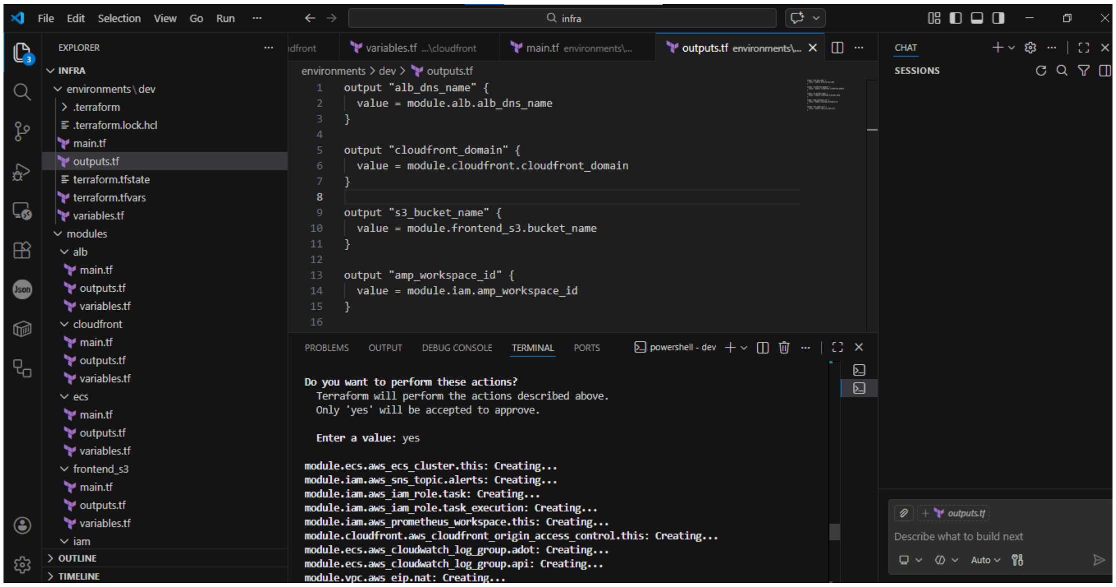
Learning:

Today’s learning improved understanding of:

- Infrastructure as Code design patterns
- Modular architecture in Terraform
- Production-level project structuring
- Clean DevOps workflows

This modular structure makes the project easier to extend, easier to maintain, and suitable for real production environments.

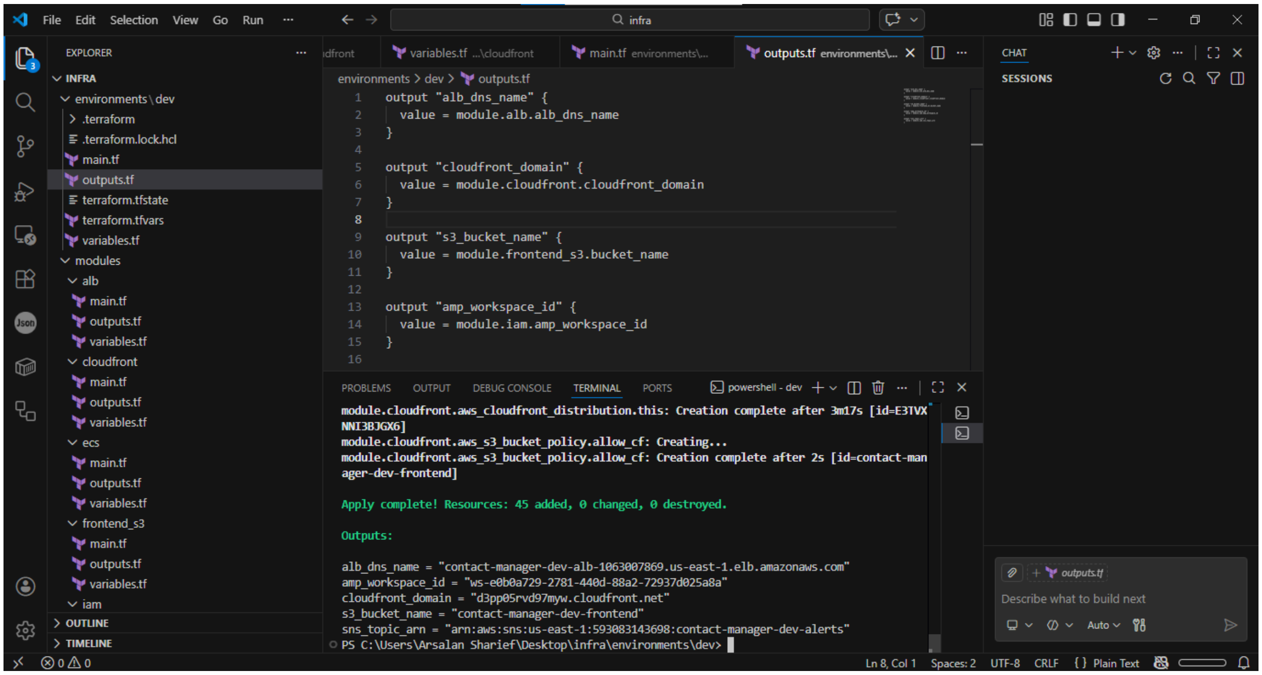
Screenshots:



```
output "alb_dns_name" {
  value = module.alb.alb_dns_name
}

output "cloudfront_domain" {
  value = module.cloudfront.cloudfront_domain
}

output "s3_bucket_name" {
  value = module.frontend_s3.bucket_name
}

output "amp_workspace_id" {
  value = module.iam.amp_workspace_id
}
```

```
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

module.ecs.aws_ecs_cluster.this: Creating...
module.iam.aws_sns_topic.alerts: Creating...
module.iam.aws_iam_role.task: Creating...
module.iam.aws_iam_role.task_execution: Creating...
module.iam.aws_prometheus_workspace.this: Creating...
module.cloudfront.aws_cloudfront_origin_access_control.this: Creating...
module.ecs.aws_cloudwatch_log_group.adot: Creating...
module.ecs.aws_cloudwatch_log_group.api: Creating...
module.vpc.aws_eip.nat: Creating...
```

