Ejercicio: Publicar una página web en Amazon S3

1. Crear un bucket en S3

Entra a la **Consola de AWS → S3**.

Haz clic en **Create bucket**.

Ponle un nombre único (ej: mi-bucket-web-demo).

Elige una región (ej: us-east-1).

Deja todas las opciones por defecto y crea el bucket.

Por defecto el bucket es privado, o sea, no se puede acceder desde internet.

2. Subir el archivo HTML

Abre el bucket creado.

Sube un archivo index.html con el siguiente contenido:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Mi primera página en S3</title>
</head>
<body style="font-family: Arial; text-align: center; margin-top: 50px;">
  <h1>¡Hola desde Amazon S3!</h1>
  <p>Esta es una página web simple almacenada en un bucket S3.</p>
</body>
</html>
```

Una vez subido, fíjate que si intentas abrir el enlace del archivo, te dirá **Access Denied** → porque es privado.

3. Configurar el bucket como hosting estático

Dentro del bucket, ve a la pestaña **Properties**.

Busca la sección **Static website hosting → Enable**.

Pon index.html como documento de inicio.

Guarda los cambios.

Esto te dará una **URL pública de sitio web** (ej: http://mi-bucket-web-demo.s3-website-us-east-1.amazonaws.com).
Pero aún no podrás verla porque los permisos no están abiertos.

4. Dar permisos públicos

Ve a la pestaña **Permissions** del bucket.

En **Bucket Policy**, pega esta política (reemplaza mi-bucket-web-demo con tu nombre real de bucket):

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowPublicRead",
   "Effect": "Allow",
   "Principal": "*",
   "Action": "s3:GetObject",
   "Resource": "arn:aws:s3:::mi-bucket-web-demo/index.html"
  }
 ]
}
```

Guarda.

Nota:
En Amazon S3 tienes **dos formas** de dar permisos públicos a los objetos de tu bucket:

1. Mediante código (Bucket Policy)

   Es es el JSON anterior.
   Muy útil si quieres dar permisos a todos los objetos del bucket de una vez.
   También es más "infraestructura como código", porque puedes copiar esa política y reutilizarla en otros buckets.
   Más flexible y escalable. Ideal si quieres que *todo lo que subas sea público automáticamente*.

   ```
   {"Version": "2012-10-17",
   ```

   Version: Indica la versión del esquema de políticas de AWS.
   "2012-10-17" es la versión más usada y recomendada.

   ```
   "Statement": [
   {
   ```

   Statement: Contiene una o varias reglas de acceso dentro de la política.
   Cada **Statement** define quién puede hacer qué y sobre qué recursos.

`"Sid": "PublicReadGetObject",`

Sid (Statement ID): Es un identificador opcional para la regla.
Aquí se llama "PublicReadGetObject" para indicar que permite lectura pública de objetos.

`"Effect": "Allow",`

Effect: Define si la regla **permite (Allow)** o **deniega (Deny)** la acción.
En este caso, **Allow** significa que se permite la acción especificada.

`"Principal": "*",`

Principal: Quién recibe los permisos.
"*" significa **todos**, es decir, cualquier usuario en Internet.

`"Action": "s3:GetObject",`

Action: Qué acción se permite.
"s3:GetObject" significa **leer o descargar objetos del bucket**.

`"Resource": "arn:aws:s3:::mi-bucket/*"`

Resource: Sobre qué recursos se aplica la regla.
"arn:aws:s3:::mi-bucket/*" significa **todos los objetos dentro del bucket llamado mi-bucket**.

Nota: La regla no da permiso para **listar el bucket** (ListBucket), solo para acceder a los objetos.

2. Mediante la consola gráfica de AWS (sin código)

Lo puedes hacer paso a paso así:

Entra en el bucket → pestaña **Permissions**.
Ve a la sección **Block Public Access** → desactiva las restricciones que bloquean acceso público.
Luego, en la pestaña **Objects**, selecciona tu index.html.
Haz clic en **Actions → Make public** (o **Make public via ACL**).
Con eso solo ese archivo queda público, sin necesidad de escribir la política JSON.
Más sencillo para una prueba rápida o para hacer público solo un archivo específico.

5. Probar en el navegador

Ahora abre la URL del **website endpoint** que te dio S3.

