Superb AI

Terraform으로 ECS 기반 API 서버 인프라 구축

Date	Dec 27, 2019
Speaker	Moonsu Cha
	Co-founder, Machine Learning Engineer, Superb AI

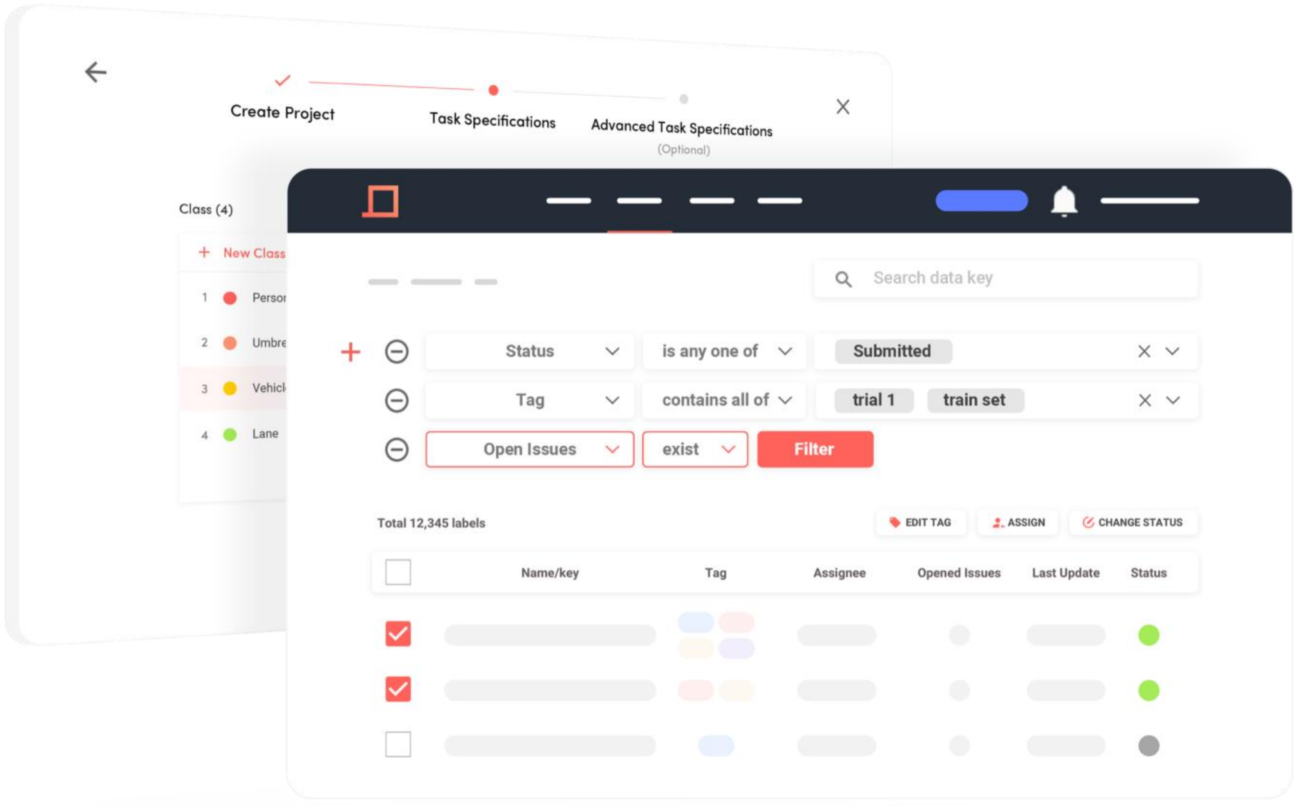
CONTENTS

1. Terraform

2. ECS

3. ECS + Terraform

Superb AI



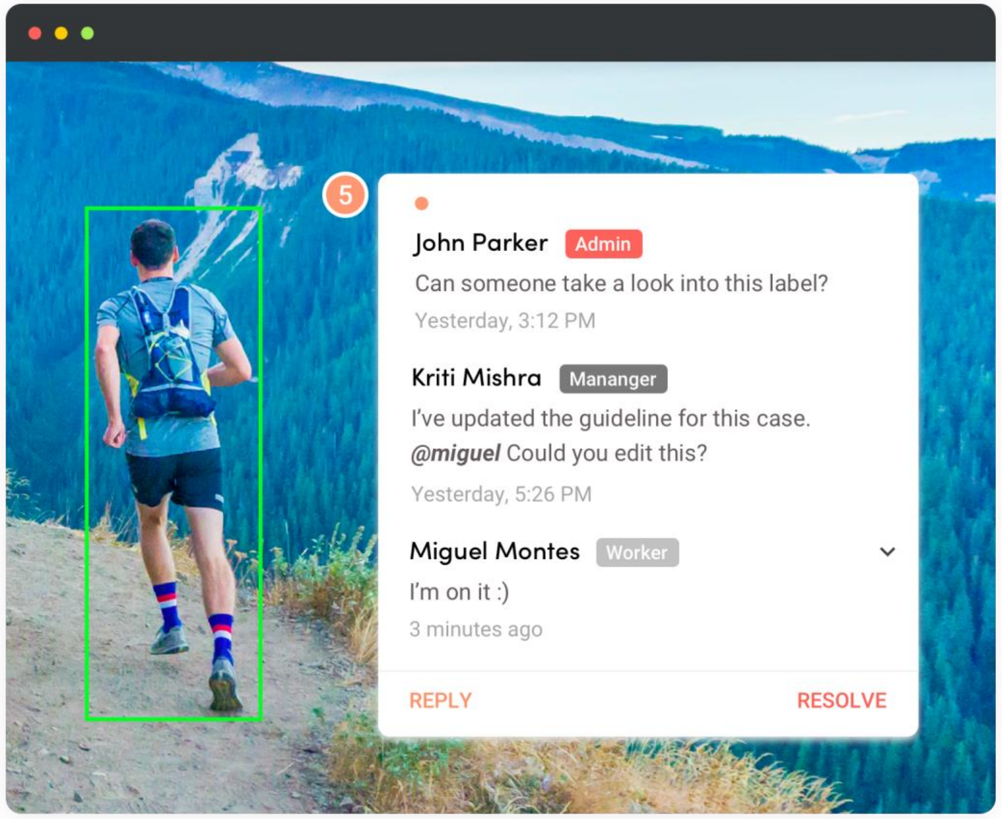
Status — is any one of — Submitted

Tag — contains all of — trial 1, train set

Open Issues — exist — Filter

Total 12,345 labels

	Name/key	Tag	Assignee	Opened Issues	Last Update	Status



5

John Parker Admin
Can someone take a look into this label?
Yesterday, 3:12 PM

Kriti Mishra Mananger
I've updated the guideline for this case. *@miguel* Could you edit this?
Yesterday, 5:26 PM

Miguel Montes Worker
I'm on it :)
3 minutes ago

IaC (Infrastructure as Code)

