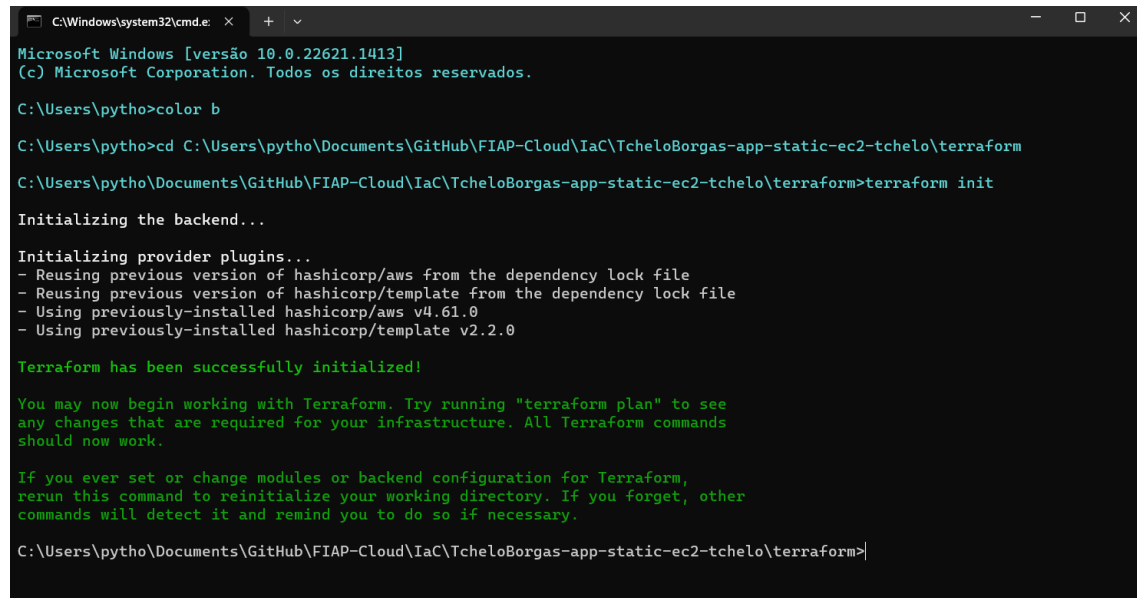


Marcelo Henrique Góes da Costa Borgas

Segue as evidências do CP 02.

As evidências estão no formato de print do prompt e do resultado app static site ec2.

<https://github.com/kledsonhugo/app-static-site-ec2>



```
C:\Windows\system32\cmd.exe X + v
Microsoft Windows [versão 10.0.22621.1413]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\pytho>color b

C:\Users\pytho>cd C:\Users\pytho\Documents\GitHub\FIAP-Cloud\IaC\TcheloBorgas-app-static-ec2-tchelo\terraform

C:\Users\pytho\Documents\GitHub\FIAP-Cloud\IaC\TcheloBorgas-app-static-ec2-tchelo\terraform>terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Reusing previous version of hashicorp/template from the dependency lock file
- Using previously-installed hashicorp/aws v4.61.0
- Using previously-installed hashicorp/template v2.2.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

C:\Users\pytho\Documents\GitHub\FIAP-Cloud\IaC\TcheloBorgas-app-static-ec2-tchelo\terraform>|
```

```
C:\Windows\system32\cmd.exe
Commands will detect it and remind you to do so if necessary.
