

INTRODUCCION

lunes, 28 de abril de 2025 9:13

Amazon Web Services.

Tenemos una nueva certificación llamada **AWS Developer Associate**

CODIGO: **DVA-C02**

<https://aws.amazon.com/es/certification/certified-developer-associate/>

Categoría	Associate
Duración del examen	130 minutos
Formato del examen	65 preguntas; varias opciones o varias respuestas
Costo	150 USD. Consulte los precios de los exámenes para obtener información adicional sobre los costos, incluidos los tipos de cambio
Opciones de evaluación	Centro examinador Pearson VUE o examen supervisado en línea
Idiomas disponibles	Inglés, japonés, coreano, portugués (Brasil), chino simplificado y español (América Latina).

<https://aws.amazon.com/es/certification/>

En cuanto a preguntas, tenemos **557** questions

<https://www.examtopics.com/exams/amazon/aws-certified-developer-associate-dva-c02/view/>

Para aprobar se necesitan **720 puntos**.

Nota: Tenemos IA dentro de los exámenes, nada de ir rápido.

Ventajas del examen:

- **El orden de las respuestas no cambia**
- **No tenemos Yes/No ni casos de estudio**

La nota no nos la dan al momento. Tardan 1 o dos días.

Para la certificación, necesitamos una cuenta de Amazon, es decir, dar de alta Nuestro correo de certificaciones en Amazon.

Necesitamos **1€** para darnos de alta en la plataforma de AWS. Nos lo devuelven.

AWS Cloud es igual a Azure e igual a Google Cloud.

Se utiliza trabajando en lo que se denomina **Console AWS**, es decir, la página Web

Una de las características distintas a Azure es que se trabaja mediante **IAM**, es decir, Necesitamos permisos para todo y comunicar todos nuestros servicios.

Trabajaremos con recursos de la capa gratuita, dentro de AWS tenemos múltiples Recursos que nos permiten trabajar con elementos gratis.

<https://aws.amazon.com/es/free/>

**Importante: Los recursos de la capa gratuita DEJAN DE SER GRATUITOS si no
Escuchamos a PACO.**

Características de la certificación:

- **IAM:** Es el módulo más importante de todo AWS. Es el control de claves, Usuarios y accesos para los servicios de AWS. Podemos crear usuarios, Grupos y Roles.
- **S3:** Es el servicio Azure Storage dentro de AWS.
- **RDS:** Son las bases de datos dentro de AWS.
- **EC2:** Son las máquinas virtuales dentro de AWS. No tenemos App Service como tal. Si necesitamos montar algo en producción, debemos crear una máquina virtual y Desplegar ahí lo que deseemos.
- **KMS:** Es el almacén de claves de AWS, lo mismo que Azure Key Vault
- **SNS/SQS:** Sirve para enviar mensajes dentro de AWS a los servicios.
- **Elastic Cache:** Es el mismo servicio que Azure Cache Redis.
- **Lambda:** Son las funciones o Logic Apps dentro de AWS. La ventaja es que podemos Realizar un montón de características por código y comunicar con los servicios.
- **Api Gateway:** Son las puertas de enlace para acceder a nuestras Apis desplegadas Dentro de AWS. Nos permiten acceder a funciones Lambda.
- **Dynamo Db, Cassandra:** Son las bases de datos NON SQL de AWS.
- **Elastic Beanstalk:** Permite desplegar una aplicación como si de un App Service Fuera, es decir, con una forma "sencilla" como si hablásemos del botón Publish De Visual Studio.

Una de las desventajas es que AWS no trabaja con **Recursos**, trabaja con **zonas**

Cuando trabajamos con **Regions**, si estamos en una región, por ejemplo, **Virginia** y Creamos un recurso en **Japón**, no lo veremos en Virginia, solamente lo veremos si Nos movemos a Japón.

Para nuestra cuenta de AWS no es necesario utilizar nuestra cuenta de certificación, son Dos características distintas.

Por un lado, tendremos una cuenta de **AMAZON** para la certificación.

Por otro lado, tendremos nuestra cuenta de **AWS**

Vamos a intentar crear nuestra cuenta AWS.

Nuestra cuenta con lo que deseáis AWS (CORREO).

Explore Free Tier products with a new AWS account.

To learn more, visit aws.amazon.com/free.



Sign up for AWS

Root user email address

Used for account recovery and as described in the [AWS Privacy Notice](#)

paco.garcia.serrano@tajamar365.com

AWS account name

Choose a name for your account. You can change this name in your account settings after you sign up.

Student

Verify email address

OR

Sign in to an existing AWS account

Una vez que nos hemos validado, seleccionamos el nivel gratuito

Registrarse en AWS

Seleccionar un plan de soporte

Elija un plan de soporte para su cuenta personal o empresarial. [Compare planes y ejemplos de precio](#). Puede cambiar su plan en cualquier momento desde la consola de administración de AWS.

Soporte de nivel Basic: gratis

- Recomendado para los usuarios nuevos que recién comienzan a utilizar AWS
- Acceso de autoservicio las 24 horas del día, los 7 días de la semana a los recursos de AWS
- Solo para problemas de facturación y cuentas
- Acceso a Personal Health Dashboard y Trusted Advisor



Soporte Developer: a partir de 29 USD al mes

- Recomendado para desarrolladores que experimentan con AWS
- Acceso por correo electrónico a AWS Support durante el horario laboral
- Tiempos de respuesta de 12 horas (horario laboral)



Soporte Business: a partir de 100 USD al mes

- Recomendado para ejecutar cargas de trabajo de producción en AWS
- Soporte técnico las 24 horas, los 7 días de la semana por correo electrónico, teléfono y chat
- Tiempos de respuesta de 1 hora
- Conjunto completo de recomendaciones de prácticas de Trusted Advisor



Los proyectos de AWS son en grupo.

Os tocará en un grupo de trabajo y debéis elegir un Proyecto para implementarlo.

IAM - Identity Access Management

martes, 29 de abril de 2025 9:19

Este es el servicio más importante de todos.
Es la base de las comunicaciones entre los servicios de AWS.

Si tengo una máquina virtual (EC2) con un API y necesito que dicho API comunique con una base de datos (RDS), necesitamos crear "algo" en IAM para que se habilite la comunicación entre los dos servicios.

Características:

- Permite gestionar usuarios para dar acceso a nuestros recursos
- Centralización de usuarios por Recurso
- Podemos gestionar grupos por tipo de recurso y asociar usuarios a dicho grupo.
- Tenemos autenticación MFA para acceder a nuestra consola. (Nuestro móvil con una App o mediante una Key)
- Podemos crear políticas de acceso a algún servicio. Ejemplo de política: Creamos un usuario y deseamos que al iniciar sesión escriba una contraseña con las características que nosotros indiquemos.