CHEF

puppet

ANSIBLE

HashiCorp Terraform

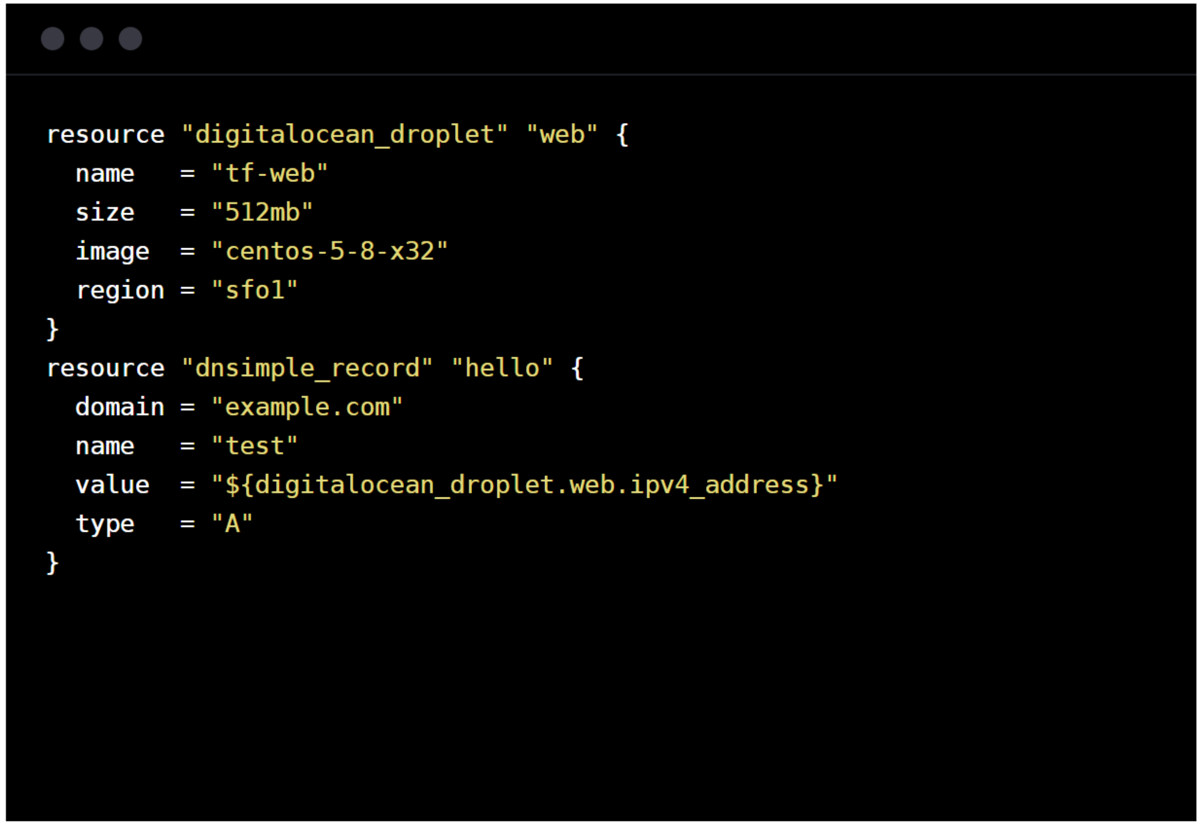
AWS CloudFormation

Deployment Manager (GCP)

aws CDK

HashiCorp Packer

Terraform



```
resource "digitalocean_droplet" "web" {
  name   = "tf-web"
  size   = "512mb"
  image  = "centos-5-8-x32"
  region = "sfo1"
}
resource "dnsimple_record" "hello" {
  domain = "example.com"
  name   = "test"
  value  = "${digitalocean_droplet.web.ipv4_address}"
  type   = "A"
}
```

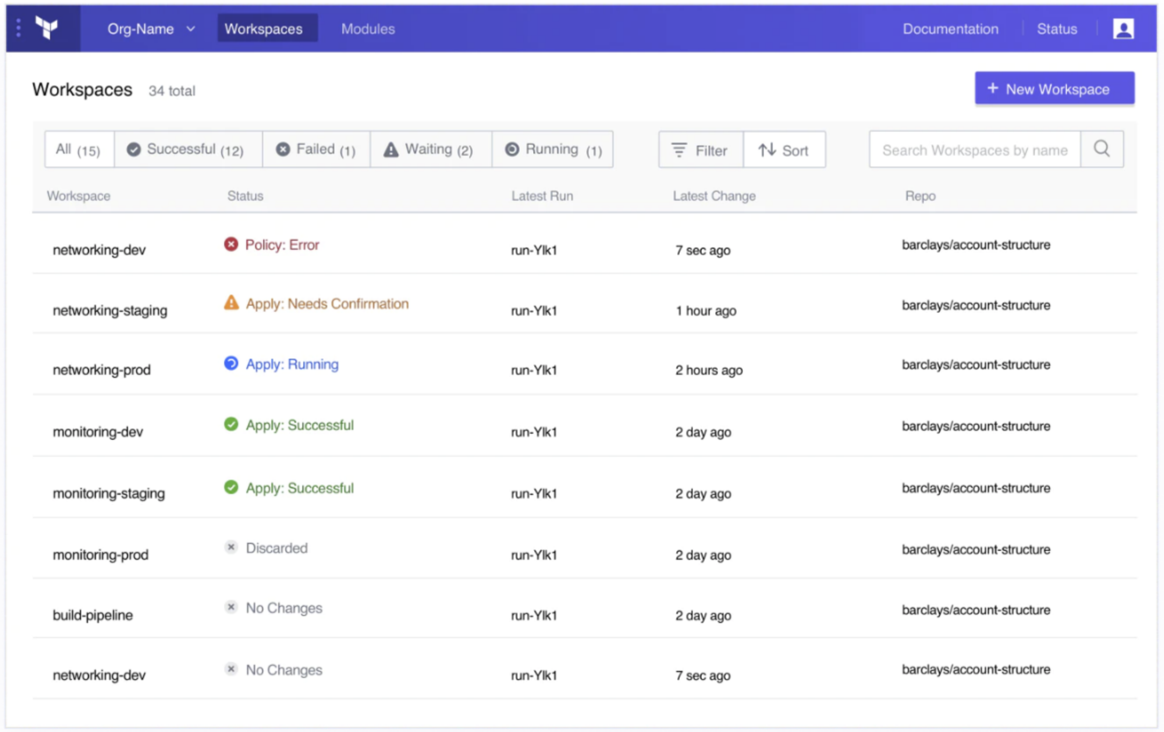
Infrastructure as Code

Codification allows infrastructure changes to be automated, while keeping the definition human readable. Automated tooling allows operators to increase their productivity, move quicker, and reduce human error.

Terraform

Workflows, not Technology

Terraform does not abstract the underlying providers, instead allowing users to leverage the differentiating features with a consistent workflow. As new technologies emerge, they can be adopted without changing the provisioning workflow: plan to preview changes and apply to make changes to any infrastructure globally.



Workspaces 34 total

Workspace	Status	Latest Run	Latest Change	Repo
networking-dev	Policy: Error	run-Ylk1	7 sec ago	barclays/account-structure
networking-staging	Apply: Needs Confirmation	run-Ylk1	1 hour ago	barclays/account-structure
networking-prod	Apply: Running	run-Ylk1	2 hours ago	barclays/account-structure
monitoring-dev	Apply: Successful	run-Ylk1	2 day ago	barclays/account-structure
monitoring-staging	Apply: Successful	run-Ylk1	2 day ago	barclays/account-structure
monitoring-prod	Discarded	run-Ylk1	2 day ago	barclays/account-structure
build-pipeline	No Changes	run-Ylk1	2 day ago	barclays/account-structure
networking-dev	No Changes	run-Ylk1	7 sec ago	barclays/account-structure

Terraform

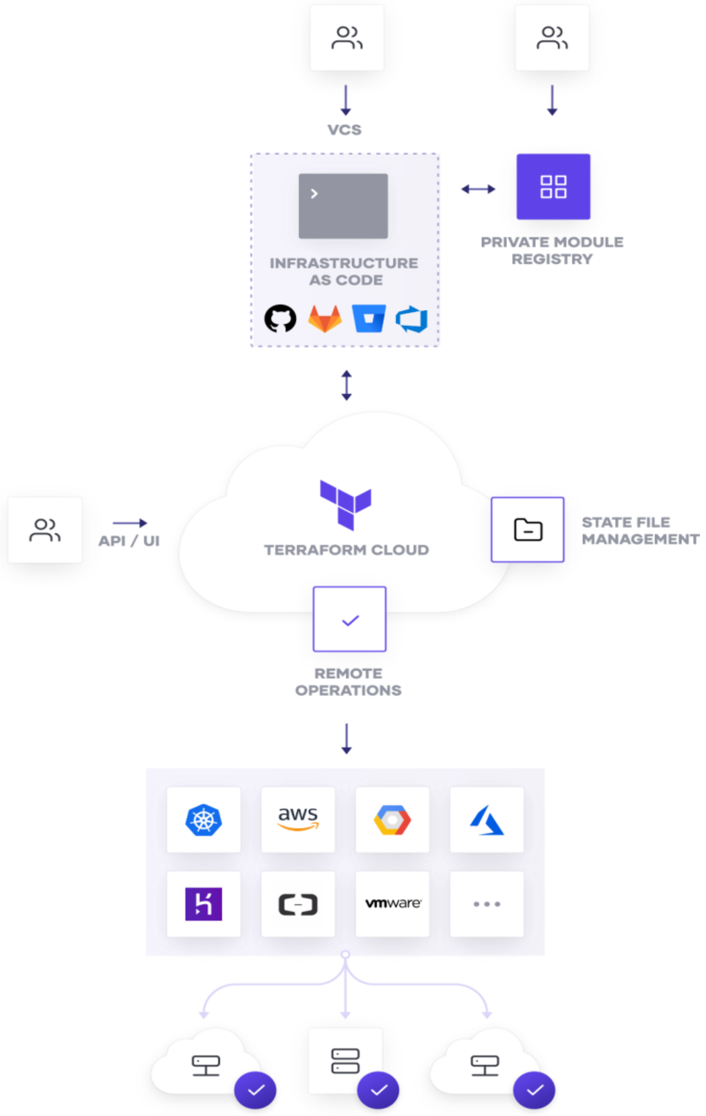
Open and Extensible

Terraform works with over 160 different providers for a broad set of common infrastructure. Provider SDK makes it simple to create new and custom providers.