¡Deberías ver la página con el mensaje *"Hola desde Amazon S3"*!

Cambia el recurso html en tu bucket y sube un pantallazo de tu consola al aula virtual para la validación del ejercicio.

Resumen del ejercicio

Con esto ya tienes un flujo completo:

Crear bucket privado.
Subir archivo.
Habilitar hosting estático.
Publicar cambiando permisos.

Antes de la política → el archivo no se puede ver (privado).
Después de la política → el archivo se hace público y se puede abrir desde el navegador.

Nota 2:

Cuándo se usa S3 para publicar web

Web estática: páginas HTML, CSS, JS simples, fotos, portfolios, documentación, landing pages.

Ventajas:

Muy barato, porque solo pagas por almacenamiento y tráfico.
Escalable automáticamente: soporta muchas visitas sin necesidad de servidores.
Fácil de mantener: no hay servidores que administrar.

Limitaciones

No soporta código del lado del servidor: PHP, Python, Node.js, bases de datos, etc.
Si quieres funcionalidades dinámicas, necesitas combinarlo con **AWS Lambda, API Gateway, o un backend separado**.

La URL por defecto de S3 no es muy bonita (ej: http://mi-bucket.s3-website-us-east-1.amazonaws.com) → se suele usar **Route 53 o CloudFront** para un dominio propio.

- Amazon Route 53

  Es el servicio de DNS de AWS.
  Su función principal es traducir nombres de dominio amigables (como www.miweb.com) en direccio nes IP donde están tus servidores o recursos en la nube.
  Características clave:

  Registrar dominios (puedes comprar y gestionar dominios directamente).
  DNS routing: dirigir el tráfico a distintas instancias, buckets S3, Load Balancers, etc.
  Health checks: verifica que tus servidores estén activos y redirige el tráfico si hay fallos.

  Ejemplo:
  Si tienes tu página en un bucket S3 con URL mi-bucket.s3-website-us-east-1.amazonaws.com, con Route 53 puedes hacer que se vea en www.miweb.com.

- Amazon CloudFront

  Es un servicio de CDN (Content Delivery Network).
  Funciona como una capa de distribución global: almacena copias de tu contenido en servidores cercanos a los usuarios.
  Ventajas:

  Mayor velocidad de carga porque los datos vienen del servidor más cercano.
  Reducción de costos de tráfico hacia el bucket S3.
  Seguridad: permite HTTPS, integración con AWS WAF (firewall) y protección DDoS.

  Ejemplo:
  Si tu página web está en S3 en EE. UU. y un usuario en España la abre:
  Sin CloudFront → S3 envía los datos desde EE. UU., más lento.
  Con CloudFront → España recibe los archivos desde un servidor cercano, más rápido.

Nota 3: **Amazon S3** con **API REST**.

API REST (Representational State Transfer) es un estilo de arquitectura que permite interactuar con un servicio usando HTTP/HTTPS.
Una API REST permite que aplicaciones se comuniquen usando HTTP/HTTPS, intercambiando datos con métodos como GET, POST, PUT y DELETE
Usando métodos como:

GET → obtener información

PUT → crear o reemplazar un recurso

POST → crear un recurso

DELETE → eliminar un recurso

Cada recurso tiene su propia URL.

2. Cómo se relaciona con S3

Amazon S3 ofrece una API REST que permite interactuar con los buckets y objetos de forma programática.
Esto significa que puedes:

Crear buckets → PUT Bucket

Subir archivos → PUT Object

Descargar archivos → GET Object

Listar archivos → GET Bucket

Eliminar archivos → DELETE Object

Todo esto se puede hacer desde la consola, SDKs (Python, JavaScript, etc.) o directamente con HTTP/HTTPS

3. Beneficios de la API REST de S3

Permite automatizar la gestión de archivos.
Funciona desde cualquier lenguaje o plataforma que pueda hacer solicitudes HTTP/HTTPS.
Es compatible con herramientas de terceros que usan el protocolo REST.

Amazon S3 es un servicio de almacenamiento que expone su funcionalidad mediante una API REST, lo que permite subir, descargar, listar y eliminar archivos de manera programática, además de integrarlo con aplicaciones web o móviles.

Diagrama conceptual

sql

Copiar código