

Desafío # 7

Fecha de entrega: 18/12/2024

Objetivo:

Configurar un sitio web estático utilizando un bucket de Amazon S3 o una instancia de Amazon EC2 como origen y distribuir el contenido a través de Amazon CloudFront, aplicando los conceptos aprendidos de Terraform e incorporando mejores prácticas de IaC.

Escenario:

Tu equipo está en un proceso de modernización y desea implementar una infraestructura que facilite la creación de un sitio web estático y su distribución global. Para ello, debes realizar una prueba de concepto que demuestre cómo aprovisionar un sitio estático utilizando AWS.

El objetivo es configurar una infraestructura que pueda utilizar Amazon S3 o Amazon EC2 como origen para los archivos estáticos, vinculándola a una distribución de CloudFront para garantizar una entrega rápida, segura y global.

Requisitos:

Configuración del entorno:

- Configura credenciales para acceder a la sandbox de AWS Academy o usa una cuenta personal de AWS (asegurándote de destruir los recursos al final).

Infraestructura a implementar:

El alumno debe elegir entre dos opciones de origen para el contenido:

Opción 1: Bucket de S3

- Configura un bucket de S3 para servir un sitio web estático.
- Configura permisos para que CloudFront pueda acceder directamente al bucket.
- Asegúrate de que el bucket sea privado y que únicamente CloudFront tenga acceso a su contenido, utilizando una Origin Access Control (OAC) en CloudFront.

Opción 2: Instancia EC2

- Configura una instancia EC2 con Nginx para servir un sitio web estático.
- Configura un Security Group para permitir únicamente tráfico HTTP desde cualquier lugar (puerto 80) y SSH restringido a tu IP (puerto 22).
- Vincula la instancia EC2 a una distribución de CloudFront como origen personalizado, asegurándote de configurar los protocolos adecuados (TLS y HTTP).
- Utiliza la Key Pair provista por AWS Academy llamada `vokey`
- Utiliza un perfil de instancia llamado **LabInstanceProfile** con un Role previamente creado llamado **LabRole** para poder acceder con Session Manager (revisar README de AWS Academy)

Pruebas:

Proporciona evidencia del sitio en funcionamiento:

- Capturas de pantalla del sitio web funcionando.
- URL generada por **CloudFront**.

Documentación:

1. Documenta el proceso de implementación en un **README**.
2. Explica los pasos para configurar la infraestructura, las configuraciones clave de S3 o EC2 y CloudFront, y las decisiones técnicas tomadas.
3. Incluye un **diagrama de alto nivel** basado en los gráficos adjuntos como referencia:
 - a. **Diagrama 1:** S3 como origen.
 - b. **Diagrama 2:** EC2 como origen.

Importante: si por algún motivo decides utilizar tu cuenta de AWS personal al finalizar el trabajo destruye todos los recursos. Recuerda que dejar los recursos corriendo puede incurrir en costos que luego te pueden ser cobrados.

Entregables:

Los entregables establecidos para este proyecto son:

1. Código fuente del proyecto alojado en un repositorio público o privado (con acceso compartido para la evaluación).
2. Documentación clara, detallada y bien redactada.
3. Evidencia del sitio web en funcionamiento (URL o capturas de pantalla).

Evaluación:

1. **Entrega en la fecha indicada:** El proyecto debe entregarse puntualmente.
2. **Código funcional y sin errores:** Terraform debe desplegar la infraestructura correctamente.
3. **Documentación completa:** La documentación debe ser comprensible y detallar cada paso realizado.
4. **Infraestructura funcional:**
 - a. En el caso de S3, el contenido debe estar disponible a través de **CloudFront**.
 - b. En el caso de EC2, la instancia debe servir contenido estático correctamente a través de **CloudFront**.
5. **Cumplimiento de las consignas planteadas:** Los recursos creados deben reflejar las opciones seleccionadas (S3 o EC2) y seguir las mejores prácticas.

Documentos de referencia:

- [Host a Static Website on AWS with S3 and CloudFront](#)
- [Terraform AWS Provider Documentation](#)

- [Restrict access to an Amazon Simple Storage Service origin](#)

Nota sobre los Diagramas

- **Diagrama 1 (S3 como origen):** Representa la infraestructura cuando se usa Amazon S3 como origen para el contenido estático.
- **Diagrama 2 (EC2 como origen):** Representa la infraestructura cuando se usa una instancia EC2 como origen para el contenido estático.