Providers leverage infrastructure-specific APIs to preserve unique capabilities for each provider.

View All Providers

Terraform

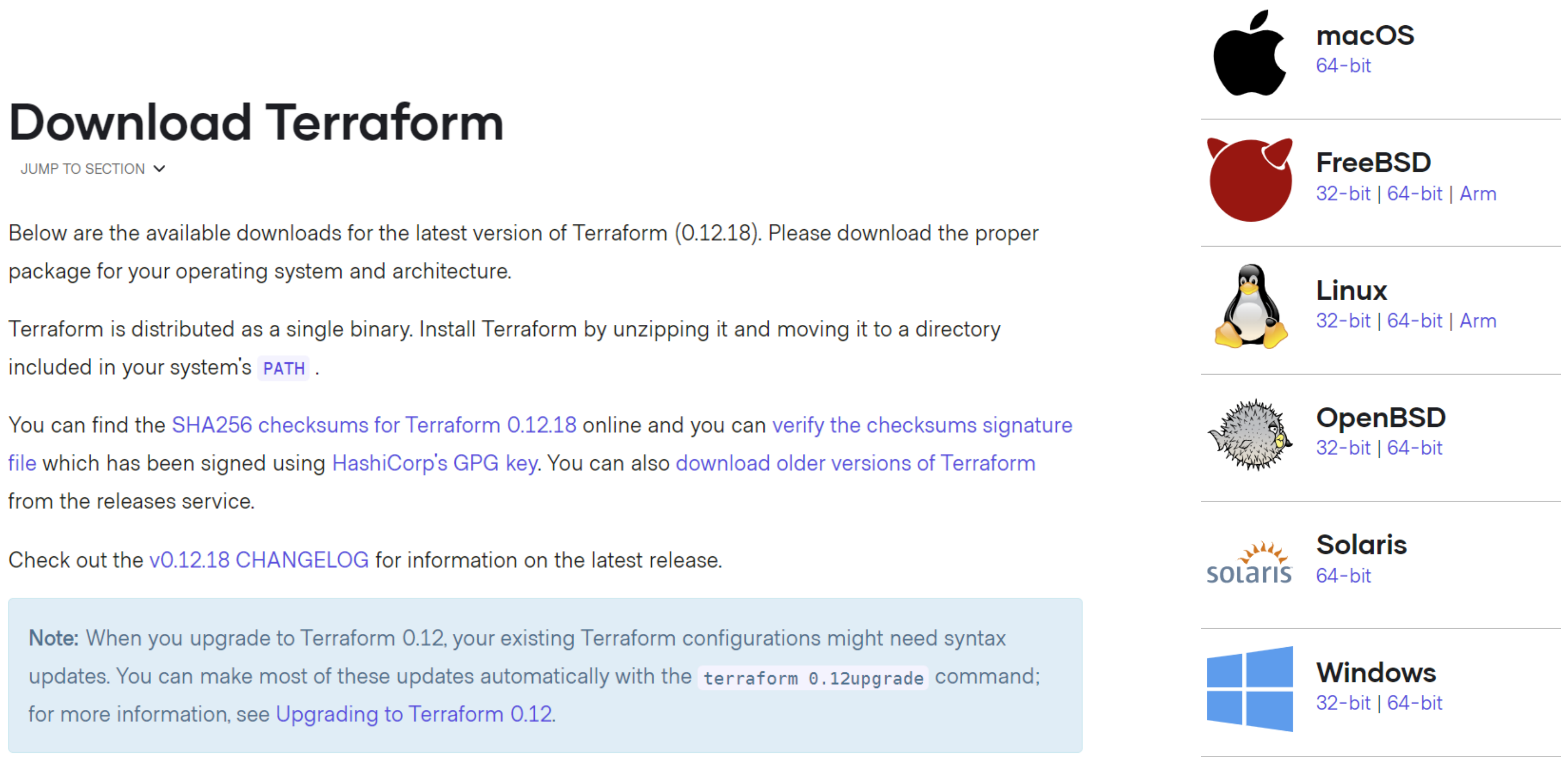


Open Source and Cloud

Terraform allows infrastructure to be expressed as code. The desired state is expressed in a simple human readable language. Terraform uses this language to provide an execution plan of changes, which can be reviewed for safety and then applied to make changes. Extensible providers allow Terraform to manage a broad range of resources, including hardware, IaaS, PaaS, and SaaS services. Terraform Cloud provides free collaboration and automation features as well as paid Team and Governance upgrades.

- ✓ Infrastructure as code
- ✓ 160+ available providers
- ✓ Provision any infrastructure

Install



Download Terraform

JUMP TO SECTION

Below are the available downloads for the latest version of Terraform (0.12.18). Please download the proper package for your operating system and architecture.

Terraform is distributed as a single binary. Install Terraform by unzipping it and moving it to a directory included in your system's `PATH` .

You can find the SHA256 checksums for Terraform 0.12.18 online and you can verify the checksums signature file which has been signed using HashiCorp's GPG key. You can also download older versions of Terraform from the releases service.

Check out the v0.12.18 CHANGELOG for information on the latest release.

Note: When you upgrade to Terraform 0.12, your existing Terraform configurations might need syntax updates. You can make most of these updates automatically with the `terraform 0.12upgrade` command; for more information, see Upgrading to Terraform 0.12.

HCL (Hashicorp Configuration Language)

1. Provider
2. Resource
3. Input Variable
4. Output Variable
5. Local Variable
6. Module
7. Data Source
8. Function
9. State
10. Backend

```
provider "google" {
  project = "acme-app"
  region  = "us-central1"
}
```

```
resource "aws_instance" "web" {
  ami           = "ami-a1b2c3d4"
  instance_type = "t2.micro"
}
```

```
variable "image_id" {
  type = string
}

variable "availability_zone_names" {
  type    = list(string)
  default = ["us-west-1a"]
}
```

```
output "instance_ip_addr" {
  value = aws_instance.server.private_ip
}
```

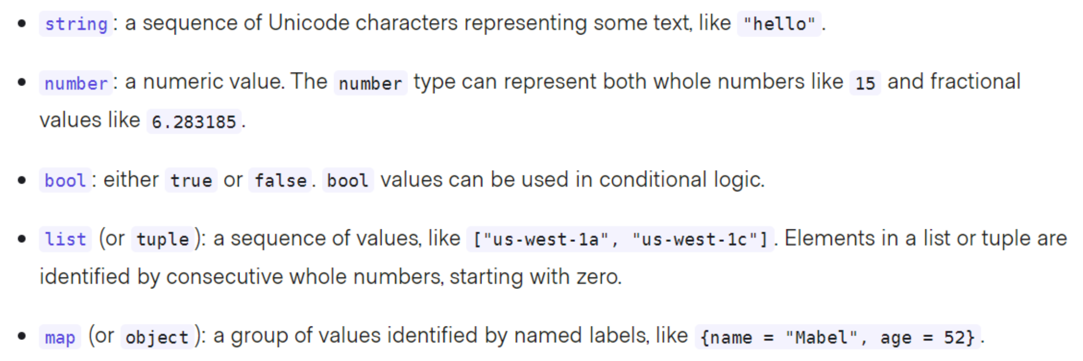
```
locals {
  service_name = "forum"
  owner        = "Community Team"
}
```

```
module "servers" {
  source = "./app-cluster"

  servers = 5
}
```

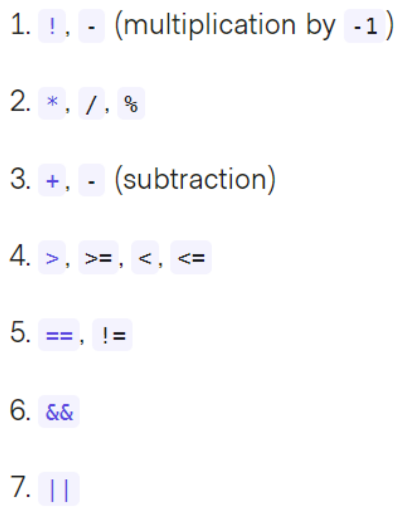
```
data "aws_ami" "example" {
  most_recent = true

  owners = ["self"]
  tags = {
    Name   = "app-server"
    Tested = "true"
  }
}
```