Elementos a gestionar:

- **Usuarios:** Diferentes/personas o cuentas para utilizar características de AWS. Podríamos crear un usuario para conectarse a nuestra consola de AWS y además, podemos limitar sus permisos.
- **Grupos:** Conjunto de usuarios con unas políticas comunes
- **Políticas:** Las políticas son permisos hacia los servicios. Existen políticas predeterminadas o podríamos generar nuestras propias políticas mediante JSON.
- **Roles:** Los Roles son permisos para permitir que un servicio sea capaz de comunicar con otro servicio.

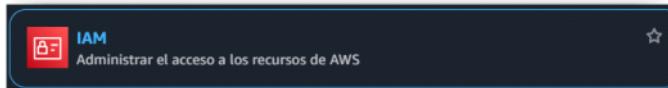
Vamos a comenzar creando un nuevo grupo y usuarios en dicho grupo.
Los nombres de grupo no pueden comenzar ni con símbolos ni con números.

Los nombres de AWS suelen tener esta sintaxis:

AlumnosTajamar, **alumnos-tajamar**, Alumnos-Tajamar

Al crear un grupo, se deben indicar los permisos a la vez.

Buscamos IAM dentro de nuestros servicios. No funciona por regiones, es global.



En la parte de la derecha, tenemos el ID de nuestra cuenta que será único.
Tenemos una URL de acceso a nuestra cuenta para los usuarios de IAM que generemos.

Vamos a comenzar creando un grupo llamado **Developers** con política de acceso Administrador.

El siguiente paso es crear un usuario para poder incluirlo dentro del grupo

Dicho usuario, en el momento de almacenarlo en el grupo, se convertirá en Administrador por las políticas.

Vamos a crearlo para que pueda entrar en la Consola, es decir, para que se conecte a la página Web.

Podemos crear usuarios para que tengan permisos de acceso a la Consola de AWS o podemos crear usuarios para que tengan permisos de otra forma (NO CONSOLA)

Creamos un nuevo usuario llamado **martes**

Especificar los detalles de la persona

Detalles de la persona

Nombre de usuario

martes

El nombre de usuario puede tener un máximo de 64 caracteres. Caracteres válidos: A-Z, a-z, 0-9, and + = , . @ _ - (guion)

Proporcione acceso de usuario a la consola de administración de AWS: opcional

Si proporciona acceso a la consola a una persona, se trata de una práctica recomendada para administrar su acceso en IAM Identity Center.

i ¿Está proporcionando acceso a la consola a una persona?

Tipo de persona

Especificar una persona en Identity Center: recomendado

Le recomendamos que utilice Identity Center para proporcionar acceso a la consola a una persona. Con Identity Center, puede administrar de forma centralizada el acceso de las personas a sus cuentas de AWS y aplicaciones en la nube.

Quiero crear un usuario de IAM

Le recomendamos que cree usuarios de IAM solo si necesita habilitar el acceso mediante programación a través de claves de acceso, credenciales específicas de servicios para AWS CodeCommit o Amazon Keypairs o una credencial de copia de seguridad para el acceso a la cuenta de emergencia.

Contraseña de la consola

Contraseña generada automáticamente

Puede ver la contraseña después de crear la persona.

Contraseña personalizada

Ingrésese una contraseña personalizada para la persona.

- Debe tener como mínimo 8 caracteres
- Debe incluir al menos tres de los siguientes tipos de caracteres: letras mayúsculas (A-Z), letras minúsculas (a-z), números (0-9) y símbolos ! @ # \$ % ^ & * () _ +

Mostrar contraseña

Las personas deben crear una nueva contraseña en el siguiente inicio de sesión (recomendado).

Las personas obtienen automáticamente la IAMUserChangePassword política para poder cambiar su propia contraseña.

i Si está creando acceso mediante programación a través de claves de acceso o credenciales específicas de servicios para AWS CodeCommit o / crear este usuario de IAM. [Más información](#)

Establecer permisos

Agregue una persona a un grupo existente o cree uno nuevo. El uso de grupos es una práctica recomendada para administrar los permisos de

Opciones de permisos

Agregar persona al grupo

Agregue la persona a un grupo existente o cree uno nuevo. Le recomendamos que utilice grupos para administrar los permisos de usuario según las funciones laborales.

Copiar permisos

Copie todas las suscripciones a grupos, las políticas administradas adjuntas y las políticas insertadas de una persona existente.

Grupos de personas (1/1)

Buscar

Nombre del grupo

▲ | Personas

▼ | Políticas adjuntas

Developers

0

AdministratorAccess

Recuperar contraseña

Puede ver y descargar la contraseña de la persona a continuación o enviar por correo electrónico instrucciones a las personas para iniciar sesión en la consola de administración de AWS. Esta es la única vez que puede ver y descargar esta contraseña.

Detalles de inicio de sesión en la consola

[Instrucciones de inicio de sesión por correo electrónico](#)

URL de inicio de sesión de la consola

https://772056227005.siginin.aws.amazon.com/console

Nombre de usuario

martes

Contraseña de la consola

xyT66&59 [Ocultar](#)

[Cancelar](#)

[Descargar archivo.csv](#)

[Volver a la lista de personas](#)

Al crear el usuario, podremos visualizar su nombre, contraseña y URL de acceso con el ID de nuestra cuenta.

Nuestra Url de acceso es única e indicar el ROOT, que es el SUPER usuario, propietario de la cuenta.

Column 1	Column 2	Column 3
Nombre de usuario: martes	Contraseña: xyT66&59	URL de inicio de sesión de la consola: https://772056227005signin.aws.amazon.com/console

Vamos a probar la conexión con nuestro nuevo usuario **martes**

Utilizamos otro explorador para acceder a la nueva URL.

La contraseña, como nos va a pedir una nueva, ponemos la de **Am@zon321**

IAM user sign in ⓘ

Account ID or alias (Don't have?)
772056227005

Remember this account

IAM username
martes

Password
xyT66&59

Show Password [Having trouble?](#)

Sign in

[Sign in using root user email](#)

[Create a new AWS account](#)

Como podemos comprobar, podemos realizar todo lo que deseemos como administradores.

Para comprobar las políticas de acceso, vamos a quitar permisos del Grupo Developers a nuestro usuario **martes**.

Nos desconectamos del usuario **martes**



Developers [Información](#) [Eliminar](#)

Resumen [Editar](#)

Nombre del grupo de personas Developers	Hora de creación April 29, 2025, 09:36 (UTC+02:00)	ARN arn:aws:iam::772056227005:group/Developers
--	---	---

[Personas \(1\)](#) [Permisos](#) [Access Advisor](#)

Políticas de permisos (1) [Información](#) [Simular](#) [Eliminar](#) [Añadir permisos ▾](#)

Puede asociar hasta 10 políticas administradas.