C:\Users\pytho\Documents\GitHub\FIAP-Cloud\Iac\TcheloBorgas-app-static-ec2-tchelo\terraform>terraform plan -input=false -out tfplan
data.template.file.user_data: Reading...
data.template.file.user_data: Read complete after 0s [id=6411703d585f729c88139d118bfae3f21c710891c55498ce173a6a97b727433f]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  + create

Terraform will perform the following actions:

# aws_instance.instance will be created
+ resource "aws_instance" "instance" {
  + ami              = "ami-02c136e904f3da879"
  + arn              = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone = (known after apply)
  + cpu_core_count   = (known after apply)
  + cpu_threads_per_core = (known after apply)
  + disable_api_stop  = (known after apply)
  + disable_api_termination = (known after apply)
  + ebs_optimized     = (known after apply)
  + get_password_data = false
  + host_id           = (known after apply)
  + host_resource_group_arn = (known after apply)
  + iam_instance_profile = (known after apply)
  + id               = (known after apply)
  + instance_initiated_shutdown_behavior = (known after apply)
  + instance_state    = (known after apply)
  + instance_type     = "t2.micro"
  + ipv6_address_count = (known after apply)
  + ipv6_addresses    = (known after apply)
  + key_name          = (known after apply)
  + monitoring        = (known after apply)
  + outpost_arn       = (known after apply)
  + password_data     = (known after apply)
  + placement_group   = (known after apply)
  + placement_partition_number = (known after apply)
  + primary_network_interface_id = (known after apply)
  + private_dns       = (known after apply)
  + private_ip        = (known after apply)
  + public_dns        = (known after apply)
  + public_ip         = (known after apply)
  + secondary_private_ips = (known after apply)
  + security_groups    = (known after apply)
  + source_dest_check  = true
  + subnet_id         = (known after apply)
  + tags_all          = (known after apply)
  + tenancy            = (known after apply)
}

# aws_internet_gateway.igw will be created
+ resource "aws_internet_gateway" "igw" {
  + arn              = (known after apply)
  + id              = (known after apply)
  + owner_id         = (known after apply)
  + tags_all         = (known after apply)
  + vpc_id           = (known after apply)
}

# aws_route_table.rt_public will be created
+ resource "aws_route_table" "rt_public" {
  + arn              = (known after apply)
  + id              = (known after apply)
  + owner_id         = (known after apply)
  + propagating_vgws = (known after apply)
  + routes           = [
    + {
      + carrier_gateway_id = ""
      + cidr_block         = "0.0.0.0/0"
      + core_network_arn   = ""
      + destination_prefix_list_id = ""
      + egress_only_gateway_id = ""
      + gateway_id         = (known after apply)
      + instance_id        = ""
      + ipv6_cidr_block     = ""
      + local_gateway_id    = ""
      + nat_gateway_id     = ""
      + network_interface_id = ""
      + transit_gateway_id  = ""
      + vpc_endpoint_id    = ""
      + vpc_peering_connection_id = ""
    }
  ]
  + tags_all         = (known after apply)
  + vpc_id           = (known after apply)
}

# aws_route_table_association.rt_public_to_sn_public will be created
+ resource "aws_route_table_association" "rt_public_to_sn_public" {
  + id              = (known after apply)
  + route_table_id  = (known after apply)
  + subnet_id       = (known after apply)
}

# aws_security_group.sg_public will be created
+ resource "aws_security_group" "sg_public" {
  + arn              = (known after apply)
  + description      = "Managed by Terraform"
}
```

```
C:\Windows\system32\cmd.exe
+ [
+   cidr_blocks = [
+     "0.0.0.0/0",
+   ]
+   description = ""
+   from_port = 0
+   ipv6_cidr_blocks = []
+   prefix_list_ids = []
+   protocol = "-1"
+   security_groups = []
+   self = false
+   to_port = 0
+ ],