HCL (Hashicorp Configuration Language)

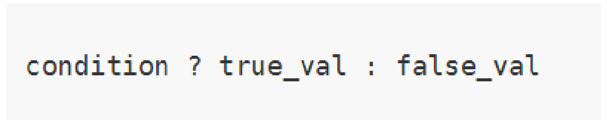
1. Expression
    1. Types and Values
    2. Arithmetic and Logical Operators
    3. Conditional Expressions
    4. For Expressions
    5. Dynamic Block
    6. String Templates
2. Function
    1. Numeric Functions
    2. String Functions
    3. Collection Functions
    4. Encoding Functions
    5. Filesystem Functions
    6. Date and Time Functions
    7. Hash and Crypto Functions
    8. IP network Functions
    9. Type Conversion Functions



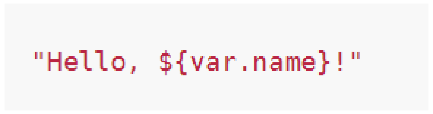
- `string`: a sequence of Unicode characters representing some text, like `"hello"`.
- `number`: a numeric value. The `number` type can represent both whole numbers like `15` and fractional values like `6.283185`.
- `bool`: either `true` or `false`. `bool` values can be used in conditional logic.
- `list` (or `tuple`): a sequence of values, like `["us-west-1a", "us-west-1c"]`. Elements in a list or tuple are identified by consecutive whole numbers, starting with zero.
- `map` (or `object`): a group of values identified by named labels, like `{name = "Mabel", age = 52}`.



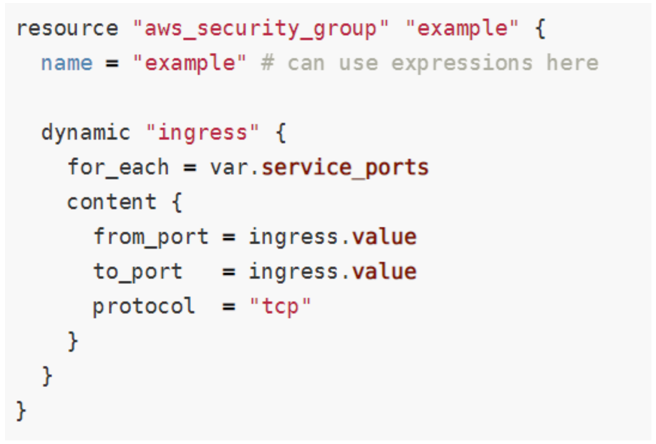
1. `!`, `-` (multiplication by `-1`)
2. `*`, `/`, `%`
3. `+`, `-` (subtraction)
4. `>`, `>=`, `<`, `<=`
5. `==`, `!=`
6. `&&`
7. `||`



```
condition ? true_val : false_val
```

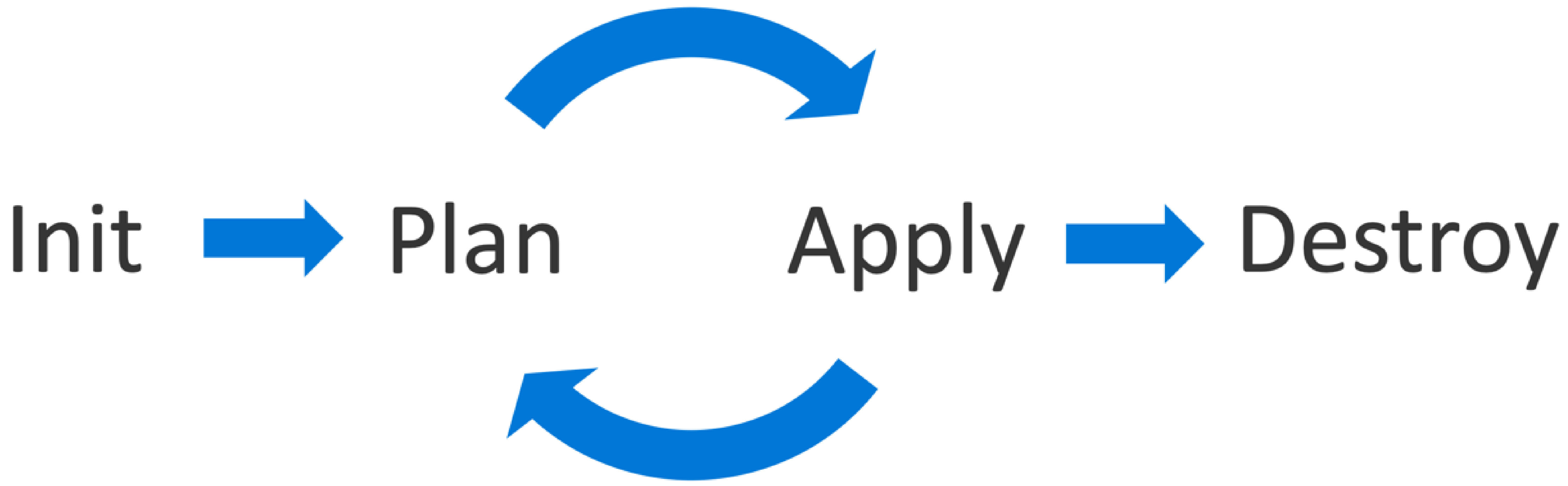


```
"Hello, ${var.name}!"
```



```
resource "aws_security_group" "example" {
  name = "example" # can use expressions here

  dynamic "ingress" {
    for_each = var.service_ports
    content {
      from_port = ingress.value
      to_port   = ingress.value
      protocol  = "tcp"
    }
  }
}
```

Terraform CLI



Remote State

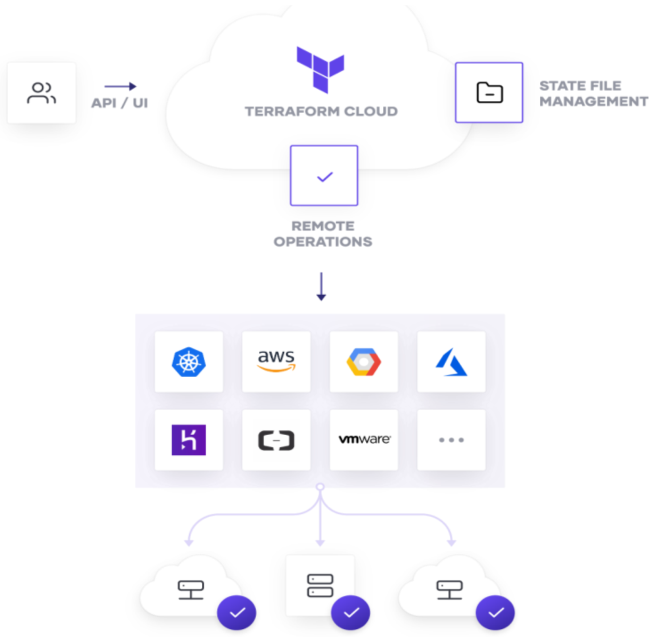
AWS S3 Remote State

```
backend "s3" {
    region         = "us-east-1"
    bucket         = "< the name of the S3 bucket >"
    key            = "terraform.tfstate"
    dynamodb_table = "< the name of the DynamoDB table >"
    encrypt        = true
  }
}
```