Diagrama con S3 como Origen

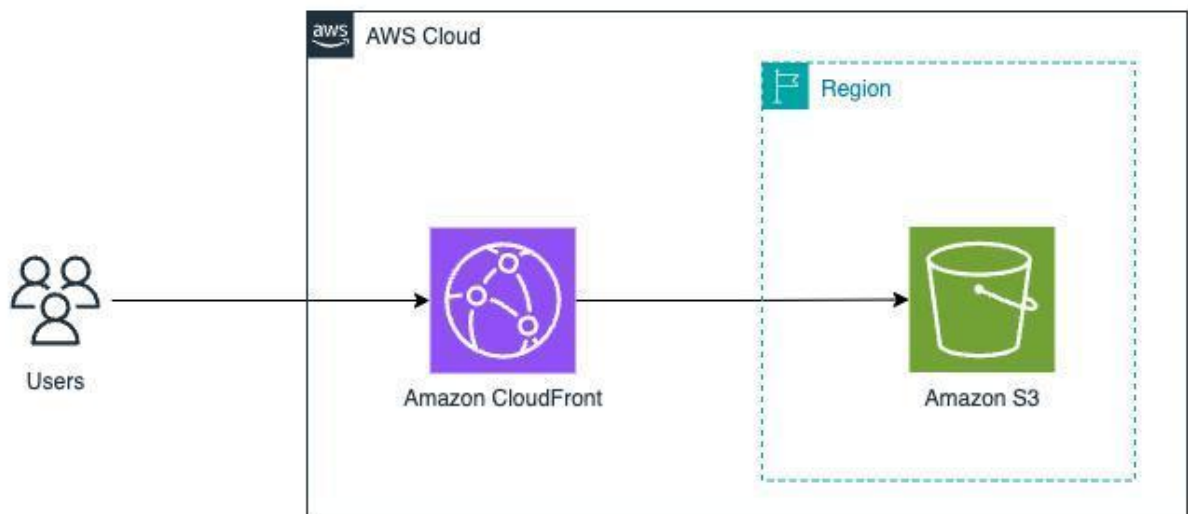
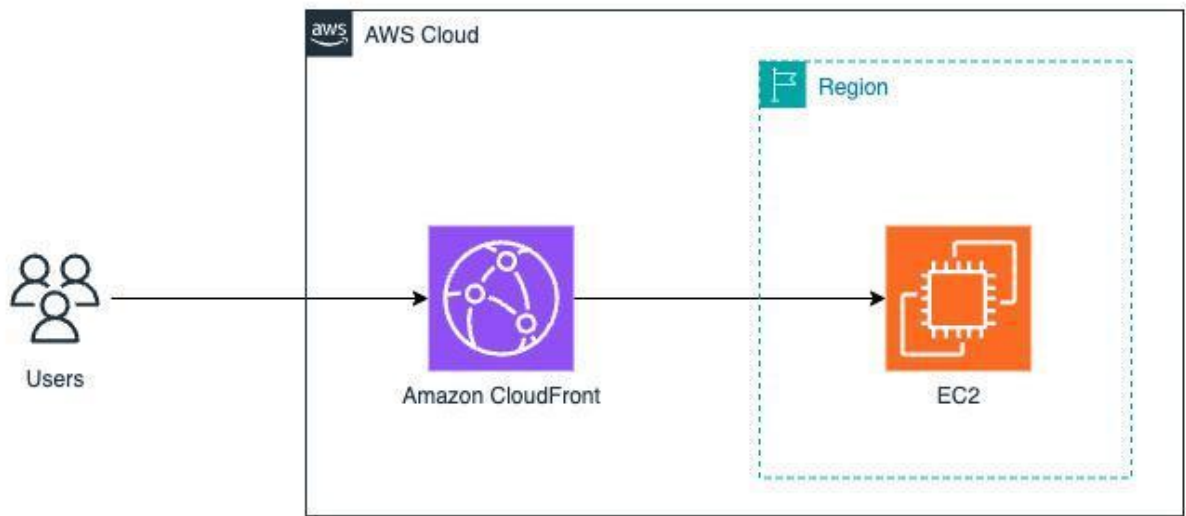
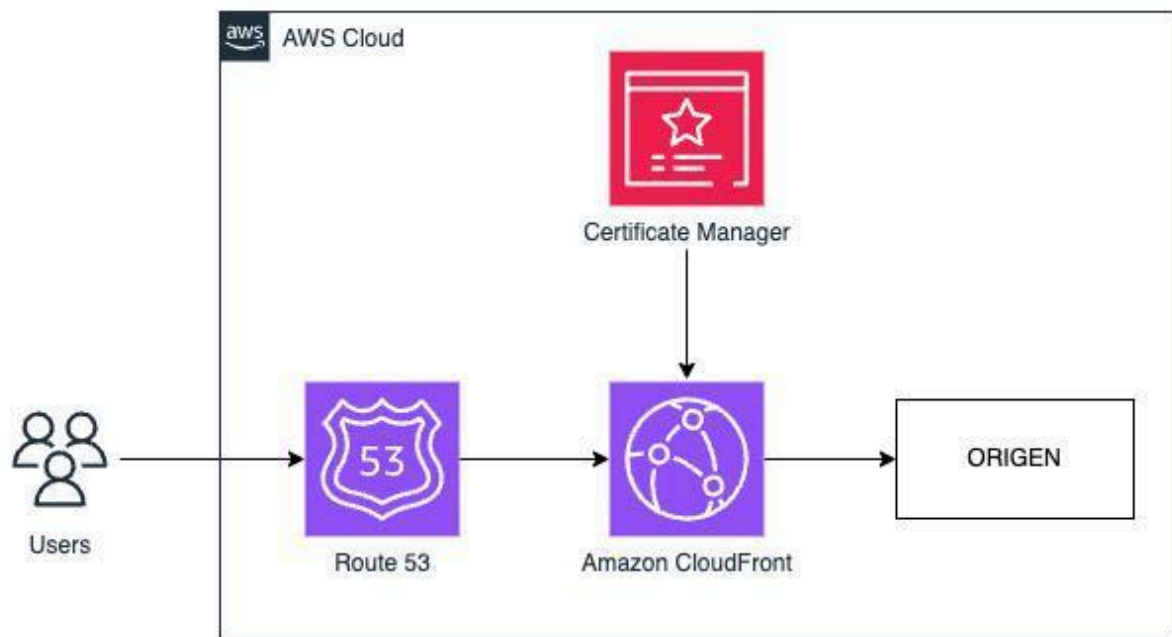