+ id = (known after apply)
+ ingress = [
+   {
+     cidr_blocks = [
+       "0.0.0.0/0",
+     ]
+     description = ""
+     from_port = 22
+     ipv6_cidr_blocks = []
+     prefix_list_ids = []
+     protocol = "tcp"
+     security_groups = []
+     self = false
+     to_port = 22
+   },
+   {
+     cidr_blocks = [
+       "0.0.0.0/0",
+     ]
+     description = "TCP/80 from All"
+     from_port = 80
+     ipv6_cidr_blocks = []
+     prefix_list_ids = []
+     protocol = "tcp"
+     security_groups = []
+     self = false
+     to_port = 80
+   },
+   {
+     cidr_blocks = [
+       "10.0.0.0/16",
+     ]
+     description = ""
+     from_port = 0
+     ipv6_cidr_blocks = []
+     prefix_list_ids = []
+   }
+ ]
+ }

# aws_vpc.vpc will be created
+ resource "aws_vpc" "vpc" {
+   arn = (known after apply)
+   cidr_block = "10.0.0.0/16"
+   default_network_acl_id = (known after apply)
+   default_route_table_id = (known after apply)
+   default_security_group_id = (known after apply)
+   dhcp_options_id = (known after apply)
+   enable_classiclink = (known after apply)
+   enable_classiclink_dns_support = (known after apply)
+   enable_dns_hostnames = true
+   enable_dns_support = true
+   enable_network_address_usage_metrics = (known after apply)
+   id = (known after apply)
+   instance_tenancy = "default"
+   ipv6_association_id = (known after apply)
+   ipv6_cidr_block = (known after apply)
+   ipv6_cidr_block_network_border_group = (known after apply)
+   main_route_table_id = (known after apply)
+   owner_id = (known after apply)
+   tags_all = (known after apply)
+ }

Plan: 7 to add, 0 to change, 0 to destroy.

Saved the plan to: tfplan

To perform exactly these actions, run the following command to apply:
  terraform apply "tfplan"

C:\Users\pytho\Documents\GitHub\FIAP-Cloud\Iac\TcheloBorgas-app-static-ec2-tchelo\terraform=
```

```
C:\Windows\system32\cmd.exe X + v

+ default_security_group_id = (known after apply)
+ dhcp_options_id = (known after apply)
+ enable_classiclink = (known after apply)
+ enable_classiclink_dns_support = (known after apply)
+ enable_dns_hostnames = true
+ enable_dns_support = true
+ enable_network_address_usage_metrics = (known after apply)
+ id = (known after apply)
+ instance_tenancy = "default"
+ ipv6_association_id = (known after apply)
+ ipv6_cidr_block = (known after apply)
+ ipv6_cidr_block_network_border_group = (known after apply)
+ main_route_table_id = (known after apply)
+ owner_id = (known after apply)
+ tags_all = (known after apply)
}

Plan: 7 to add, 0 to change, 0 to destroy.

Saved the plan to: tfplan

To perform exactly these actions, run the following command to apply:
    terraform apply "tfplan"

C:\Users\pytho\Documents\GitHub\FIAP-Cloud\IaC\TcheloBorgas-app-static-ec2-tchelo\terraform>terraform apply -auto-approve -input=false tfplan
aws_vpc.vpc: Creating...
aws_vpc.vpc: Still creating... [10s elapsed]
aws_vpc.vpc: Creation complete after 14s [id=vpc-09fb8591e95cc79f8]
aws_internet_gateway.igw: Creating...
aws_subnet.sn_public: Creating...
aws_security_group.sg_public: Creating...
aws_internet_gateway.igw: Creation complete after 1s [id=igw-06522baf2caa2d8d1]
aws_route_table.rt_public: Creating...
aws_route_table.rt_public: Creation complete after 2s [id=rtb-05c4bbdaab516aee]
aws_security_group.sg_public: Creation complete after 4s [id=sg-0d2fc0d597bb60bd3]
aws_subnet.sn_public: Still creating... [16s elapsed]
aws_subnet.sn_public: Creation complete after 11s [id=subnet-087b3e9be4db8c3e3]
aws_route_table_association.rt_public_to_sn_public: Creating...
aws_route_table_association.rt_public_to_sn_public: Creation complete after 1s [id=rtbassoc-00230f6a04096bd55]
aws_instance.instance: Still creating... [10s elapsed]
aws_instance.instance: Still creating... [30s elapsed]
aws_instance.instance: Still creating... [38s elapsed]
aws_instance.instance: Creation complete after 34s [id=i-0ba1b7fb1de13c51a]

Apply complete! Resources: 7 added, 0 changed, 0 destroyed.