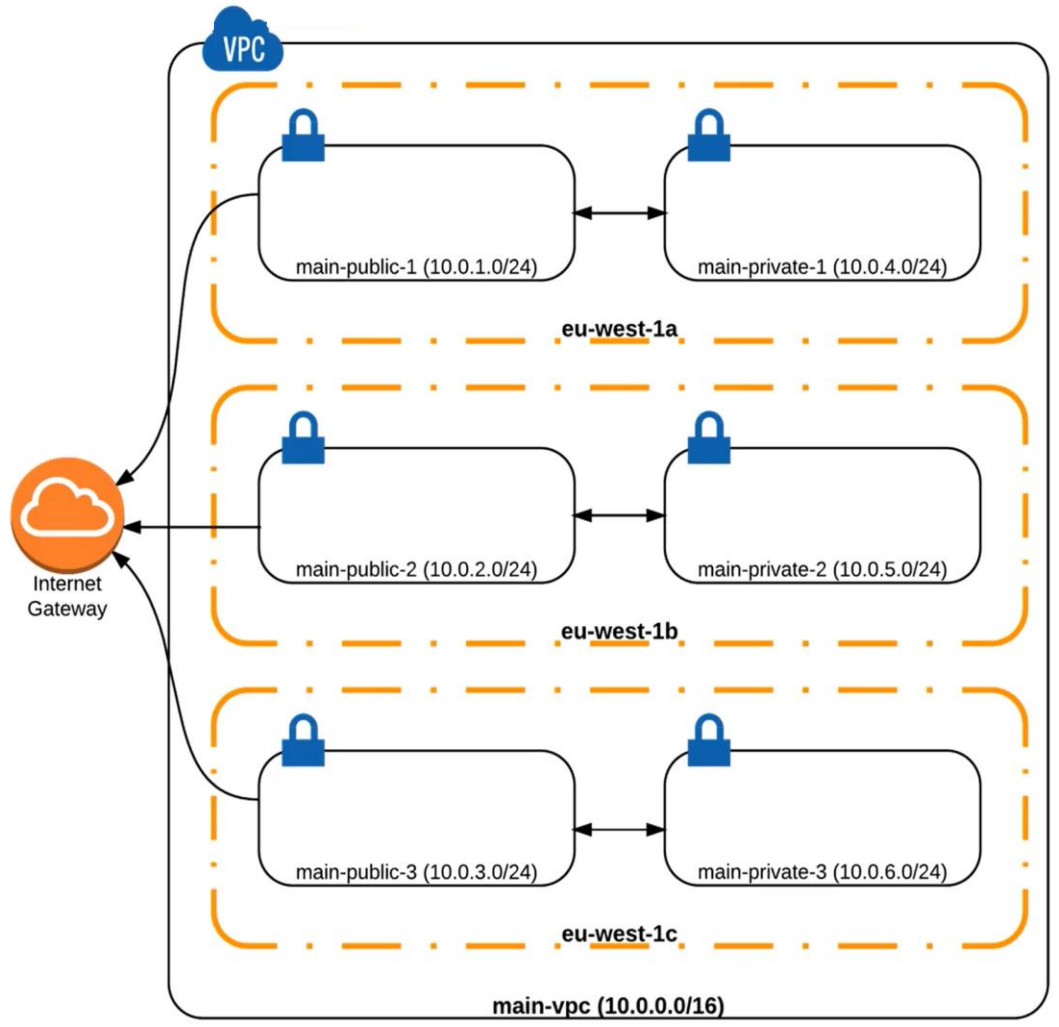
https://github.com/cloudposse/terraform-aws-tfstate-backend

Terraform Cloud

```
terraform {
  backend "remote" {
    organization = "<ORG_NAME>"

    workspaces {
      name = "Example-Workspace"
    }
  }
}
```



VPC Demo


VPC
main-public-1 (10.0.1.0/24)
main-private-1 (10.0.4.0/24)
eu-west-1a
main-public-2 (10.0.2.0/24)
main-private-2 (10.0.5.0/24)
eu-west-1b
main-public-3 (10.0.3.0/24)
main-private-3 (10.0.6.0/24)
eu-west-1c
main-vpc (10.0.0.0/16)
Internet Gateway

```
# Internet VPC
resource "aws_vpc" "main" {
  cidr_block           = "10.0.0.0/16"
  instance_tenancy     = "default"
  enable_dns_support   = "true"
  enable_dns_hostnames = "true"
  enable_classiclink   = "false"
  tags = {
    Name = "main"
  }
}
```

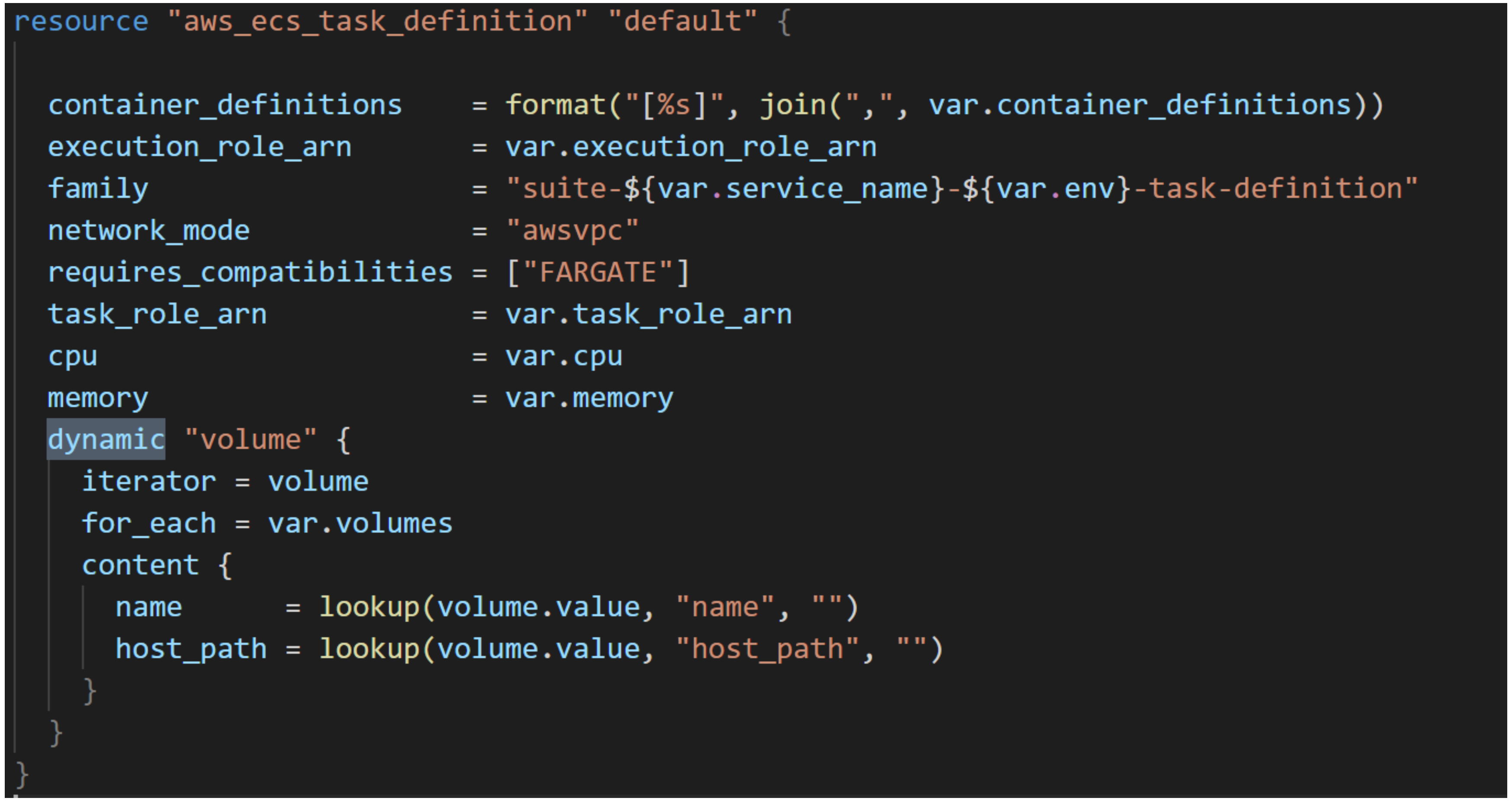
```
# Subnets
resource "aws_subnet" "main-public-1" {
  vpc_id                  = aws_vpc.main.id
  cidr_block              = "10.0.1.0/24"
  map_public_ip_on_launch = "true"
  availability_zone       = "eu-west-1a"

  tags = {
    Name = "main-public-1"
  }
}
```

```
# Internet GW
resource "aws_internet_gateway" "main-gw" {
  vpc_id = aws_vpc.main.id

  tags = {
    Name = "main"
  }
}
```

```
# route tables
resource "aws_route_table" "main-public" {
  vpc_id = aws_vpc.main.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.main-gw.id
  }

  tags = {
    Name = "main-public-1"
  }
}
```