Diagrama con EC2 como Origen



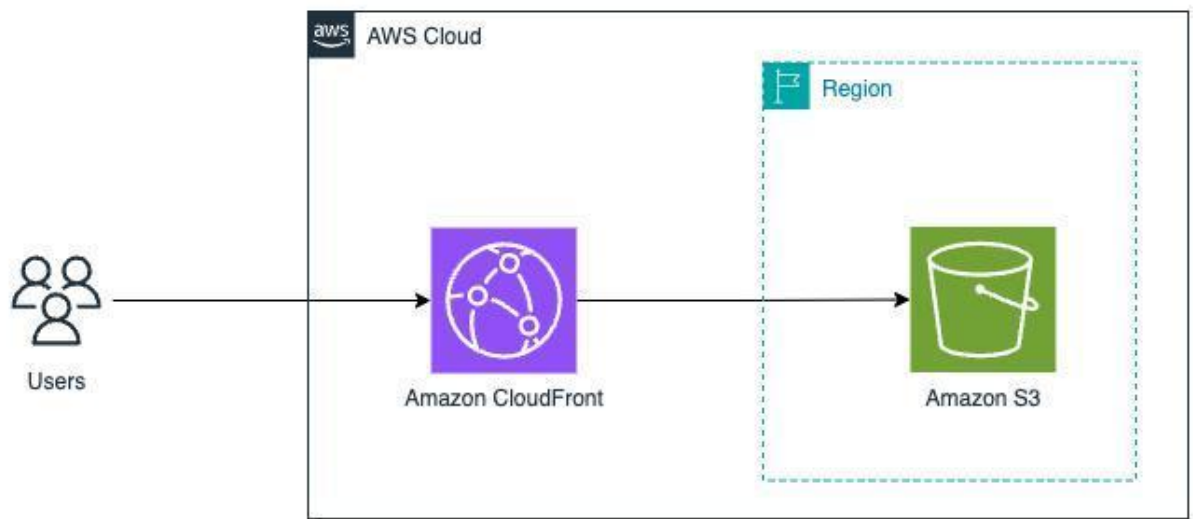
Opcional: Si utilizamos Route53, podemos crear un certificado SSL con ACM (Certification Manager)



Solución:

Setup:

Usaré el siguiente esquema para la implementación:



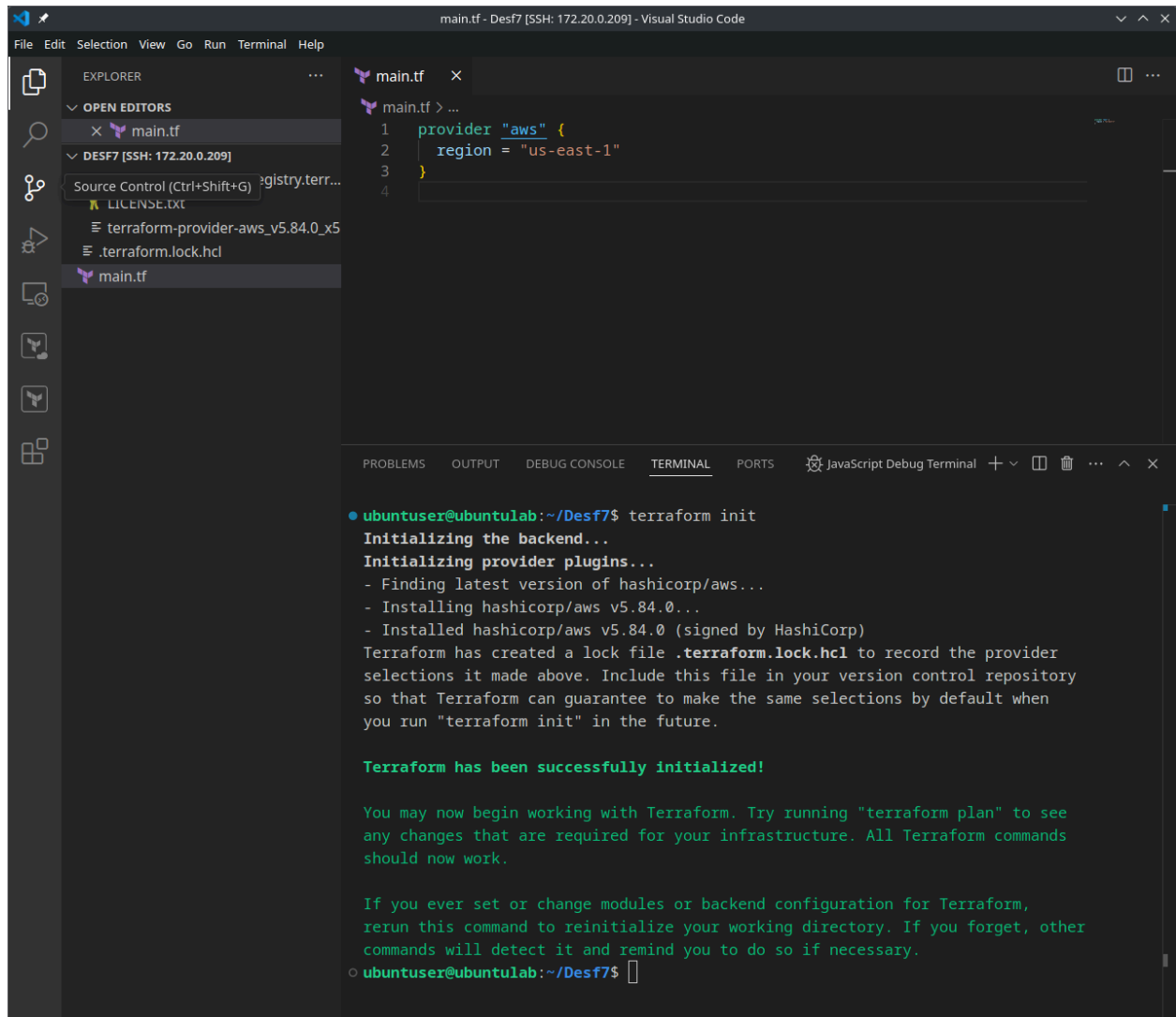
Voy a partir creando un documento principal, main.tf y configurar el proveedor de aws y la región para este proyecto:

La imagen muestra la interfaz de Visual Studio Code con el archivo main.tf abierto. El editor contiene el siguiente código Terraform:

```
1 provider "aws" {  
2   region = "us-east-1"  
3 }  
4
```

El explorador de archivos a la izquierda muestra la estructura del proyecto, incluyendo el archivo main.tf y la extensión HashiCorp Terraform.

terraform init para descargar plugins y hacer el bootstrap del proyecto:



The image shows a Visual Studio Code editor window with a Terraform configuration file named `main.tf` open. The file contains the following code:

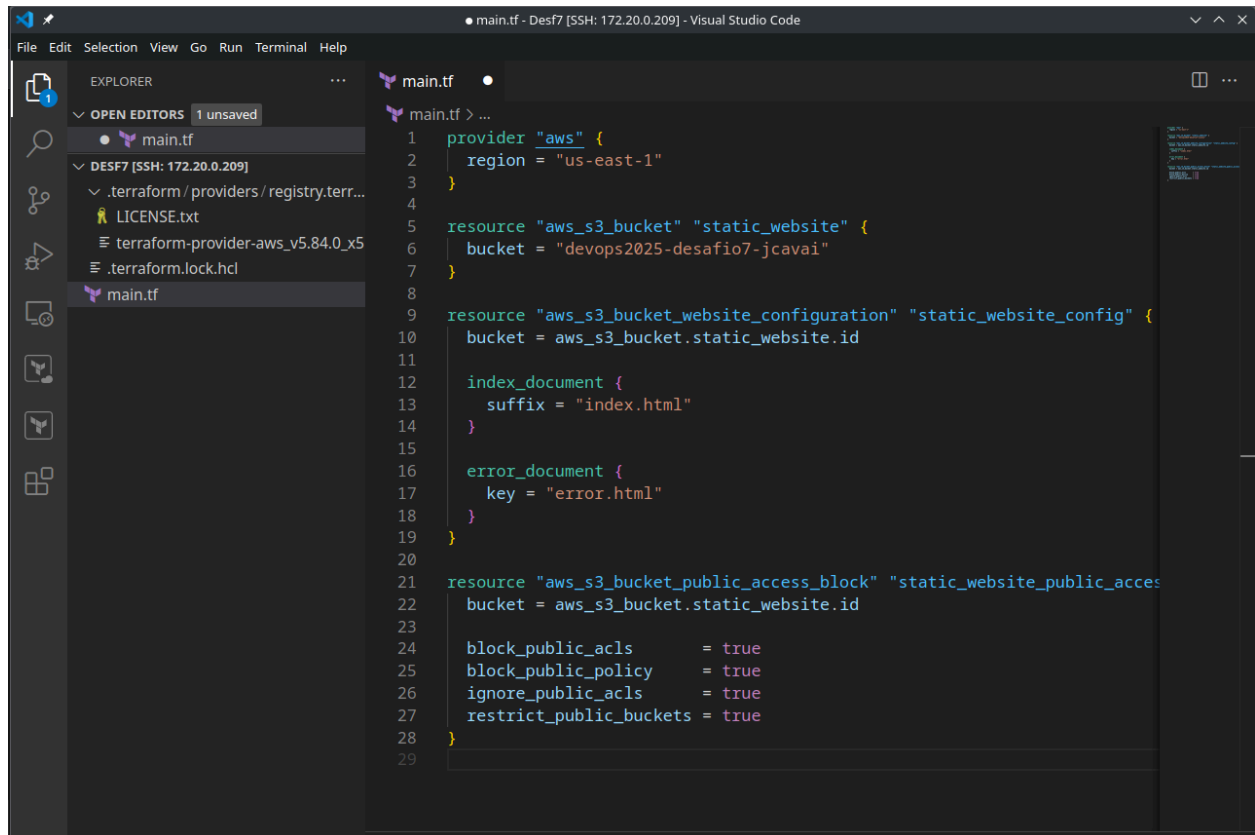
```
1 provider "aws" {  
2   region = "us-east-1"  
3 }  
4
```

The Explorer sidebar on the left shows the file structure, including `main.tf`, `Source Control (Ctrl+Shift+G)`, `gistry.terr...`, `LICENSE.txt`, `terraform-provider-aws_v5.84.0_x5`, `.terraform.lock.hcl`, and `main.tf`.