Nombre de la política	Tipo	Entidades asociadas
<input type="checkbox"/> AdministratorAccess	Administrada por AWS: función de trabajo	1

[Buscar](#) [Filtrar por Tipo](#) [Todos los tipos](#)

[Asociar políticas](#) [Crear política insertada](#)

Asociar políticas de permisos a Developers

► Políticas de permisos actuales (1)

Otra políticas de permisos (1/1044)

Puede asociar hasta 10 políticas administradas a este grupo de personas. Todas las personas de este grupo tienen acceso.

Nombre de la política	Tipo
<input type="checkbox"/>  AmazonDMSRedshiftS3Role	Administrada por AWS
<input checked="" type="checkbox"/>  AmazonS3FullAccess	Administrada por AWS

Personas (1)

Permisos

Access Advisor

Políticas de permisos (1/2) Información

Puede asociar hasta 10 políticas administradas.



Simular

Eliminar

Añadir

Filtrar por Tipo

Todos los tipos

Veremos al entrar con **martes** que no tenemos servicio

IAM > Panel

Identity and Access Management (IAM)

Buscar en IAM

Panel

Administración del acceso

- Grupos de personas
- Personas
- Roles
- Políticas
- Proveedores de identidad
- Configuración de cuenta
- Administración del acceso raíz

Novedad

Informes de acceso

- Access Analyzer
- Acceso externo
- Acceso no utilizado
- Configuración del analizador

Informe de credenciales

Actividad de la organización

Políticas de control de servicios

Panel de IAM Información

Recomendaciones de seguridad 0

① Acceso denegado
No tiene permiso para `iam:GetAccountSummary`. Para solicitar acceso, copie el siguiente texto y envíelo a su administrador de AWS. [Más información sobre la solución de problemas de acceso denegado.](#)

Usuario: arn:aws:iam::772056227005:user/martes
Acción: iam:GetAccountSummary
Contexto: no identity-based policy allows the action

② Acceso denegado
No tiene permiso para `iam>ListMFADevices`. Para solicitar acceso, copie el siguiente texto y envíelo a su administrador de AWS. [Más información sobre la solución de problemas de acceso denegado.](#)

Usuario: arn:aws:iam::772056227005:user/martes
Acción: iam>ListMFADevices
Contexto: no identity-based policy allows the action

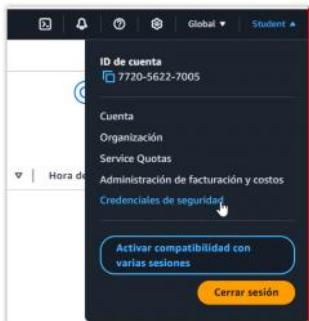
③ Acceso denegado
No tiene permiso para `iam>ListAccessKeys`. Para solicitar acceso, copie el siguiente texto y envíelo a su administrador de AWS. [Más información sobre la solución de problemas de acceso denegado.](#)

Usuario: arn:aws:iam::772056227005:user/martes
Acción: iam>ListAccessKeys
Contexto: no identity-based policy allows the action

Eliminamos nuestro Usuario martes y el grupo Developers

HABILITAR SEGURIDAD MFA CUENTA AWS

Este es habilitar la seguridad Multi Factor. Es necesario una aplicación móvil o una llave de Amazon. Debemos entrar en nuestra cuenta y en credenciales de seguridad.



ID de cuenta
7720-5622-7005

Cuenta
Organización
Service Quotas
Administración de facturación y costos
Credenciales de seguridad

Activar compatibilidad con varias sesiones

Cerrar sesión

Mis credenciales de seguridad

Root user Información

El usuario raíz tiene acceso a todos los recursos de AWS de esta cuenta y le recomendamos que haga lo siguiente [prácticas recomendadas](#). Para obtener más información sobre los tipos de credenciales de AWS y cómo se usan, consulte [Credenciales de seguridad de AWS](#) en la referencia general de AWS.



Como práctica recomendada de seguridad, aconsejamos asignar la MFA.

Asignar MFA

MFA device name

Nombre del dispositivo

Este nombre se utilizará en el ARN de identificación de este dispositivo.

movilpaco

Máximo 64 caracteres. Caracteres válidos: A-Z, a-z, 0-9, y + = , - _ (guion)

MFA device

Opciones de dispositivo

Además del nombre de usuario y la contraseña, utilizará este dispositivo para autenticarse en la cuenta.



Clave de paso o clave de seguridad

Autentíquese mediante la huella digital, el rostro o el bloqueo de pantalla. Cree una clave de paso en este dispositivo o use otro dispositivo, como una clave de seguridad FIDO2.



Aplicación del autenticador

Autentíquese mediante un código generado por una aplicación instalada en el dispositivo móvil o la computadora.



Token de contraseña temporal de un solo uso (TOTP) de hardware

Autentíquese mediante un código generado por el token de TOTP de hardware u otros dispositivos de hardware.

Android

[Twilio Authy Authenticator](#), [Duo Mobile](#), [Microsoft Authenticator](#), [Google Authenticator](#), [Symantec VIP](#)

iOS

[Twilio Authy Authenticator](#), [Duo Mobile](#), [Microsoft Authenticator](#), [Google Authenticator](#), [Symantec VIP](#)

Aplicación del autenticador

Un dispositivo MFA virtual es una aplicación que se ejecuta en el dispositivo y que puede configurar al escanear un código QR.

1

Instale una aplicación compatible, como Google Authenticator, Duo Mobile o Authy, en su dispositivo móvil o computadora.

[Consultar una lista de aplicaciones compatibles](#)

2



Abra la aplicación de autenticación, elija **Mostrar código QR** en esta página y, a continuación, utilice la aplicación para escanear el código. También puede escribir una clave secreta.

[Mostrar clave secreta](#)

3

Escriba dos códigos MFA consecutivos a continuación

Ingrese un código de su aplicación virtual a continuación

059629

Espere 30 segundos e introduzca una segunda entrada de código.

727415

[Cancelar](#)

[Anterior](#)

[Agregar MFA](#)

Si no queremos autenticación con MFA, lo podemos eliminar perfectamente

Autenticación multifactor (MFA) (1)

[Eliminar](#)[Volver a sincronizar](#)[Asignar dispositivo MFA](#)

Utilice MFA para aumentar la seguridad del entorno de AWS. Para iniciar sesión con MFA, se requiere un código de autenticación de un dispositivo MFA. Cada persona puede tener un máximo de 8 dispositivos MFA asignados. [Más información](#)

Tipo	Identificador	Certificaciones	Creada el
<input checked="" type="radio"/> Virtual	arn:aws:iam::772056227005:mfa/movilpaco	No aplicable	Tue Apr 29 2025

ROLES

Un Role dentro de AWS nos permite comunicar servicios dentro del entorno AWS.