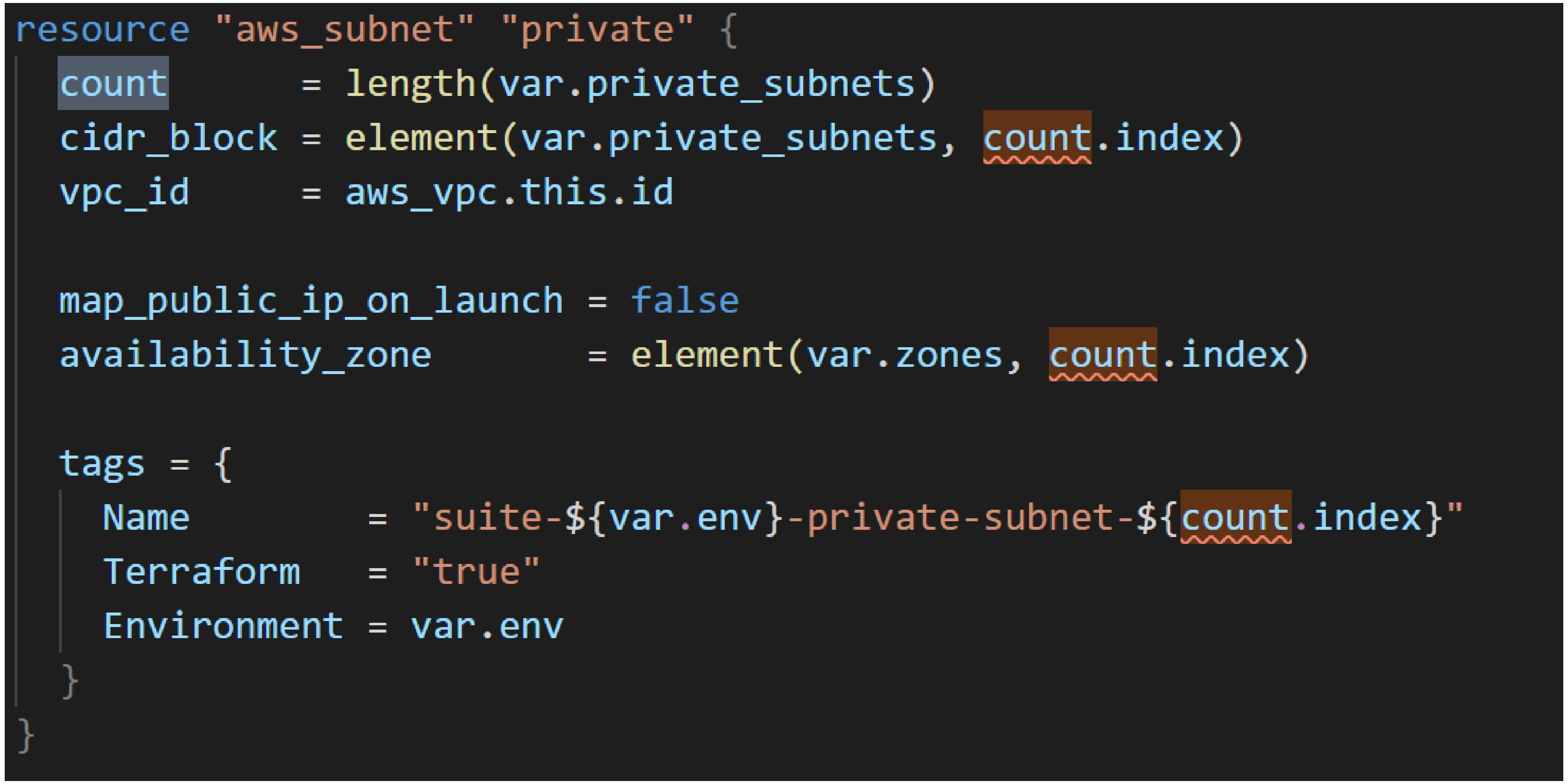
```
# route associations public
resource "aws_route_table_association" "main-public-1-a" {
  subnet_id      = aws_subnet.main-public-1.id
  route_table_id = aws_route_table.main-public.id
}
```

Example



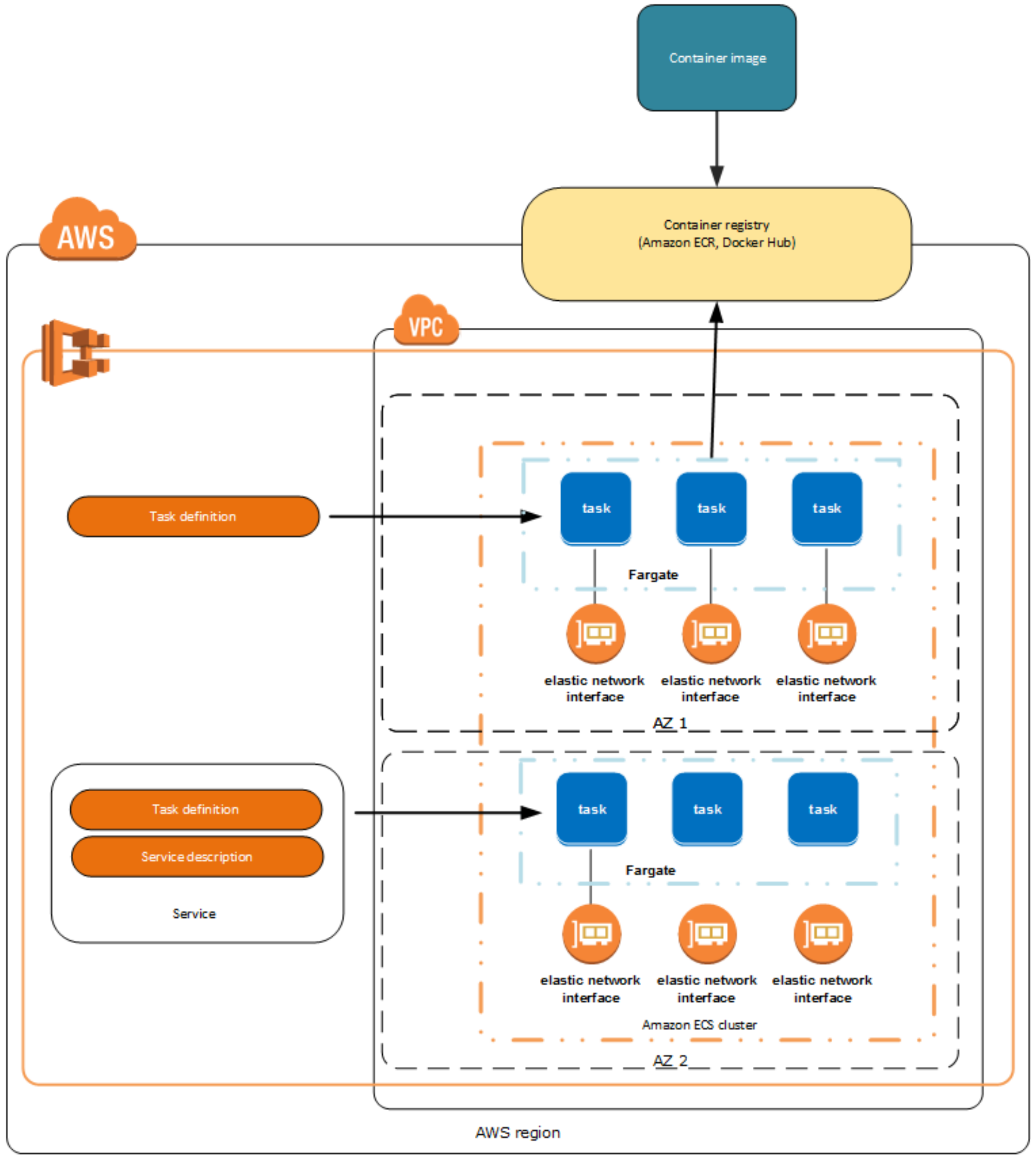
```
resource "aws_ecs_task_definition" "default" {

  container_definitions    = format("[%s]", join(",", var.container_definitions))
  execution_role_arn       = var.execution_role_arn
  family                   = "suite-${var.service_name}-${var.env}-task-definition"
  network_mode             = "awsvpc"
  requires_compatibilities = ["FARGATE"]
  task_role_arn            = var.task_role_arn
  cpu                      = var.cpu
  memory                   = var.memory
  dynamic "volume" {
    iterator = volume
    for_each = var.volumes
    content {
      name      = lookup(volume.value, "name", "")
      host_path = lookup(volume.value, "host_path", "")
    }
  }
}
```

Example



```
resource "aws_subnet" "private" {
  count      = length(var.private_subnets)
  cidr_block = element(var.private_subnets, count.index)
  vpc_id     = aws_vpc.this.id

  map_public_ip_on_launch = false
  availability_zone       = element(var.zones, count.index)

  tags = {
    Name        = "suite-${var.env}-private-subnet-${count.index}"
    Terraform   = "true"
    Environment = var.env
  }
}
```

ECS (Elastic Container Service)



ECR

ECS

EC2

Fargate

Task Definition

Service

Container Definition

Task

Task Definition

- Docker Image
- CPU, Memory
- Launch Type
- Launch Type
- Docker Networking
- Logging Configuration
- Command the Container
- Data Volumes
- IAM Role

