



QUERO
EDUCATION

DevOps Tools na prática (IAC)

[AWS + Gitlab + terraform]

Porque Infraestrutura como código?



Porque Infraestrutura como código?

Há muito tempo em um **Data Center** distante, um antigo grupo de seres **poderosos** conhecidos como **SysAmins**, criavam e gerenciavam a infraestrutura **manualmente**. Era uma era sombria e medonha se os administradores caíssem no lado escuro (ou seja, tirassem férias).

A boa notícia é:

- Graças à iniciativa **DevOps** agora temos uma maneira melhor de fazer as coisas: **Infrastructure-as-Code (IAC)**



Benefícios da IAC

- Guardar todo o ***código*** da a sua ***infraestrutura*** em um versionador como por exemplo o ***Github/GitLab/Bitbucket***, isso torna mais fácil o Debug de problemas ou rollback se necessário
- Você pode validar a sua infraestrutura com ***code-reviews*** e ***testes automatizados***.
- Você pode ***reaproveitar*** o ***código***



Benefícios da IAC

- ***Automatizar*** todo o processo de provisionamento o que torna tudo mais ***rápido*** e ***resiliente*** do que processos manuais
- Possibilita o ***time de desenvolvimento*** a implementar modificações na infraestrutura de forma ***autônoma***



Tecnologias utilizadas



GitLab

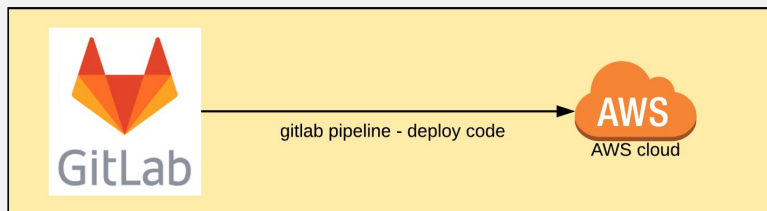
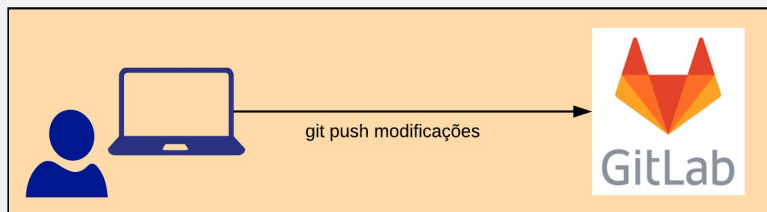
- gitlab (*repositório*)
- gitlab-ci/cd (*pipeline*)



- **Multi Cloud** (AWS, GCP, Azure)
- Escrito em **HCL**



Fluxo de trabalho



Pré-Requisito: Gitlab Pipeline

The screenshot shows the GitLab interface for the 'CI / CD Settings' of a project. The breadcrumb path is 'Fernando Pereira > terraform-tdc-poa > CI / CD Settings'. The left sidebar contains navigation options: Issues (0), Merge Requests (0), CI / CD, Operations, Registry, Packages, Wiki, Snippets, and Settings (selected). Under Settings, 'CI / CD' is highlighted. The main content area is divided into sections: 'General pipelines' (Customize your pipeline configuration, view your pipeline status and coverage report.), 'Auto DevOps' (Auto DevOps will automatically build, test, and deploy your application based on a predefined Continuous Integration and Delivery configuration. [Learn more about Auto DevOps](#)), 'Runners' (Register and see your runners for this project.), and 'Variables' (Variables are applied to environments via the runner. They can be protected by only exposing them to protected branches or tags. You can use variables for passwords, secret keys, or whatever you want.). The 'Variables' section shows two entries: 'AWS_ACCESS_KEY_ID' and 'AWS_SECRET_ACCESS_KEY', both with masked values and 'Protected' status toggles.

GitLab Projects Groups Activity Milestones Snippets

Search or jump to...

Issues 0

Merge Requests 0

CI / CD

Operations

Registry

Packages

Wiki

Snippets

Settings

General

Members

Integrations

Repository

CI / CD

Fernando Pereira > terraform-tdc-poa > CI / CD Settings

General pipelines

Customize your pipeline configuration, view your pipeline status and coverage report.

Auto DevOps

Auto DevOps will automatically build, test, and deploy your application based on a predefined Continuous Integration and Delivery configuration. [Learn more about Auto DevOps](#)

Runners

Register and see your runners for this project.