Si tenemos una máquina EC2 con un MVC y queremos comunicar con un servicio S3 Storage, debemos crear un Role e incluirlo dentro de la máquina para poder comunicar los dos servicios.

Seleccionar entidad de confianza [Información](#)

Tipo de entidad de confianza

- Servicio de AWS
Permita que servicios de AWS como EC2, Lambda u otros realicen acciones en esta cuenta.
- Cuenta de AWS
Permitir a las entidades de otras cuentas de AWS que le pertenezcan a usted o a un tercero realizar acciones en esta cuenta.
- Identidad web
Permite a las personas federadas por el proveedor de identidad web externo especificado asumir este rol para realizar acciones en esta cuenta.
- Federación SAML 2.0
Permitir que las personas federadas con SAML 2.0 a partir de un directorio corporativo realicen acciones en esta cuenta.
- Política de confianza personalizada
Cree una política de confianza personalizada para permitir que otras personas realicen acciones en esta cuenta.

Caso de uso

Permita que un servicio de AWS, como EC2, Lambda u otros, realicen acciones en esta cuenta.

Servicio o caso de uso

Elegir un servicio o caso de uso ▾

Caso de uso

Permita que un servicio de AWS, como EC2, Lambda u otros, realicen acciones en esta cuenta.

Servicio o caso de uso

EC2 ▾

Elija un caso de uso para el servicio especificado.

Caso de uso

- EC2
Allows EC2 instances to call AWS services on your behalf.
- EC2 Role for AWS Systems Manager
Allows EC2 instances to call AWS services like CloudWatch and Systems Manager on your behalf.
- EC2 Spot Fleet Role
Allows EC2 Spot Fleet to request and terminate Spot Instances on your behalf.
- EC2 - Spot Fleet Auto Scaling
Allows Auto Scaling to access and update EC2 spot fleets on your behalf.
- EC2 - Spot Fleet Tagging
Allows EC2 to launch spot instances and attach tags to the launched instances on your behalf.
- EC2 - Spot Instances
Allows EC2 Spot Instances to launch and manage spot instances on your behalf.
- EC2 - Spot Fleet
Allows EC2 Spot Fleet to launch and manage spot fleet instances on your behalf.
- EC2 - Scheduled Instances
Allows EC2 Scheduled Instances to manage instances on your behalf.

Agregar permisos [Información](#)

Políticas de permisos (1/1045) [Información](#)

Elija una o varias políticas para adjuntarlas al nuevo rol.

Filtrar por Tipo

<input type="text" value="S3"/>	<input type="button" value="X"/>	<input type="button" value="Todos los tipos"/>
<input type="checkbox"/> Nombre de la política ▾	<input type="button" value="▲"/>	<input type="button" value="▼"/>
<input type="checkbox"/>  AmazonDMSRedshiftS3Role		Administrada por AWS
<input checked="" type="checkbox"/>  AmazonS3FullAccess		Administrada por AWS
<input type="checkbox"/>  AmazonS3ObjectLambdaExecutionRolePolicy		Administrada por AWS
<input type="checkbox"/>  AmazonS3OutpostsFullAccess		Administrada por AWS

Detalles del rol

Nombre del rol

Ingrese un nombre significativo para identificar a este rol.

EC2-allow-S3

64 Caracteres máximos. Utilice caracteres alfanuméricos y '+, @-_.

Descripción

Agregue una breve explicación para este rol.

Allows EC2 instances to call AWS services on your behalf.

Paso 1: seleccionar entidades de confianza

Política de confianza

```
1 * [{}  
2     "Version": "2012-10-17",  
3     "Statement": [  
4         {  
5             "Effect": "Allow",  
6             "Action": [  
7                 "sts:AssumeRole"  
8             ],  
9             "Principal": {  
10                 "Service": [  
11                     "ec2.amazonaws.com"  
12                 ]  
13             }  
14         }  
15     ]  
16 ]
```

POLITICAS

Las políticas son permisos para usuarios/grupos en nuestros entornos de desarrollo.

Esto es básico para nosotros ya que AWS no utiliza Keys tal y como las conocemos dentro del entorno de Azure, sino que

Se utilizan Roles de usuario para acceder a los permisos de forma programática en VS.

Acabamos de utilizar una política llamada **AmazonS3FullAccess**, pero también podríamos crear nuestra propia política personalizada.

S3 -Storage Service

martes, 29 de abril de 2025 10:51

Es el servicio de almacenamiento en la nube de AWS.

Es el servicio igual a Azure Storage, pero solamente con la parte que depende de Containers y Blobs.

Nosotros actualmente estamos utilizando nuestro Servicio Azure Storage en nuestro proyecto, simplemente debemos configurar el acceso al almacenamiento de AWS, es decir, solamente cambiar la URL de devolución del Api.

Todo lo que almacena son Stream y permite almacenar hasta 5Gb gratis.



En este servicio los datos se almacenan en **Buckets**. El nombre de los buckets es único y global.

La disponibilidad de los ficheros del Bucket es inmediata, pero si los eliminamos o modificamos, tenemos un tiempo para visualizar los cambios.

Los buckets tienen un cifrado nativo para subir sus ficheros (KMS).

Cada fichero contiene la siguiente información:

- Key: Es el nombre del fichero
- Versión ID: El número de versión del fichero
- Value: Contenido del propio fichero (stream)
- Metadatos: Información extra (key/value) que podemos almacenar en los ficheros.
- Access Control: Quién tiene acceso al fichero (ACL)

Este servicio estará en Producción con **Regiones**. Trabajamos con **Virginia**

Tenemos varios tipos de S3:

- **S3 Glacier**: Es el servicio más barato y se utiliza para ficheros de larga duración, es decir, que no vamos a estar realizando consultas de acción sobre ellos. No nos importa el tiempo de acceso a los ficheros
- **S3 Standard**: Este servicio contiene una alta disponibilidad del 99.99% y se utiliza para acceso frecuente.
- **S3 IA**: Acceso poco frecuente a los ficheros y nos permite habilitar el cobro por acceso a ficheros.
- **S3 One-Zone-IA**: Es igual al anterior en cuanto a disponibilidad pero estará limitado el acceso solamente a una zona, es decir, si queremos que una máquina EC2 acceda a un fichero en Virginia, dicha máquina tiene que estar también en Virginia.

Tenemos una máquina EC2 en Irlanda.

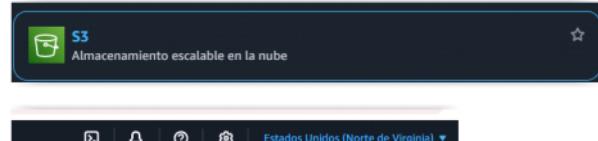
Tenemos un servicio S3 One-Zone-IA en Virginia

Si deseamos que la máquina virtual pueda acceder al servicio S3, ¿Qué debemos hacer?