C:\Users\pytho\Documents\GitHub\FIAP-Cloud\IaC\TcheloBorgas-app-static-ec2-tchelo\terraform>

C:\Windows\system32\cmd.exe X + v

Apply complete! Resources: 7 added, 0 changed, 0 destroyed.

C:\Users\pytho\Documents\GitHub\FIAP-Cloud\IaC\TcheloBorgas-app-static-ec2-tchelo\terraform>terraform show
# data.template.file.user_data:
data "template_file" "user_data" {
  id = "041793d585f729c08139d110bfae321c714093c55498ce173a6a97b727433f"
  rendered = <--EOT
  #!/bin/bash

  echo "Update with latest packages"
  yum update -y

  echo "Install Apache"
  yum install -y httpd git

  echo "Enable Apache service to start after reboot"
  sudo systemctl enable httpd

  echo "Install application"
  cd /tmp
  git clone https://github.com/TcheloBorgas/TcheloBorgas-app-static-ec2-tchelo.git
  cp /tmp/TcheloBorgas-app-static-ec2-tchelo/app/*.html /var/www/html/

  echo "Start Apache service"
  service httpd restart
EOT
template = <--EOT
#!/bin/bash

echo "Update with latest packages"
yum update -y

echo "Install Apache"
yum install -y httpd git

echo "Enable Apache service to start after reboot"
sudo systemctl enable httpd

echo "Install application"
cd /tmp
git clone https://github.com/TcheloBorgas/TcheloBorgas-app-static-ec2-tchelo.git
cp /tmp/TcheloBorgas-app-static-ec2-tchelo/app/*.html /var/www/html/

echo "Start Apache service"
service httpd restart
EOT
}
# aws_instance.instance:
```

```
C:\Windows\system32\cmd.exe
# aws_instance.instance:
resource "aws_instance" "instance" {
  ami           = "ami-82e136e90af3da870"
  arn           = "arn:aws:ec2:us-east-1:489467430147:instance/i-8ba1b7fb1de13c518"
  associate_public_ip_address = true
  availability_zone = "us-east-1a"
  cpu_core_count = 1
  cpu_threads_per_core = 1
  disable_api_stop = false
  disable_api_termination = false
  ebs_optimized = false
  get_password_data = false
  hibernation = false
  id           = "i-8ba1b7fb1de13c518"
  instance_initiated_shutdown_behavior = "stop"
  instance_state = "running"
  instance_type = "t2.micro"
  ipv6_address_count = 0
  ipv6_addresses = []
  monitoring = false
  placement_partition_number = 0
  primary_network_interface_id = "eni-8f41a9613a120b7e5"
  private_dns = "ip-18-8-1-87.ec2.internal"
  private_ip = "18.8.1.87"
  public_dns = "ec2-3-83-243-187.compute-1.amazonaws.com"
  public_ip = "3.83.243.187"
  secondary_private_ips = []
  security_groups = []
  source_dest_check = true
  subnet_id = "subnet-887b3e9be4db8c3e3"
  tags_all = {}
  tenancy = "default"
  user_data = "A35ad895b1f36ffed5af969a5616348f74be0c48"
  user_data_replace_on_change = false
  vpc_security_group_ids = [
    "sg-8d2fc8d597bb6bd3",
  ]
}

capacity_reservation_specification {
  capacity_reservation_preference = "open"
}

credit_specification {
  cpu_credits = "standard"
}

enclave_options {
  enabled = false
}

enclave_options {
  enabled = false
}

maintenance_options {
  auto_recovery = "default"
}

metadata_options {
  http_endpoint = "enabled"
  http_put_response_hop_limit = 1
  http_tokens = "optional"
  instance_metadata_tags = "disabled"
}

private_dns_name_options {