The Terminal panel at the bottom shows the output of the `terraform init` command:

```
● ubuntuuser@ubuntulab:~/Desf7$ terraform init  
Initializing the backend...  
Initializing provider plugins...  
- Finding latest version of hashicorp/aws...  
- Installing hashicorp/aws v5.84.0...  
- Installed hashicorp/aws v5.84.0 (signed by HashiCorp)  
Terraform has created a lock file .terraform.lock.hcl to record the provider  
selections it made above. Include this file in your version control repository  
so that Terraform can guarantee to make the same selections by default when  
you run "terraform init" in the future.  
  
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.  
○ ubuntuuser@ubuntulab:~/Desf7$
```

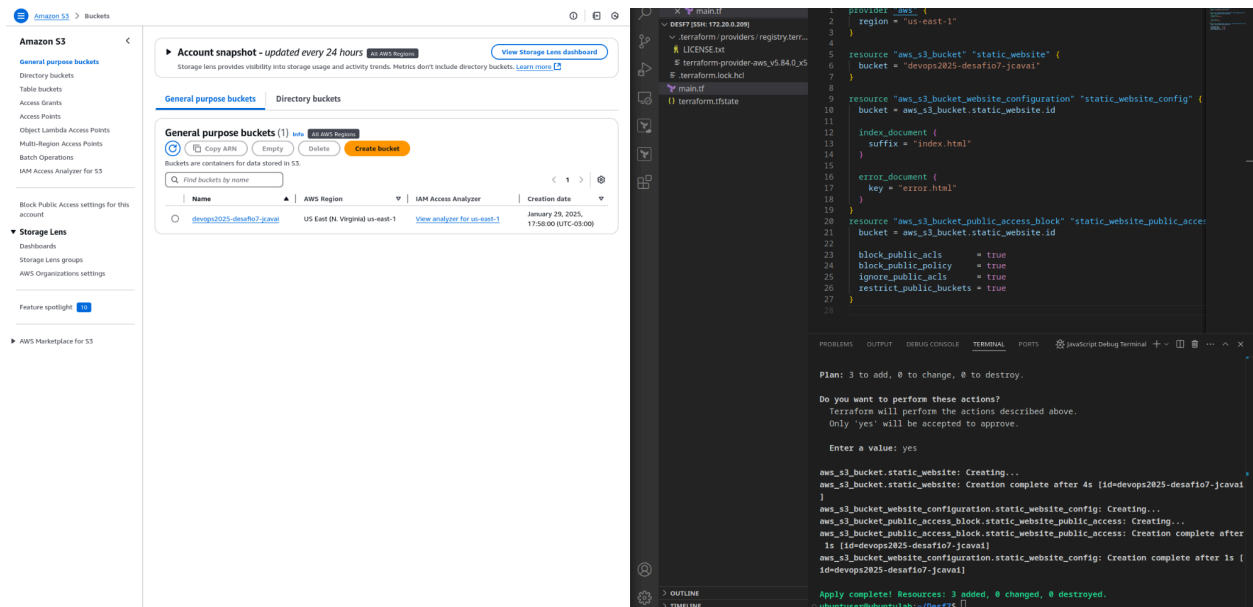
Provisión del Bucket:



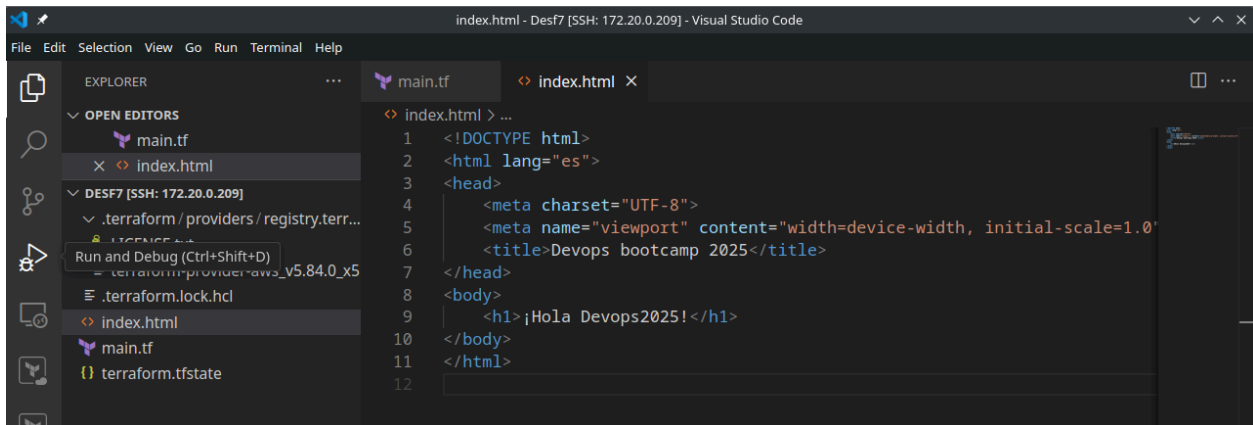
```
1 provider "aws" {
2   region = "us-east-1"
3 }
4
5 resource "aws_s3_bucket" "static_website" {
6   bucket = "devops2025-desafio7-jcavai"
7 }
8
9 resource "aws_s3_bucket_website_configuration" "static_website_config" {
10  bucket = aws_s3_bucket.static_website.id
11
12  index_document {
13    suffix = "index.html"
14  }
15
16  error_document {
17    key = "error.html"
18  }
19 }
20
21 resource "aws_s3_bucket_public_access_block" "static_website_public_access_block" {
22  bucket = aws_s3_bucket.static_website.id
23
24  block_public_acls       = true
25  block_public_policy     = true
26  ignore_public_acls     = true
27  restrict_public_buckets = true
28 }
29
```