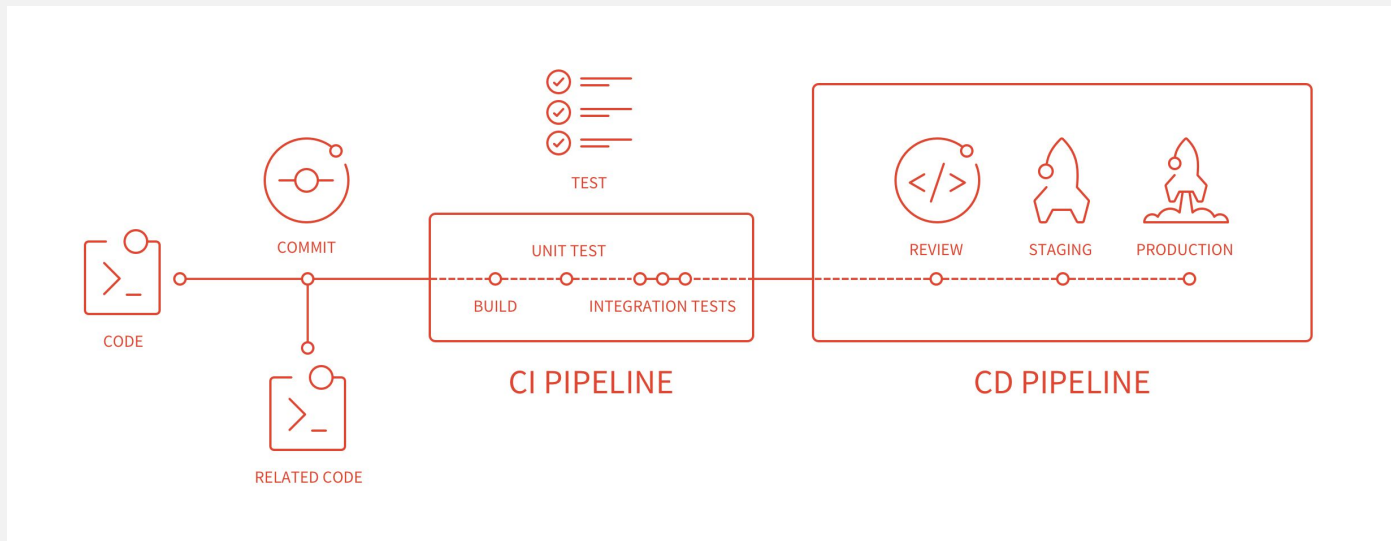
Variables ?

Variables are applied to environments via the runner. They can be protected by only exposing them to protected branches or tags. You can use variables for passwords, secret keys, or whatever you want.

AWS_ACCESS_KEY_ID	*****	Protected	<input checked="" type="checkbox"/>
AWS_SECRET_ACCESS_KEY	*****	Protected	<input checked="" type="checkbox"/>



