

|  |  |
| --- | --- |
| **Site Bike** | |
| Curso: | Redes de Computadores |
| Prof.: | Danilo Sibov e Fernando de Oliveira |
| Aluno: | Lucas de Oliveira Rubia |
| **Turma 3RM** | |

## Subir o Site Bike no Ubuntu, Debian e no Docker

1. Script utilizado para o Ubuntu

#!/bin/bash

# Atualizar todos os pacotes do sistema

sudo apt-get update -y

# Instalar o Apache

sudo apt-get install -y apache2

# Habilitar o Apache para iniciar no boot

sudo systemctl enable apache2

# Instalar o Git

sudo apt-get install -y git

# Clonar o repositório Git

git clone https://github.com/FofuxoSibov/sitebike

# Mover os arquivos para o diretório do Apache

sudo mv sitebike/\* /var/www/html/

# Reiniciar o serviço Apache

sudo systemctl restart apache2

* 1. Código utilizado para o Ubuntu

### GRUPO DE SEGURANÇA ####

resource "aws\_security\_group" "Grupo-Sec-Linux" {

name = "Grupo-Sec-Linux"

description = "Libera SSH e HTTP."

vpc\_id = "vpc-07eea5ad6182cf264"

#Liberar porta SSH

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

#Liberar porta HTTP

ingress {

from\_port = 80

to\_port = 80

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

#Liberar porta PING

ingress {

protocol = "icmp"

from\_port = 8

to\_port = 0

cidr\_blocks = ["0.0.0.0/0"]

}

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

}

### GRUPO EC2 ####

data "template\_file" "user\_data" {

template = file("./scripts/site\_bike\_ubuntu.sh")

}

resource "aws\_instance" "Linux" {

ami = "ami-080e1f13689e07408" #Padrão da imagem vinda da AWS

instance\_type = "t2.micro"

subnet\_id = "subnet-07aad13de50b3d76a"

key\_name = "Linux"

associate\_public\_ip\_address = "true"

vpc\_security\_group\_ids = [aws\_security\_group.Grupo-Sec-Linux.id]

user\_data = base64encode(data.template\_file.user\_data.rendered)

tags = {

Name = "Ubuntu-Site"

}

}

output "instance\_public\_ip" {

description = "IP Publico da Instancia EC2"

value = aws\_instance.Linux.public\_ip

}

* 1. Site funcionando



1. Script utilizado para o Debian

#!/bin/bash

# Atualizar todos os pacotes do sistema

sudo apt-get update -y

# Instalar o Apache

sudo apt-get install -y apache2

# Habilitar o Apache para iniciar no boot

sudo systemctl enable apache2

# Instalar o Git

sudo apt-get install -y git

# Clonar o repositório Git

git clone https://github.com/FofuxoSibov/sitebike

# Mover os arquivos para o diretório do Apache

sudo mv sitebike/\* /var/www/html/

# Reiniciar o serviço Apache

sudo systemctl restart apache2

* 1. Código utilizado para o Debian

### GRUPO DE SEGURANÇA ####

resource "aws\_security\_group" "Grupo-Sec-Linux" {

name = "Grupo-Sec-Linux"

description = "Libera SSH e HTTP."

vpc\_id = "vpc-07eea5ad6182cf264"

#Liberar porta SSH

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

#Liberar porta HTTP

ingress {

from\_port = 80

to\_port = 80

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

#Liberar porta PING

ingress {

protocol = "icmp"

from\_port = 8

to\_port = 0

cidr\_blocks = ["0.0.0.0/0"]

}

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

}

### GRUPO EC2 ####

data "template\_file" "user\_data" {

template = file("./scripts/site\_bike\_debian.sh")

}

resource "aws\_instance" "Linux" {

ami = "ami-058bd2d568351da34" #Padrão da imagem vinda da AWS

instance\_type = "t2.micro"

subnet\_id = "subnet-07aad13de50b3d76a"

key\_name = "Linux"

associate\_public\_ip\_address = "true"

vpc\_security\_group\_ids = [aws\_security\_group.Grupo-Sec-Linux.id]

user\_data = base64encode(data.template\_file.user\_data.rendered)

tags = {

Name = "Debian-Site"

}

}

output "instance\_public\_ip" {

description = "IP Publico da Instancia EC2"

value = aws\_instance.Linux.public\_ip

}

2.2 Site funcionando



1. Script utilizado para o Docker

#!/bin/bash

# Atualizar todos os pacotes do sistema

sudo apt-get update -y

# Instalar pré-requisitos

sudo apt-get install -y \

apt-transport-https \

ca-certificates \

curl \

gnupg \

lsb-release

# Adicionar a chave GPG oficial do Docker

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

# Configurar o repositório estável do Docker

echo \

"deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \

$(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

# Atualizar a lista de pacotes

sudo apt-get update -y

# Instalar o Docker

sudo apt-get install -y docker-ce docker-ce-cli containerd.io

# Verificar se o Docker está funcionando

sudo docker run hello-world

# Clonar o repositório do seu site

git clone hhttps://github.com/FofuxoSibov/sitebike

# Construir a imagem Docker

sudo docker build -t meu-site ./sitebike

# Executar o container Docker

sudo docker run -d -p 80:80 meu-site

3.1. Código utilizado para o Docker

### GRUPO DE SEGURANÇA ####

resource "aws\_security\_group" "Grupo-Sec-Linux" {

name = "Grupo-Sec-Linux"

description = "Libera SSH e HTTP."

vpc\_id = "vpc-07eea5ad6182cf264"

#Liberar porta SSH

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

#Liberar porta HTTP

ingress {

from\_port = 80

to\_port = 80

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

#Liberar porta PING

ingress {

protocol = "icmp"

from\_port = 8

to\_port = 0

cidr\_blocks = ["0.0.0.0/0"]

}

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

}

### GRUPO EC2 ####

data "template\_file" "user\_data" {

template = file("./scripts/site\_bike\_debian\_docker.sh")

}

resource "aws\_instance" "Linux" {

ami = "ami-058bd2d568351da34" #Padrão da imagem vinda da AWS

instance\_type = "t2.micro"

subnet\_id = "subnet-07aad13de50b3d76a"

key\_name = "Linux"

associate\_public\_ip\_address = "true"

vpc\_security\_group\_ids = [aws\_security\_group.Grupo-Sec-Linux.id]

user\_data = base64encode(data.template\_file.user\_data.rendered)

tags = {

Name = "Docker-Site"

}

}

output "instance\_public\_ip" {

description = "IP Publico da Instancia EC2"

value = aws\_instance.Linux.public\_ip

}

3.2. Site funcionando