  enable_resource_name_dns_a_record = false
  enable_resource_name_dns_aaaa_record = false
  hostname_type = "ip-name"
}

root_block_device {
  delete_on_termination = true
  device_name = "/dev/xvda"
  encrypted = false
  iops = 100
  tags = {}
  throughput = 0
  volume_id = "vol-84c8fed9bdf4e5be"
  volume_size = 8
  volume_type = "gp2"
}

# aws_internet_gateway.igw:
resource "aws_internet_gateway" "igw" {
  arn = "arn:aws:ec2:us-east-1:489467430147:internet-gateway/igw-86522ba72caa2dbd1"
  id = "igw-86522ba72caa2dbd1"
  owner_id = "489467430147"
  tags_all = {}
  vpc_id = "vpc-09fb8591095cc79fa"
}

# aws_route_table.rt_public:
resource "aws_route_table" "rt_public" {
  arn = "arn:aws:ec2:us-east-1:489467430147:route-table/rtb-85c4b8daab516aee"
  id = "rtb-85c4b8daab516aee"
  owner_id = "489467430147"
  propagating_vgwes = []
  route = [

```

```
C:\Windows\system32\cmd.exe X + v
destination_prefix_list_id = ""
egress_only_gateway_id = ""
gateway_id = ""
instance_id = "igw-06522baf2caa2dbd1"
instance_ip = ""
ipv4_cidr_block = ""
local_gateway_id = ""
nat_gateway_id = ""
network_interface_id = ""
transit_gateway_id = ""
vpc_endpoint_id = ""
vpc_peering_connection_id = ""
},
],
tags_all = {}
vpc_id = "vpc-09fb8591895cc79f8"
}

# aws_route_table_association rt_public_to_sn_public:
resource "aws_route_table_association" "rt_public_to_sn_public" {
  id = "rtbassoc-00238f6a84u96bd55"
  route_table_id = "rtb-05c4b0b0aabb15aee"
  subnet_id = "subnet-087b3e9be4db8c3e3"
}

# aws_security_group sg_public:
resource "aws_security_group" "sg_public" {
  arn = "arn:aws:ec2:us-east-1:489467438147:security-group/sg-0d2fc0d597bb6bd3"
  description = "Managed by Terraform"
  egress = [
    {
      cidr_blocks = [
        "0.0.0.0/0",
      ]
      description = ""
      from_port = 0
      ipv4_cidr_blocks = []
      prefix_list_ids = []
      protocol = "-1"
      security_groups = []
      self = false
      to_port = 0
    },
  ],
  id = "sg-0d2fc0d597bb6bd3"
  ingress = [
    {
      cidr_blocks = [
        "0.0.0.0/0",
      ]
      description = ""
    },
  ],
  name = "sg_public"
  owner_id = "489467438147"
  revoke_rules_on_delete = false
  tags_all = {}
  vpc_id = "vpc-09fb8591895cc79f8"
}

# aws_subnet sn_public:
resource "aws_subnet" "sn_public" {
  arn = "arn:aws:ec2:us-east-1:489467438147:subnet/subnet-087b3e9be4db8c3e3"
  assign_ipv6_address_on_creation = false
  availability_zone = "us-east-1a"
  availability_zone_id = "us-east-1a"
  cidr_block = "10.0.0.0/24"
  enable_dns64 = false
  enable_resource_name_dns_a_record_on_launch = false
  enable_resource_name_dns_aaaa_record_on_launch = false
  id = "subnet-087b3e9be4db8c3e3"
  ipv6_native = false
  map_customer_owned_ip_on_launch = false
  map_public_ip_on_launch = true
  owner_id = "489467438147"
  private_dns_hostnames_type_on_launch = "ip-name"
  tags_all = {}
  vpc_id = "vpc-09fb8591895cc79f8"
}

# aws_vpc vpc:
resource "aws_vpc" "vpc" {
  arn = "arn:aws:ec2:us-east-1:489467438147:vpc/vpc-09fb8591895cc79f8"
  assign_generated_ipv6_cidr_block = false
  cidr_block = "10.0.0.0/16"
}
```