Fluxo Pipeline



Gitlab Pipeline: .gitlab-ci.yml

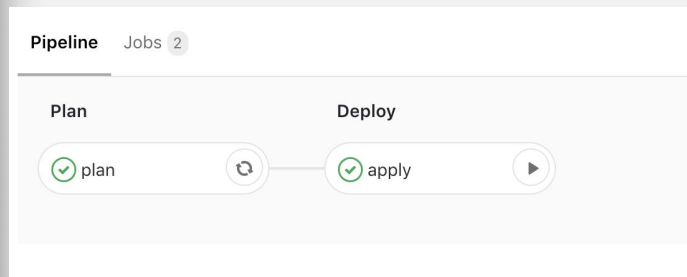
```
2. ubuntu@jp-10-0-1-98: ~ (vim)
image:
  name: queredevops/gitlab-tf-build

before_script:
  - terraform init

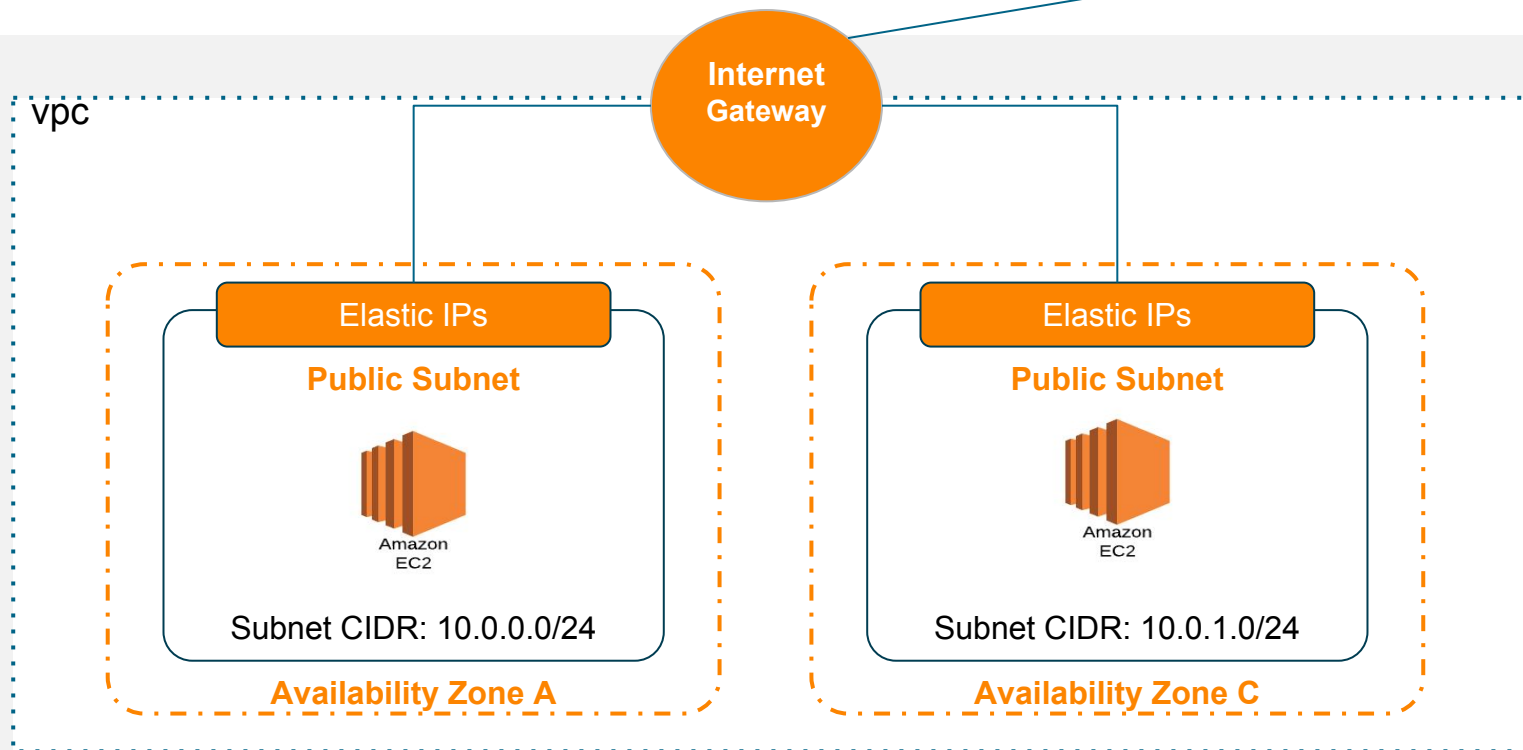
stages:
  - plan
  - deploy

plan:
  stage: plan
  script:
    - terraform init
    - terraform plan

# Separate apply job for manual launching Terraform as it can be destructive
# action.
apply:
  stage: deploy
  environment:
    name: production
  script:
    - terraform init
    - terraform apply -auto-approve
.gitlab-ci.yml" 31L, 480C
```



Arquitectura propuesta



Código: terraform resources

```
2. ubuntu@ip-10-0-1-98: ~ (vim)
resource "aws_instance" "ec2-a" {
  ami           = "ami-0f9cf087c1f27d9b1"
  instance_type = "t3.nano"
  vpc_security_group_ids = ["${aws_security_group.sg_ec2_a.id}"]
  subnet_id     = "${var.subnet_dmz_a}"
  associate_public_ip_address = true
  user_data     = "${data.template_file.ec2_a.rendered}"

  tags {
    Name = "ec2-a"
  }
}

data "template_file" "ec2_a" {
  template = "${file("${path.module}/files/ec2-a.sh")}"
}

resource "aws_security_group" "sg_ec2_a" {
  name        = "sg_ec2_a"
  description = "Permite trafego para ssh na instancia"
  vpc_id      = "${var.vpc_id}"

  ingress {
    from_port = 22
    to_port   = 22
  }
}
```