Service

Task Definition

Load Balancer

Service Discovery

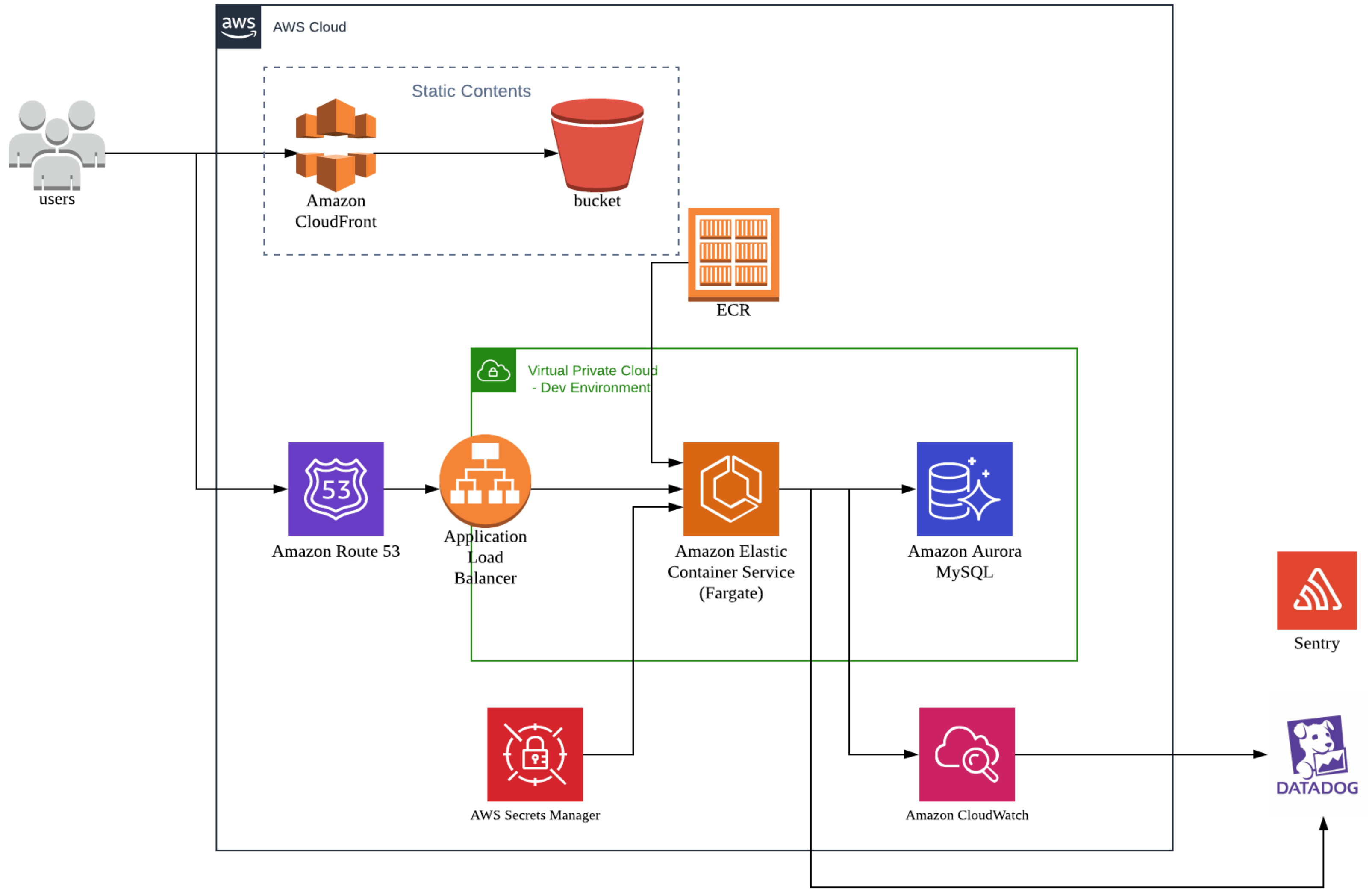
Number of Tasks

Deployment Configure

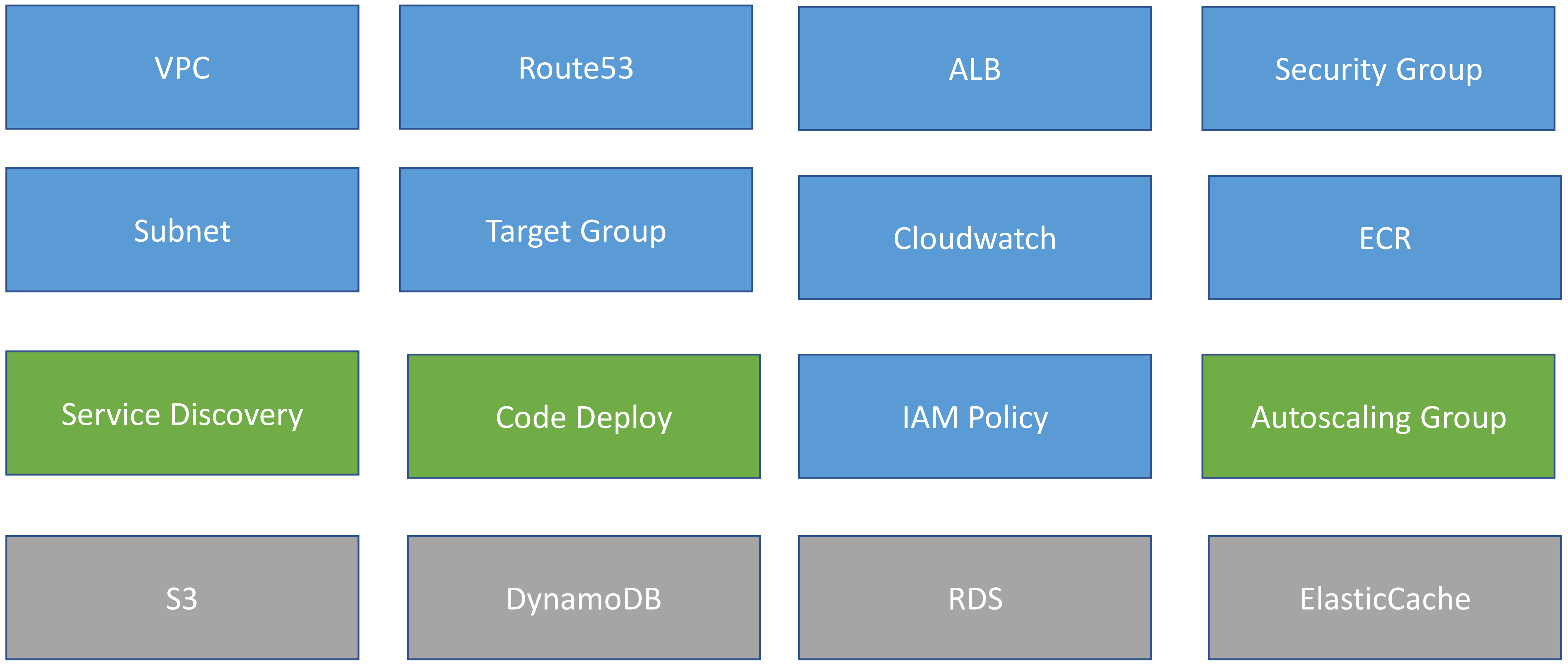
Scheduling Strategy

Network Configure

ECS API Architecture



AWS Resources



AWS Console Demo

Terraform + ECS

1. AWS Architecture 구상하기

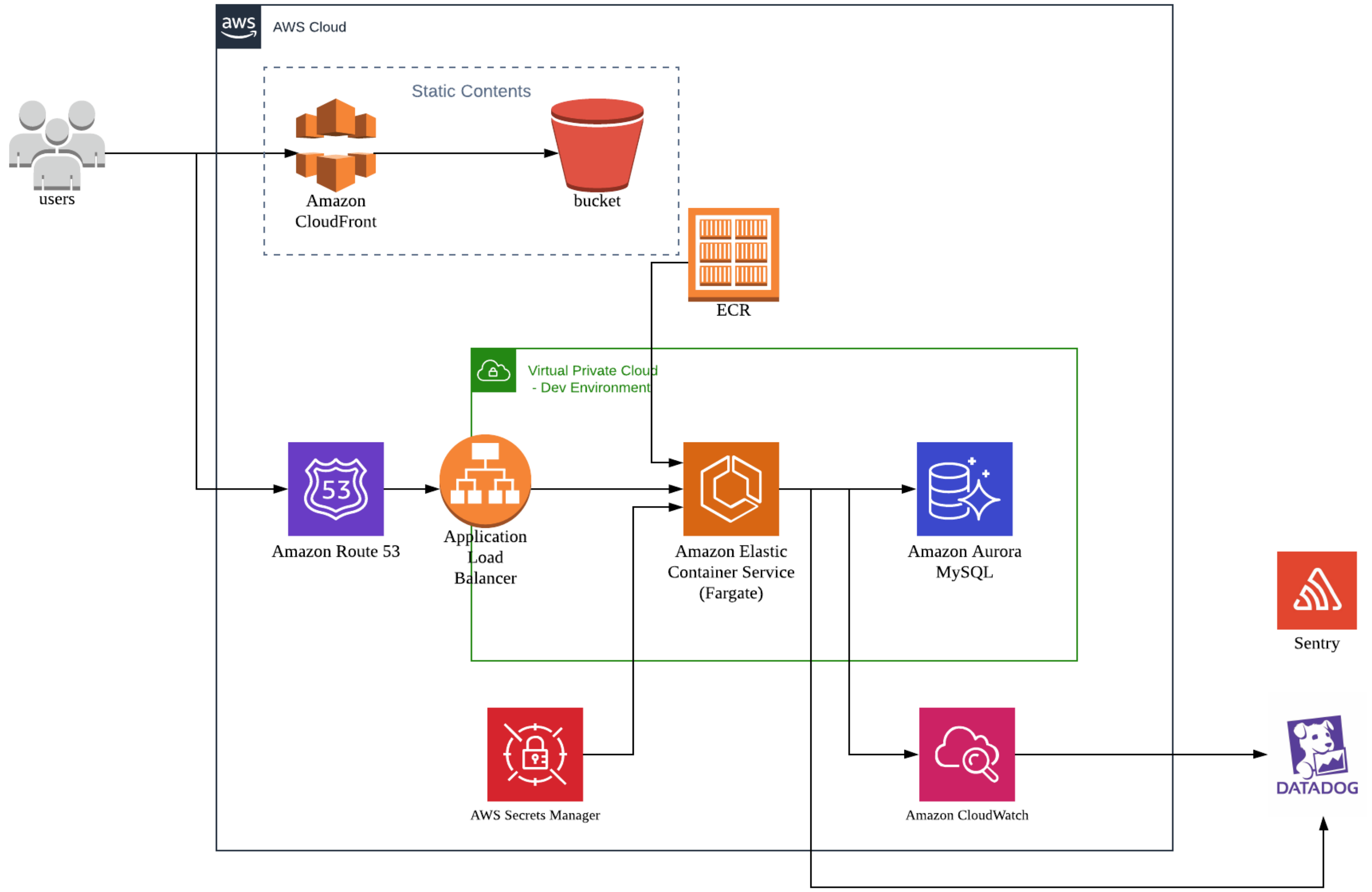
2. AWS Resource 파악 하기

3. Terraform Destroy 단위 정하기

4. Terraform Module 개발

5. Terraform Service 개발

ECS API Architecture



AWS Resource

VPC
Subnet
NAT
Internet Gateway
Routing Table
Network ACL

Application Load Balancer
Listener
Listener Rule
Target Group

Route53
Service Discovery

RDS Cluster
Subnet Group
RDS Cluster Instance
Parameter Group
IAM Policy
Security Group

ECS
ECR
Task Definition
Container Definition
Service
IAM Policy
Security Group
Cloudwatch

Code Build
Code Deploy
WAF
S3
Elastic Cache
SES
Secret Manager

Terraform 설계

1. What is the complexity of your project?
   1. Number of related resources
   2. Number of Terraform providers