- La primera sección (línea 5) corresponde a la definición del tipo de recurso a deployar
`resource "aws_s3_bucket" "static_website"`
- La segunda sección (línea 9) corresponde a la configuración del bucket como sitio estático
`resource "aws_s3_bucket_website_configuration" "static_website_config"`
- La tercera sección (línea 20) corresponde a la configuración de los permisos de Acceso al bucket
`resource "aws_s3_bucket_public_access_block" "static_website_public_access"`

Aplico con `terraform plan` y `terraform apply` y verifico en la consola que se haya aprovisionado el bucket:



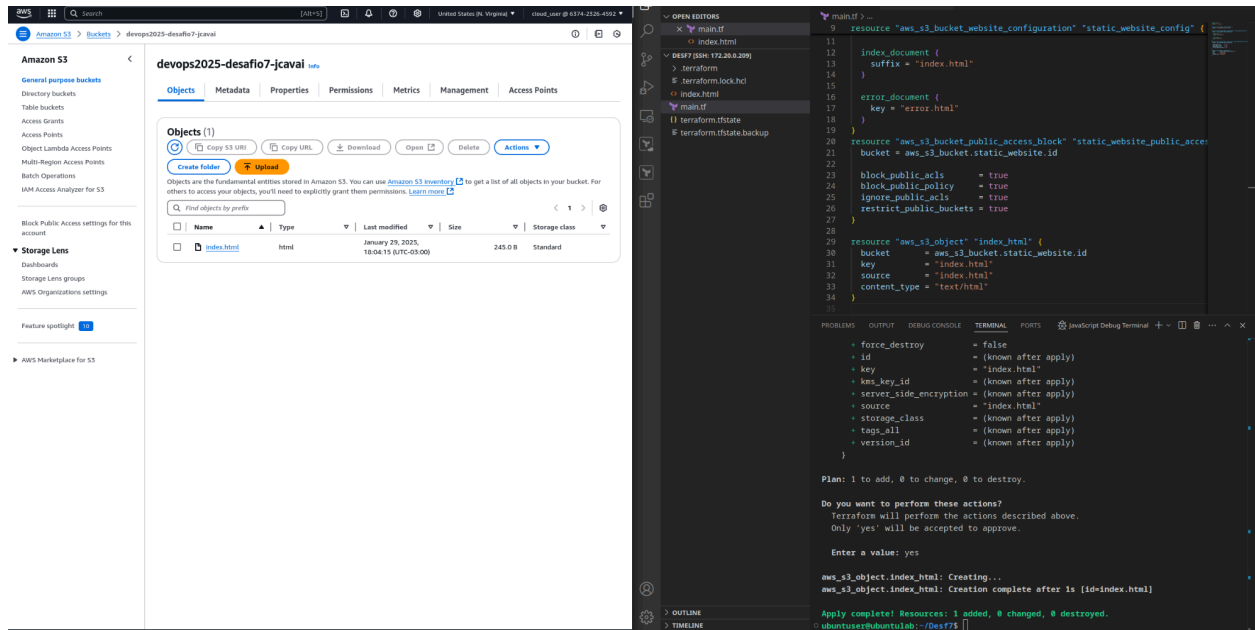
Creo un archivo `index.html` a la raíz del bucket para probar los permisos:



Y agrego el bloque `aws_s3_object` para subir el archivo al bucket:



Aplico con `terraform plan` y `terraform apply` y valido que se subió el archivo y que no lo pueda invocar desde internet usando la url pública:



Acceso denegado al archivo:



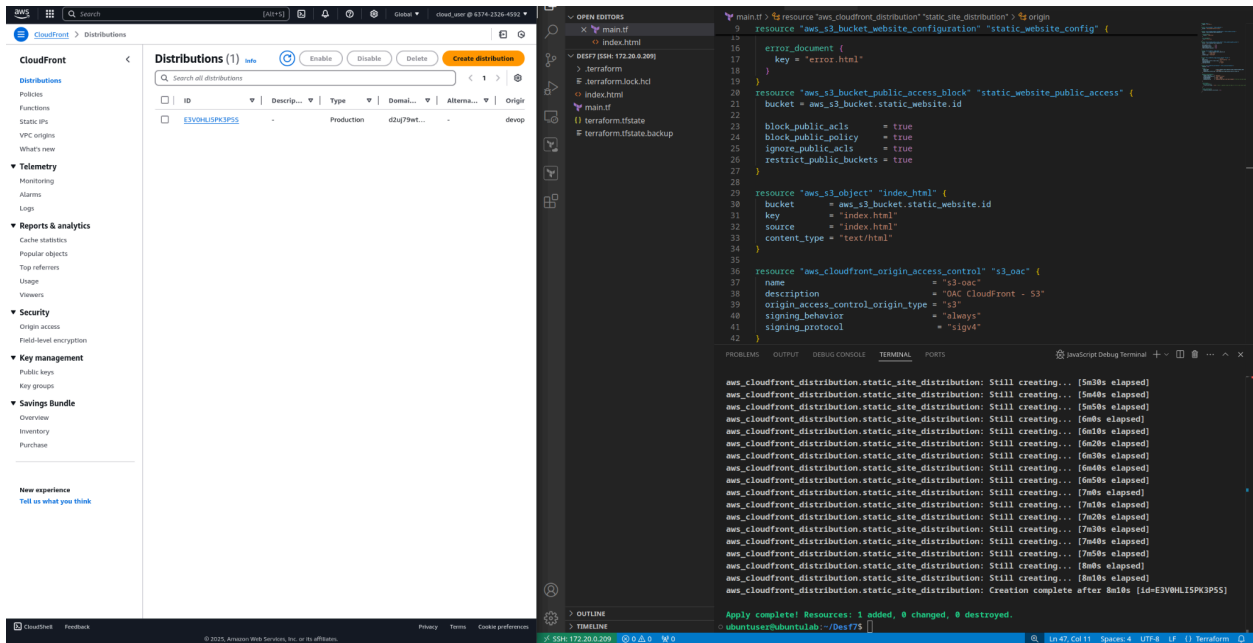
Provisi3n de Cloudfront:

Agregar3 al main.tf dos secciones:

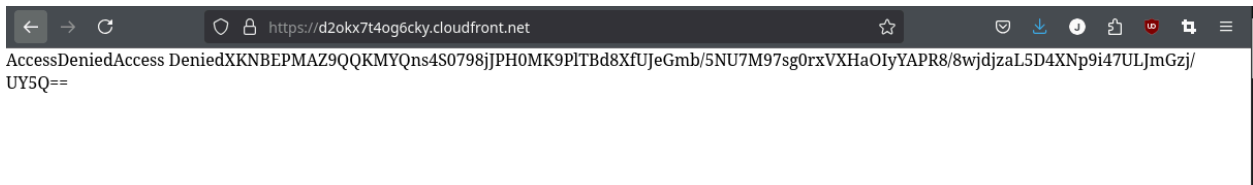
```
34 }
35
36 resource "aws_cloudfront_origin_access_control" "s3_oac" {
37   name           = "s3-oac"
38   description    = "OAC CloudFront - S3"
39   origin_access_control_origin_type = "s3"
40   signing_behavior = "always"
41   signing_protocol = "sigv4"
42 }
43
44 resource "aws_cloudfront_distribution" "static_site_distribution" {
45   enabled = true
46
47   origin {
48     domain_name      = aws_s3_bucket.static_website.bucket_regional_domain_name
49     origin_id        = "S3origin"
50     origin_access_control_id = aws_cloudfront_origin_access_control.s3_oac.id
51   }
52
53   default_cache_behavior {
54     target_origin_id = "S3origin"
55     viewer_protocol_policy = "redirect-to-https"
56     allowed_methods       = ["GET", "HEAD"]
57     cached_methods        = ["GET", "HEAD"]
58     cache_policy_id       = "b2884449-e4de-46a7-ac36-70bc7f1ddd6d" # Política de caché ad
59   }
60 }
61
62 restrictions {
63   geo_restriction {
64     restriction_type = "none" # Esto lo agregue porque me daba un error de que tenia poc
65   }
66 }
67
68 viewer_certificate {
69   cloudfront_default_certificate = true
70 }
71 }
72
73 }
```

- Primera sección (línea 36) definición del *origin access control* para permitir el acceso desde Cloudfront al bucket S3.
- Segunda sección (línea 44) definición de la distribución de cloudfront en sí misma.

Aplico con `terraform plan` y `terraform apply` y verifico en la consola que se haya aprovisionado.

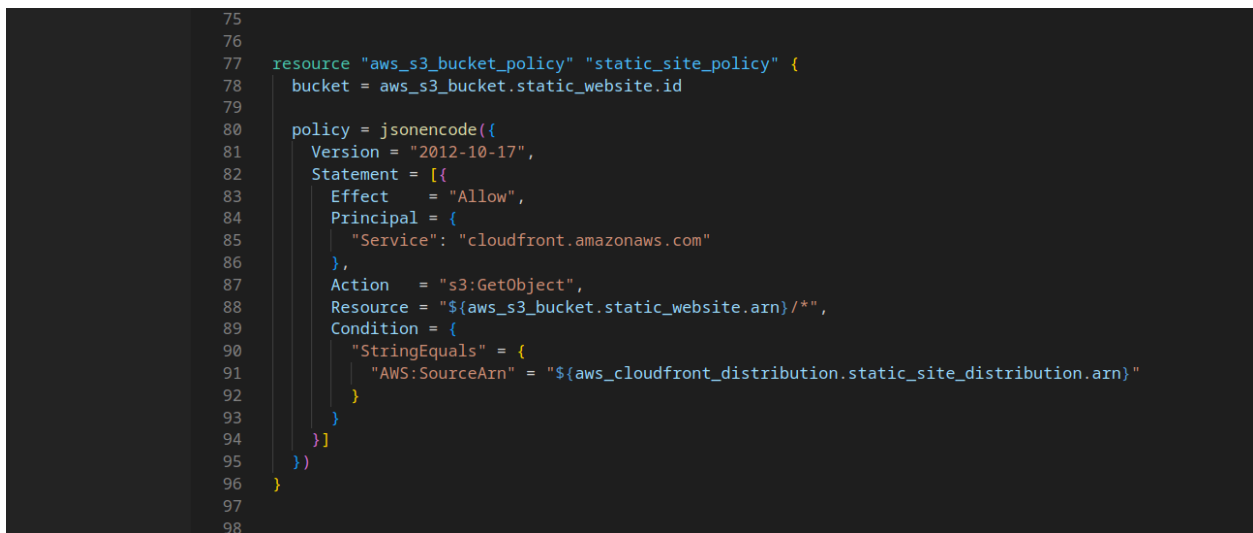


Se verifica que el distribution domain name responde aunque no entrega el contenido (esto es esperable ya que falta la política de acceso Cloudfront <> S3)