Código: ec2-userdata

```
1. vim
#!/bin/bash

Sleep 10

#Update system - instala nginx
apt-get update -y
apt-get install nginx -y

#Inicia o nginx
/etc/init.d/nginx start
update-rc.d nginx enable 2

#Faz o deploy da chave ssh para login caso necessario
mkdir -p /home/ubuntu/.ssh

cat << KEYS > /home/ubuntu/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQ3mVywxwTCL3hnrB5x74YCd0ouls0X03VlPPsSAg+i3yVCQuBh3kLDwUL2bj04b0cEs
GS61Ro5Ex1JueXqKDbgIZ54970mKH16GCv8N3+XDrAiqXgVBwDRiabGu0RsjhnRJT5aas0kPc1syVZA1r3YVVxLr+tMvA8U3K/231UYO
NLH09r0VdK5LfUvd9AgxYh9VOMChC7YDh49dzsh5YMKh/Qp22x/4D0Xc6FPz4pY2NMEvwXBrFeLiK4aTJHbYIVm+98Fawss6J0L+Z12qL
+xiPvXcJwh05Haog2Nq24mbG1lcKkzu46F4EhC2YP2Xcfy+dmBrfqKyRi1IU7c0x fernando@fpereira-mac.local
KEYS
~
~
~
~
~
~
```



Código: terraform modules

```
2. ubuntu@ip-10-0-1-98: ~ (vim)
variable "project" {
  default = "terraform-lab"
}

variable "cidr_block" {
  default = "10.0.0.0/16"
}

module "vpc-base" {
  source      = "./modules/vpc-base"
  region     = "us-east-1"
  project_name = "${var.project}"
  cidr_block  = "${var.cidr_block}"
  cidrblock_za = "${cidrsubnet(var.cidr_block, 1, 0)}"
  cidrblock_zc = "${cidrsubnet(var.cidr_block, 1, 1)}"
}

module "arch-instances" {
  source      = "./modules/instances"
  project     = "${var.project}"
  vpc_id     = "${module.vpc-base.vpc_id}"
  subnet_dmz_a = "${module.vpc-base.subnet_dmz_a_id}"
  subnet_dmz_c = "${module.vpc-base.subnet_dmz_c_id}"
}

~
"arch-main-state.tf" 24L, 630C
```



Que os Deuses das apresentações nos abençoem



"I'm glad they introduced cloud computing up here."



Quero Educação

The screenshot shows the top navigation bar with a WhatsApp icon, the phone number 0800 123 2222, the text 'Envie mensagem ou ligue', the 'QUERO BOLSA' logo, and a link 'Entre ou crie sua conta'. Below the navigation bar, there are two tabs: 'Graduação' (selected) and 'Pós-graduação'. The search filters include: 'Qual a sua localização?' with a dropdown menu showing 'Jacareí, SP'; 'Qual curso você está buscando?' with a dropdown menu showing 'Selecione um curso'; 'Até quanto pode pagar?' with a slider set to 'R\$ 10.000/mês'; and 'Prefere alguma faculdade?' with a dropdown menu showing 'Selecione uma faculdade'. There are also checkboxes for 'Presencial' and 'A distância'. A green button at the bottom left says 'Encontrar minha bolsa'. The main content area features a large blue banner with a man pointing, the text 'Encontre aqui a sua bolsa de estudo', and a red banner that says 'MATRÍCULA ANTECIPADA'. Below this, there is a play button icon and the text 'Veja como entrar na faculdade com um super desconto'. At the bottom of the banner, it says 'Daniel, querobolista'. A progress bar at the very bottom shows three steps: '1 ENCONTRE SUA BOLSA DE ESTUDO', '2 FAÇA SUA PRÉ-MATRÍCULA ONLINE', and '3 ECONOMIZE ATÉ O FIM DO CURSO'.

- Marketplace Universidades
- Desenvolve o site querobolsa.com.br
- **+1200** universidades parceiras
- **+300 mil** vagas ativas no site
- **Desde 2012 +420 mil** alunos matriculados
- **+500mil** acessos em 1 dia





Obrigado!

que.bo/vagas-dev

- *Fernando Gonçalves Pereira*

- Head of Infrastructure@*QueroEducação*

LinkedIn: *fernando-pereira-br* | **Github:** *pexaorj*