- 1) Cambiar de Zona el EC2
- 2) Cambiar el plan del S3
- 3) Crear un Role de acceso desde el EC2 hasta el S3

Abrimos el servicio S3 y vamos a crear un bucket llamado **bucket-test-NOMBRE**

Trabajamos en **Virginia**



Crear bucket Información

Los buckets son contenedores de datos almacenados en S3.

Configuración general

Región de AWS
EE.UU. Este (Norte de Virginia) us-east-1

Tipo de bucket Información

Uso general
Recomendado para la mayoría de los casos de uso y patrones de acceso. Los buckets de uso general son del tipo de bucket de S3 original. Permiten una combinación de clases de almacenamiento que almacenan objetos de forma redundante en múltiples zonas de disponibilidad.

Directorio
Recomendado para casos de uso de baja latencia. Estos Zone, que proporciona un procesamiento más rápido de

Nombre del bucket Información

Los nombres de los buckets deben tener entre 3 y 63 caracteres y ser únicos dentro del espacio de nombres global. Los nombres de los buckets también deben empezar y terminar con una letra o un número.

Copiar la configuración del bucket existente: opcional
Solo se copia la configuración del bucket en los siguientes ajustes.

Elegir el bucket
Formato: s3://bucket/prefixo

Vamos a dejar todas las opciones por defecto, simplemente creamos el Bucket de forma privada

Propiedad de objetos Información

Controle la propiedad de los objetos escritos en este bucket desde otras cuentas de AWS y el uso de listas de control de acceso (ACL). La propiedad de los objetos determina quién puede especificar el acceso a los objetos.

ACL deshabilitadas (recomendado)

Todos los objetos de este bucket son propiedad de esta cuenta. El acceso a este bucket y sus objetos se especifica solo mediante políticas.

ACL habilitadas

Los objetos de este bucket pueden ser propiedad de otras cuentas de AWS. El acceso a este bucket y sus objetos se puede especificar mediante ACL.

Propiedad del objeto

Aplicada al propietario del bucket

Configuración de bloqueo de acceso público para este bucket

Se concede acceso público a los buckets y objetos a través de listas de control de acceso (ACL), políticas de bucket, políticas de puntos de acceso o todas las anteriores. A fin de garantizar que se bloquee el acceso activo Bloquear todo el acceso público. Esta configuración se aplica exclusivamente a este bucket y a sus puntos de acceso. AWS recomienda activar Bloquear todo el acceso público, pero, antes de aplicar cualquier otra configuración, las aplicaciones funcionarán correctamente sin acceso público. Si necesita cierto nivel de acceso público a los buckets u objetos, puede personalizar la configuración individual a continuación para adaptarla a sus necesidades específicas. [Más información](#)

Bloquear todo el acceso público

Activar esta configuración equivale a activar las cuatro opciones que aparecen a continuación. Cada uno de los siguientes ajustes son independientes entre sí.

- Bloquear el acceso público a buckets y objetos concedido a través de nuevas listas de control de acceso (ACL)**
S3 bloquera los permisos de acceso público aplicados a objetos o buckets agregados recientemente, y evitara la creación de nuevas ACL de acceso público para buckets y objetos existentes. Esta configuración no cambia los permisos existentes de S3 mediante ACL.
- Bloquear el acceso público a buckets y objetos concedido a través de cualquier lista de control de acceso (ACL)**
S3 ignorará todas las ACL que conceden acceso público a buckets y objetos.
- Bloquear el acceso público a buckets y objetos concedido a través de políticas de bucket y puntos de acceso públicas nuevas**
S3 bloquera las nuevas políticas de buckets y puntos de acceso que concedan acceso público a buckets y objetos. Esta configuración no afecta a las políticas ya existentes que permiten acceso público a los recursos de S3.
- Bloquear el acceso público y entre cuentas a buckets y objetos concedido a través de cualquier política de bucket y puntos de acceso pública**
S3 ignorará el acceso público y entre cuentas en el caso de buckets o puntos de acceso que tengan políticas que concedan acceso público a buckets y objetos.

Vamos a subir un fichero y comprobar si tenemos acceso a él.

1a.jpg Información

[Copiar URI de S3](#) [Descargar](#) [Abrir](#)

[Propiedades](#) [Permisos](#) [Versiones](#)

Información general sobre el objeto

Propietario

paco.garcia.serrano.3213

Región de AWS

EE.UU. Este (Norte de Virginia) us-east-1

Última modificación

29 Apr 2025 11:17:18 AM CEST

Tamaño

151.3 KB

Tipo

jpg

Clave

[1a.jpg](#)

URI DE S3

[s3://bucket-test-pgs/1a.jpg](#)

Nombre de recurso de Amazon (ARN)

[arn:aws:s3:::bucket-test-pgs/1a.jpg](#)

Etiqueta de entidad (Etag)

[ee557fbb408b80ddf9496a31f39757de](#)

URL del objeto

[https://bucket-test-pgs.s3.us-east-1.amazonaws.com/1a.jpg](#)