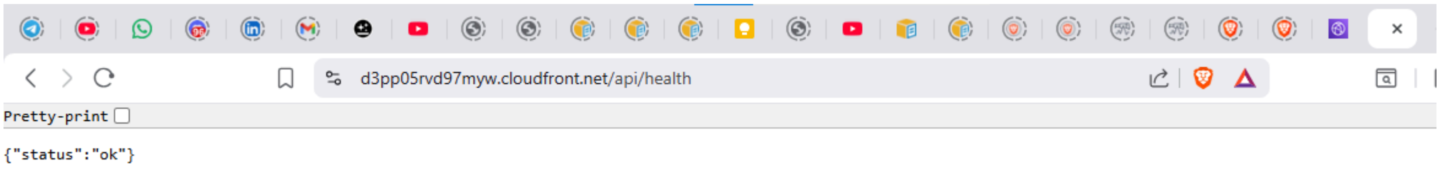
VS Code workspace showing a Terraform project structure with `outputs.tf` open and a terminal running `terraform apply` to provision AWS infrastructure.



```
output "alb_dns_name" {
  value = module.alb.alb_dns_name
}

output "cloudfront_domain" {
  value = module.cloudfront.cloudfront_domain
}

output "s3_bucket_name" {
  value = module.frontend_s3.bucket_name
}

output "amp_workspace_id" {
  value = module.iam.amp_workspace_id
}
```

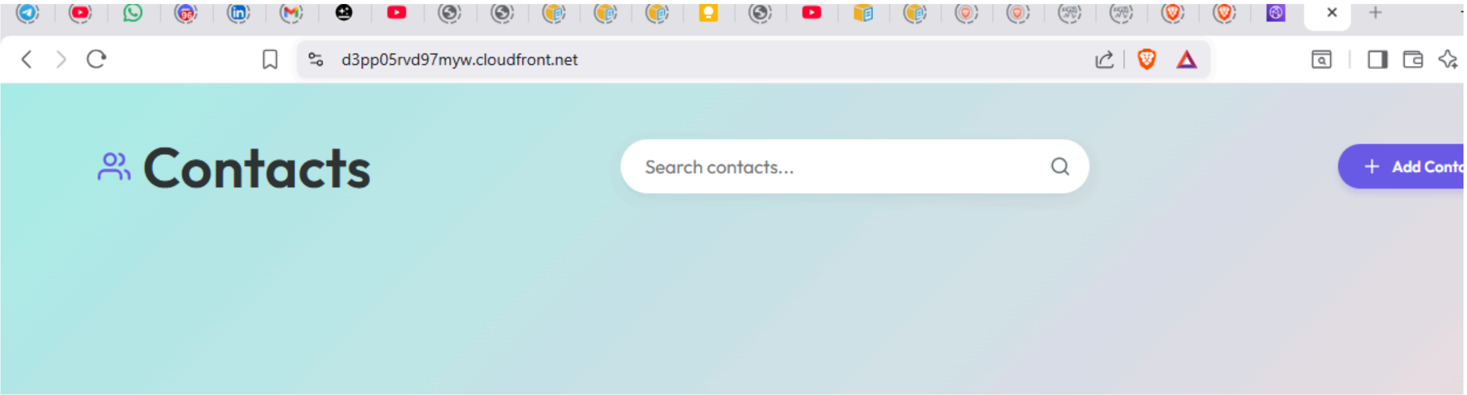
```
module.cloudfront.aws_cloudfront_distribution.this: Creation complete after 3m17s [id=E3TVX
NNI3BJGX6]
module.cloudfront.aws_s3_bucket_policy.allow_cf: Creating...
module.cloudfront.aws_s3_bucket_policy.allow_cf: Creation complete after 2s [id=contact-man
ager-dev-frontend]

Apply complete! Resources: 45 added, 0 changed, 0 destroyed.

Outputs:

alb_dns_name = "contact-manager-dev-alb-1063007869.us-east-1.elb.amazonaws.com"
amp_workspace_id = "ws-e0b0a729-2781-440d-88a2-72937d025a8a"
cloudfront_domain = "d3pp05rvd97myw.cloudfront.net"
s3_bucket_name = "contact-manager-dev-frontend"
sns_topic_arn = "arn:aws:sns:us-east-1:593083143698:contact-manager-dev-alerts"
PS C:\Users\Arsalan Sharief\Desktop\infra\environments\dev>
```

Terminal output confirming successful Terraform deployment with 45 resources added and displaying generated outputs like ALB DNS, CloudFront domain, and S3 bucket name.



```
{"status":"ok"}
```

Browser window displaying a successful API health check at `/api/health` returning JSON response `{"status":"ok"}` via CloudFront.



Web application frontend loaded through CloudFront

Prepared by: ArsalanSharief
DevOps Intern – Signiance Technology