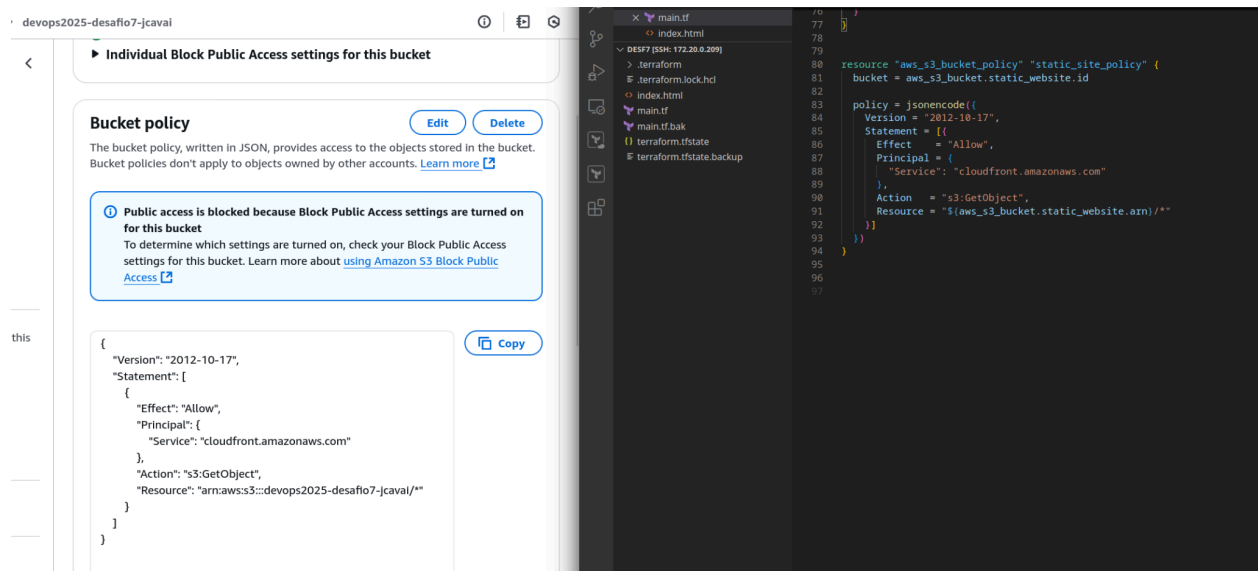


Política de acceso para el bucket de S3:

Solo queda crear la política de acceso y aplicarla al bucket. Agrego el siguiente codigo al final del main.tf:



Esta política de permite que el **CloudFront** (principal: cloudfront.amazonaws.com) acceda (action: getObject) a los objetos almacenados en el bucket de S3 aprovisionado más arriba.



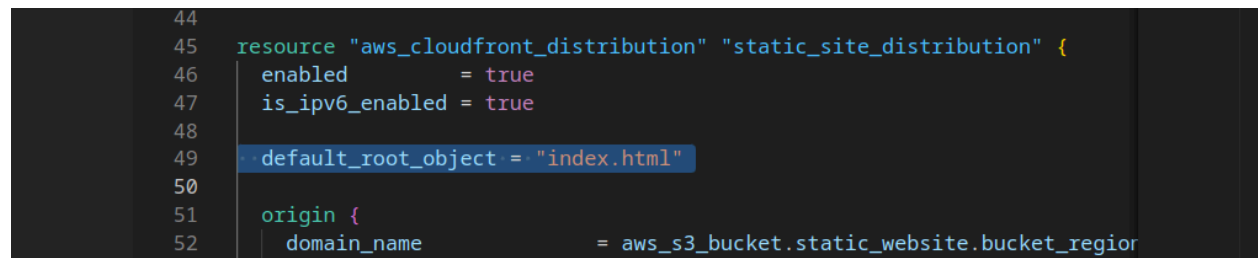
Validación:

Al acceder a la url pública de la distribución de cloudfront se entrega correctamente el archivo del índice



¡Hola Devops2025!

PERO, encontré que necesitaba explicitar que quería el index.html. Aunque en S3 está declarado que el default file es index.html al poner Cloudfront delante del bucket esta instrucción se ignora. Así que agregué `default_root_object = "index.html"` al script en el paso de creación de Cloudfront



Luego de re aplicar:



https://d5adxocna2r45.cloudfront.net

¡Hola Devops2025!

Bonus:

para asegurarme de que todos los recursos provisionados con terraform sean eliminados de la cuenta:

```
terraform destroy
```

```
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 2m40s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 2m50s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 3m0s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 3m10s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 3m20s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 3m30s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 3m40s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 3m50s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 4m0s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 4m10s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 4m20s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 4m30s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 4m40s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 4m50s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 5m0s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 5m10s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 5m20s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 5m30s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 5m40s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 5m50s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 6m0s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 6m10s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 6m20s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 6m30s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 6m40s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 6m50s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 7m0s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 7m10s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 7m20s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 7m30s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 7m40s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 7m50s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 8m0s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 8m10s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 8m20s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 8m30s elapsed]
aws_cloudfront_distribution.static_site_distribution: Still destroying... [id=E3AQAZ3W509L5T, 8m40s elapsed]
aws_cloudfront_origin_access_control.s3_oac: Destroying... [id=E21AJLGI9CVZG5]
aws_s3_bucket.static_website: Destroying... [id=devops2025-desafio7-jcavai]
aws_cloudfront_origin_access_control.s3_oac: Destruction complete after 1s
aws_s3_bucket.static_website: Destruction complete after 1s

Destroy complete! Resources: 7 destroyed.
ubuntu@ubuntu:~/Desf7$
```