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<Error>
<Code>AccessDenied</Code>
<Message>Access Denied</Message>
<RequestId>GMG627V6VGZ3PSY</RequestId>
<HostId>znQ/2wHs9u1Tk4txPF8SJAkauHl86Bw9JiVhbQInlqeRlPiskmR01bk015jlrwayfEcqg1A=</HostId>
</Error>
```

El siguiente paso que vamos a realizar es modificar el acceso al Bucket y sus ficheros. Para ello, necesitamos cambiar los permisos del Bucket.

Bloquear acceso público (configuración del bucket)

Se concede acceso público a buckets y objetos a través de listas de control de acceso (ACL), políticas de bucket, políticas de puntos de acceso o todas las anteriores. A fin de garantizar que se bloquee el acceso público a todos sus buckets y objetos de S3, active Bloquear todo acceso público. Esta configuración se aplica en exclusiva a este bucket y a sus puntos de acceso. AWS recomienda activar Bloquear todo acceso público pero, antes de aplicar cualquier otra configuración, asegúrese de que sus aplicaciones funcionarán correctamente sin acceso público. Si necesita cierto nivel de acceso público a sus buckets u objetos, puede personalizar los valores de configuración individuales a continuación para que se ajusten mejor a sus necesidades específicas de almacenamiento. [Más información](#)

Bloquear todo el acceso público

Activado

► Configuración de bloqueo de acceso público individual para este bucket

[Editar](#)

Bloquear acceso público (configuración del bucket)

Se concede acceso público a buckets y objetos a través de listas de control de acceso (ACL), políticas de bucket, políticas de puntos de acceso o todas las anteriores. A fin de garantizar que se bloquee el acceso público a todos sus buckets y objetos de S3, active Bloquear todo acceso público. Esta configuración se aplica en exclusiva a este bucket y a sus puntos de acceso. AWS recomienda activar Bloquear todo acceso público pero, antes de aplicar cualquier otra configuración, asegúrese de que sus aplicaciones funcionarán correctamente sin acceso público. Si necesita cierto nivel de acceso público a sus buckets u objetos, puede personalizar los valores de configuración individuales a continuación para que se ajusten mejor a sus necesidades específicas de almacenamiento.

Bloquear todo el acceso público

Activar esta configuración equivale a activar las cuatro opciones que aparecen a continuación. Cada uno de los siguientes ajustes son independientes entre sí.

- Bloquear el acceso público a buckets y objetos concedido a través de nuevas listas de control de acceso (ACL)**
S3 bloquera los permisos de acceso público aplicados a objetos o buckets agregados recientemente, y evitara la creación de nuevas ACL de acceso público para buckets y objetos existentes. Esta configuración no cambia los permisos existentes de S3 mediante ACL.
- Bloquear el acceso público a buckets y objetos concedido a través de cualquier lista de control de acceso (ACL)**
S3 ignorará todas las ACL que conceden acceso público a buckets y objetos.
- Bloquear el acceso público a buckets y objetos concedido a través de políticas de bucket y puntos de acceso públicas nuevas**
S3 bloquera las nuevas políticas de buckets y puntos de acceso que concedan acceso público a buckets y objetos. Esta configuración no afecta a las políticas ya existentes que permiten acceso público a los recursos de S3.
- Bloquear el acceso público y entre cuentas a buckets y objetos concedido a través de cualquier política de bucket y puntos de acceso pública**
S3 ignorará el acceso público y entre cuentas en el caso de buckets o puntos de acceso que tengan políticas que concedan acceso público a buckets y objetos.

A pesar de haber modificado los permisos y habilitar el acceso público, todavía no tenemos acceso al fichero mediante su URL.

Esto significa que los ficheros dentro del Bucket NO heredan los permisos al cambiar el acceso.

Debemos modificar el acceso al fichero mediante el propio fichero.

Para poder cambiar el acceso al fichero necesitamos habilitar Access Control List, es decir, ACL, que es el acceso de las listas a los Buckets.

Entramos primero en permisos dentro del Bucket y habilitamos ACL

Propiedad de objetos [Información](#) [Editar](#)

Controla la propiedad de los objetos escritos en este bucket desde otras cuentas de AWS y el uso de listas de control de acceso (ACL). La propiedad de los objetos determina quién puede especificar el acceso a los objetos.

ACL deshabilitadas (recomendado)
Todos los objetos de este bucket son propiedad de esta cuenta. El acceso a este bucket y sus objetos se especifica solo mediante políticas.

ACL habilitadas
Los objetos de este bucket pueden ser propiedad de otras cuentas de AWS. El acceso a este bucket y sus objetos se puede especificar mediante ACL.

⚠ Recomendamos desactivar las listas de control de acceso (ACL), a menos que necesite controlar el acceso a cada objeto individualmente o que el escritor del objeto sea el propietario de los datos que carga. Utilizar una política de bucket en lugar de ACL para compartir datos con usuarios externos a la cuenta simplifica la administración de permisos y la realización de auditorías.

⚠ **Habilitar las ACL desactiva la configuración forzada del propietario del bucket en cuanto a la propiedad del objeto**
Una vez desactivada la configuración forzada del propietario del bucket, se restablecen las listas de control de acceso (ACL) y sus permisos asociados. El acceso a los objetos que no posee se basará en las ACL y no en la política del bucket.

Reconozco que las ACL se restaurarán.

Propiedad de objetos

Propietario del bucket preferido 
Si los nuevos objetos escritos en este bucket especifican la ACL preconfigurada bucket-owner-full-control, son propiedad del propietario del bucket. De lo contrario, son propiedad del escritor de objetos.

Escritor de objetos
El escritor de objetos sigue siendo el propietario del objeto.

ⓘ Si desea aplicar la propiedad de objetos solo para objetos nuevos, la política de bucket debe especificar que la ACL preconfigurada bucket-owner-full-control es obligatoria para las cargas de objetos. [Más información](#)

[Cancelar](#) [Guardar cambios](#)

Una vez habilitado ACL, debemos entrar en el propio fichero y habilitar sus propias ACL.

1a.jpg [Información](#) [Copiar URI de S3](#) [Descargar](#) [Abrir](#) [Acciones de objetos ▾](#)

[Propiedades](#) [Permisos](#) [Versiones](#)

Información general sobre el objeto

Propietario
paco.garcia.serrano.3213

Región de AWS
EE.UU. Este (Norte de Virginia) us-east-1

Última modificación
29 Apr 2025 11:17:18 AM CEST

Tamaño
131.3 KB

Tipo
jpg

Clave
 1a.jpg

URI DE S3
 s3://bucket-test-pgs/1a.jpg

Nombre de recurso de Amazon (ARN)
 arnaws:s3:::bucket-test-pgs/1a.jpg

Etiqueta de entidad (Etag)
 ee557ffbb408b80ddff9496a31f39757de

URL del objeto
 https://bucket-test-pgs.s3.us-east-1.amazonaws.com/1a.jpg

[Descargar como](#) [Compartir con una URL prefirmada](#) [Calcular el tamaño total](#) [Copiar](#) [Trasladar](#) [Iniciar restauración](#) [Consultar con S3 Select](#)

[Editar acciones](#) [Cambiar el nombre del objeto](#) [Editar clase de almacenamiento](#) [Editar cifrado del lado del servidor](#) [Editar metadatos](#) [Editar etiquetas](#) [Hacer público mediante ACL](#)

Dentro de los buckets también tenemos control de versiones de ficheros.
Para acceder a las versiones de un fichero, primero debemos habilitar el control de versiones del Bucket.

1a.jpg [Información](#) [Copiar URI de S3](#) [Descargar](#) [Abrir](#) [Acciones de objetos ▾](#)

[Propiedades](#) [Permisos](#) [Versiones](#)

⚠ **El bucket "bucket-test-pgs" no tiene habilitado el control de versiones**
Le recomendamos que habilite el control de versiones de buckets para evitar sobrescribir o eliminar objetos involuntariamente. [Más información](#) [Habilitar control de versiones de bucket](#)

Versiones (0) [Descargar](#) [Abrir](#) [Eliminar](#) [Acciones ▾](#)

ID de versión	Tipo	Última modificación	Tamaño	Clase de almacenamiento
No hay versiones				

Este objeto no tiene versiones que mostrar porque el control de versiones del bucket no se ha habilitado para este bucket.

Al cargar las versiones por primera vez, la primera versión siempre es **null** y a medida que vayamos subiendo ficheros, veremos que irá poniendo códigos de versión a cada fichero.

1a.jpg [Información](#)

[Copiar URI de S3](#)

Propiedades | Permisos | **Versiones**

Versiones (1)

ID de versión	Tipo	Última modificación	Tamaño
null (Versión actual)	jpg	29 Apr 2025 11:17:18 AM CEST	

1a.jpg [Información](#)

[Copiar URI de S3](#)

Propiedades | Permisos | **Versiones**

Versiones (2)

ID de versión	Tipo	Última modificación	Tamaño
rudTOPACu9BCUy2t3_t2tl_m3...	jpg	29 Apr 2025 11:29:51 AM CEST	
null	jpg	29 Apr 2025 11:17:18 AM CEST	

POLITICAS EN LOS BUCKETS

La seguridad de los buckets puede ser determinada por dos formas

- 1) IAM y sus políticas de acceso a los grupos
- 2) Mediante políticas propias que podemos generar dentro del Bucket utilizando ACL.

Lo que vamos a probar es la segunda forma dentro del bucket. Generaremos un documento JSON con la política y la asociaremos a nuestro Bucket.

Pudiera ser que necesitemos solamente algunos usuarios que accedan al bucket, lo podemos realizar con ACL.

También podemos indicar cómo deseamos que se comporten los ficheros del bucket.

Lo que vamos a realizar es indicar una política ACL en la que debemos cifrar los ficheros de forma obligatoria y con el formato de cifrado SSE-S3.

Cifrado predeterminado [Información](#)

El cifrado del lado del servidor se aplica automáticamente a los nuevos objetos almacenados en este bucket.

Tipo de cifrado [Información](#)

Cifrado del servidor con claves administradas de Amazon S3 (SSE-S3)
 Cifrado del servidor con claves de AWS Key Management Service (SSE-KMS)
 Cifrado de doble capa del servidor con claves de AWS Key Management Service (DSSE-KMS)
 Proteja sus objetos con dos capas de cifrado independientes. Para obtener más información sobre los precios, consulte [DSSE-KMS pricing](#) (Precios de DSSE-KMS) en la documentación.

Clave de bucket
 El uso de una clave de bucket de SSE-KMS reduce los costos de cifrado al reducir las llamadas a AWS KMS. Las claves de bucket de SSE no son compatibles con SSE-KMS.
 Desactivar
 Habilitar

Para los cifrados se utiliza una página de los años 2.000

Entramos en los **Permisos** del bucket

Vamos a necesitar el ARN de nuestro bucket

Política de bucket

La política del bucket, escrita en JSON, proporciona acceso a los objetos almacenados en el bucket.

ARN del bucket
[arn:aws:s3:::bucket-test-pgs](#)

Editar la política del bucket [Información](#)

Política de bucket

La política del bucket, escrita en JSON, proporciona acceso a los objetos almacenados en el bucket. Las políticas de bucket no se aplican a los objetos que pertenecen a otras cuentas. [Más información](#)

ARN del bucket
[arn:aws:s3:::bucket-test-pgs](#)

Política