2. How often does tour infrastructure change?
   1. From once a month/week/day
   2. To continuously (every time when there is a new commit)

3. Code change initiators?
   1. Only developers can push to infrastructure repository
   2. Everyone can propose change to anything by opening a PR (including automated tasks running on CI server)

4. Which deployment platform or deployment service do you use?
   1. Code Deploy, CircleCI or Kubernetes require slightly different approach

5. How environments are grouped?
   1. By multiple environment, multiple region, multiple project

https://www.terraform-best-practices.com/code-structure

Terraform 설계

Global

Remote State
Cert

Infra

VPC
ECR
ECS
Bastion
Service Discovery

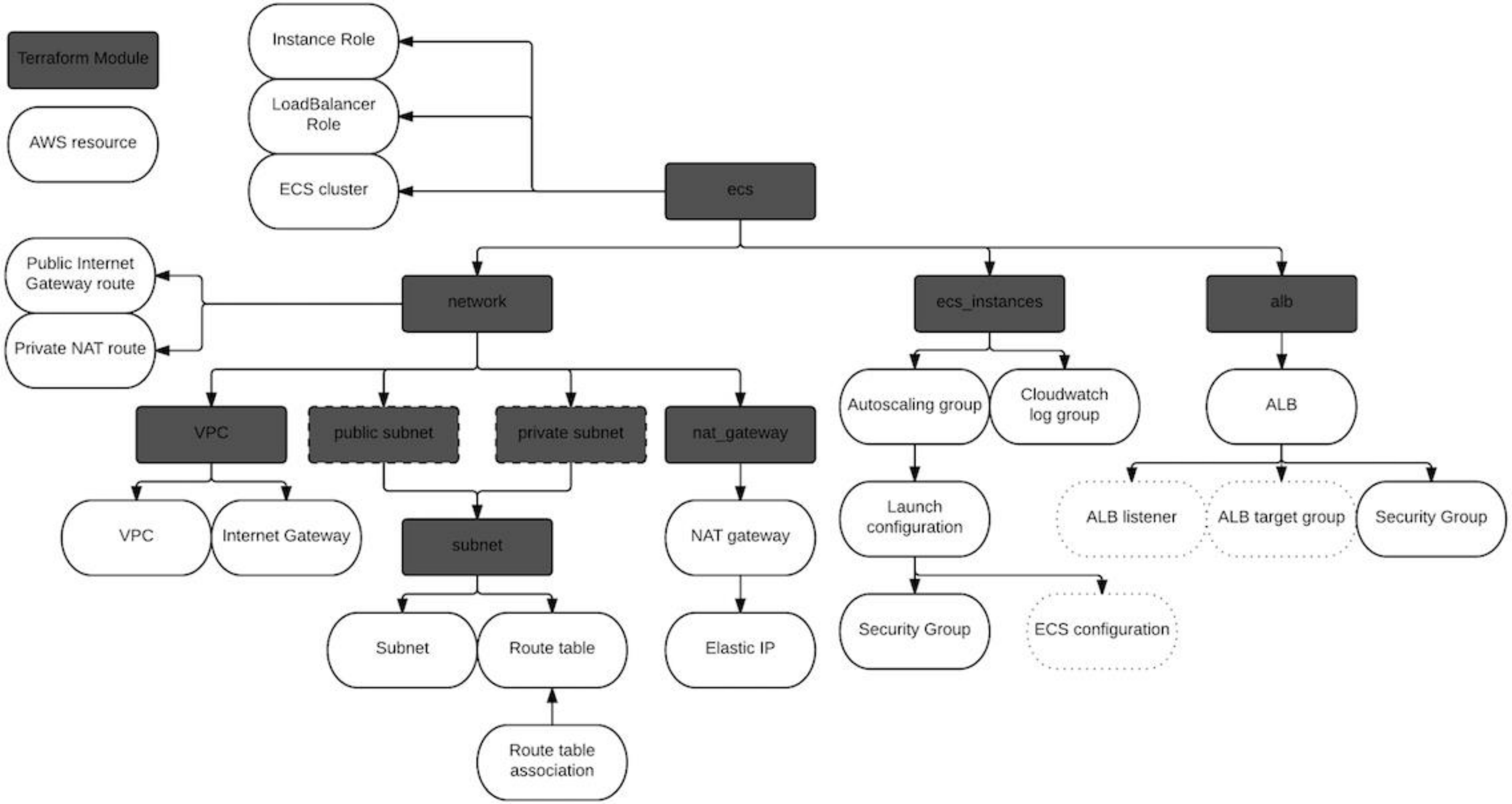
Database

RDS

Service

ALB
Route53
Task Definition
Service
Code Deploy
IAM Policy

Terraform + AWS Resource Modules



CONCLUDING REMARKS

✓ **Terraform** 자체는 어렵지는 않아요!

✓ **ECS**는 생각 보다 다른 서비스들과 많이 연결 되어있어요!

✓ Terraform으로 ECS를 구축해 보아요!

Let's talk with Superb AI!