```
1
```

[Ejemplos de políticas](#) [Generador de políticas](#)

[Editar instrucción](#)



AWS Policy Generator

The AWS Policy Generator is a tool that enables you to create policies that control access to Amazon Web Services (AWS) products and resources. For more information about creating policies, see key concepts in [Using AWS Identity and Access Management](#). Here are sample policies.

Step 1: Select Policy Type

A Policy is a container for permissions. The different types of policies you can create are an [IAM Policy](#), an [S3 Bucket Policy](#), an [SNS Topic Policy](#), a [VPC Endpoint Policy](#), and an [SQS Queue Policy](#).

Select Type of Policy

Step 2: Add Statement(s)

A statement is the formal description of a single permission. See [a description of elements](#) that you can use in statements.

Effect Allow Deny

Principal *

Use a comma to separate multiple values.

AWS Service

Amazon S3

All Services ('*')

Use multiple statements to add permissions for more than one service.

Actions

1 Action(s) Selected

All Actions ('*')

Amazon Resource Name (ARN)

- PutLifecycleConfiguration
- PutMetricsConfiguration
- PutMultiRegionAccessPointPolicy
- PutObject
- PutObjectAcl
- PutObjectLegalHold
- PutObjectRetention
- PutObjectTagging

{BucketName}/\${KeyName}.

id. You must enter a valid ARN.

Debemos copiar nuestro ARN y podríamos indicar que solamente aplique la política a uno o varios elementos dentro del Bucket, por ejemplo, una carpeta.

Para indicar que deseamos aplicar la política a todo el conjunto de elementos dentro del bucket, debemos finalizar el ARN con /*

arn:aws:s3:::bucket-test-pgs/*

Amazon Resource Name (ARN)

arn:aws:s3:::bucket-test-pgs/*

ARN should follow the following format: arn:aws:s3:::\${BucketName}/\${KeyName}.
Use a comma to separate multiple values.

Add Conditions (Optional)

Add Statement

El siguiente paso es indicar las condiciones para denegar la subida de archivos.

Add Conditions (Optional)

Conditions are any restrictions or details about the statement.([More Details](#)).

Condition

StringNotEquals

Key

s3:x-amz-server-side-encryption

Value

SSE-S3

Add Condition

Condition

Null

Key

s3:x-amz-server-side-encryption

Value

true

Add Condition

Condition

Keys

StringNotEquals

- s3:x-amz-server-side-encryption: "SSE-S3"

Null

- s3:x-amz-server-side-encryption: "true"

Principal(s)	Effect	Action	Resource	Conditions
• *	Deny	• s3:PutObject	arn:aws:s3:::bucket-test-pgs/*	<ul style="list-style-type: none"> StringNotEquals <ul style="list-style-type: none"> s3:x-amz-server-side-encryption: "SSE-S3" Null <ul style="list-style-type: none"> s3:x-amz-server-side-encryption: "true"

Pulsamos sobre **Generate** y nos ofrecerá un JSON con la política creada

Policy JSON Document

Click below to edit. To save the policy, copy the text below to a text editor.
Changes made below will not be reflected in the policy generator tool.

```
{
  "Id": "Policy1745920616940",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1745920577853",
      "Action": [
        "s3:PutObject"
      ],
      "Effect": "Deny",
      "Resource": "arn:aws:s3:::bucket-test-pgs/*",
      "Condition": {
        "StringNotEquals": {
          "s3:x-amz-server-side-encryption": "SSE-S3"
        },
        "Null": {
          "s3:x-amz-server-side-encryption": "true"
        }
      },
      "Principal": "*"
    }
  ]
}
```

Close

Por último, copiamos la política del JSON generado y volvemos a nuestro Bucket

Editar la política del bucket información

Política de bucket

La política del bucket, escrita en JSON, proporciona acceso a los objetos almacenados en el bucket. Las políticas de bucket no se aplican a los objetos individuales.

ARN del bucket
 arn:aws:s3:::bucket-test-pgs

Política

```

1 ▾ []
2   "Id": "Policy1745920616940",
3   "Version": "2012-10-17",
4   "Statement": [
5     {
6       "Sid": "Stmt1745920577853",
7       "Action": [
8         "s3:PutObject"
9       ],
10      "Effect": "Deny",
11      "Resource": "arn:aws:s3:::bucket-test-pgs/*",
12      "Condition": {
13        "StringNotEquals": {
14          "s3:x-amz-server-side-encryption": "SSE-S3"
15        },
16        "Null": {
17          "s3:x-amz-server-side-encryption": "true"
18        }
19      },
20      "Principal": "*"
21    }
22  ]

```

Vamos a intentar subir un fichero a nuestro bucket.

Error al cargar
 Para obtener más información, consulte la columna **Error** de la tabla **Archivos y carpetas**.

Podemos comprobar que nos está dando un error debido a que no hemos indicado el cifrado del fichero, debemos indicarlo según nuestra política ACL del bucket.

Cifrado del lado del servidor [Información](#)

El cifrado del lado del servidor protege los datos en reposo.

Cifrado del lado del servidor

- No especificar una clave de cifrado

La configuración del bucket para el cifrado predeterminado se utiliza para cifrar objetos al almacenarlos en Amazon S3.

- Especificar una clave de cifrado

La clave de cifrado especificada se utiliza para cifrar objetos antes de almacenarlos en Amazon S3.

Configuración del cifrado [Información](#)

- Usar la configuración del bucket para el cifrado predeterminado

- Anular la configuración del bucket para el cifrado predeterminado

Tipo de cifrado [Información](#)

- Cifrado del servidor con claves administradas de Amazon S3 (SSE-S3)

- Cifrado del servidor con claves de AWS Key Management Service (SSE-KMS)

- Cifrado de doble capa del servidor con claves de AWS Key Management Service (DSSE-KMS)

Proteja sus objetos con dos capas de cifrado independientes. Para obtener más información sobre los precios, consulte [DSSE-KMS pricing](#) (Precios de DSSE-KMS) en la pestaña [Storage](#) (Almacenamiento) de la página

Y ya visualizaremos nuestro bucket

Se ha realizado la carga correctamente

Para obtener más información, consulte la tabla [Archivos y carpetas](#).

Por último, eliminamos el bucket que hemos utilizado en las pruebas

Nota: Para eliminar un bucket, primero debemos vaciarlo

BUCKET STATIC WEB SITE

Un bucket nos permite también poder realizar la funcionalidad para un sitio web static.

Será un alojamiento que no tiene Back y solamente podría contener ficheros Front (Diseño y JS)

Nada de Frameworks, ni React, ni Vue...

Funcionalidades:

- Alojamiento Web estático
- Generar una URL Web amigable

Al ser un sitio web público por URL necesitamos que todo el bucket sea público desde el inicio.

Configuraremos el bucket con ACL activado y siendo público desde su creación

Comenzamos creando un nuevo bucket llamado **bucket-static-web-pgs**

Configuración general

Región de AWS

EE.UU. Este (Norte de Virginia) us-east-1

Tipo de bucket [Información](#)

- Uso general

Recomendado para la mayoría de los casos de uso y patrones de acceso. Los buckets de uso general son del tipo de bucket de S3 original. Permiten una combinación de clases de almacenamiento que almacenan objetos de forma redundante en múltiples zonas de disponibilidad.

- Directorio

Recomendada Zone, que pri

Nombre del bucket [Información](#)

bucket-static-web-pgs

Los nombres de los buckets deben tener entre 3 y 63 caracteres y ser únicos dentro del espacio de nombres global. Los nombres de los buckets también deben empe

Copiar la configuración del bucket existente: [opcional](#)

Solo se copia la configuración del bucket en los siguientes ajustes.

Elegir el bucket

Formato: s3://bucket/prefijo

Propiedad de objetos [Información](#)

Controle la propiedad de los objetos escritos en este bucket desde otras cuentas de AWS y el uso de listas de control de acceso (ACL). La propiedad de los objetos determina quién puede especificar el acceso a los objetos.

- ACL deshabilitadas (recomendado)

Todos los objetos de este bucket son propiedad de esta cuenta. El acceso a este bucket y sus objetos se especifica solo mediante políticas.

- ACL habilitadas

Los objetos de este bucket pueden ser propiedad de otras cuentas de AWS. El acceso a este bucket y sus objetos se puede especificar mediante ACL.

 Recomendamos desactivar las listas de control de acceso (ACL), a menos que necesite controlar el acceso a cada objeto individualmente o que el escritor del objeto sea el propietario de los datos que carga. Utilizar una política de bucket en lugar de ACL para compartir datos con usuarios externos a la cuenta simplifica la administración de permisos y la realización de auditorías.

Propiedad de objetos

- Propietario del bucket preferido

Si los nuevos objetos escritos en este bucket especifican la ACL preconfigurada bucket-owner-full-control, son propiedad del propietario del bucket. De lo contrario, son propiedad del escritor de objetos.

- Escritor de objetos

El escritor de objetos sigue siendo el propietario del objeto.

 Si desea aplicar la propiedad de objetos solo para objetos nuevos, la política de bucket debe especificar que la ACL preconfigurada bucket-owner-full-control es obligatoria para las cargas de objetos. [Más información](#)

Configuración de bloqueo de acceso público para este bucket

Se concede acceso público a los buckets y objetos a través de listas de control de acceso (ACL), políticas de bucket, políticas de puntos de acceso o todas las anteriores. A fin de garantizar que se bloquee el acceso público a todos los recursos, Bloquear todo el acceso público. Esta configuración se aplica exclusivamente a este bucket y a sus puntos de acceso. AWS recomienda activar Bloquear todo el acceso público, pero, antes de aplicar cualquiera de estos ajustes, las aplicaciones funcionarán correctamente sin acceso público. Si necesita cierto nivel de acceso público a los buckets u objetos, puede personalizar la configuración individual a continuación para adaptarla a sus casos de uso de acuerdo con lo que necesite.

[Más información](#)

Bloquear todo el acceso público

Activar esta configuración equivale a activar las cuatro opciones que aparecen a continuación. Cada uno de los siguientes ajustes son independientes entre sí.

- Bloquear el acceso público a buckets y objetos concedido a través de nuevas listas de control de acceso (ACL)**
S3 bloqueará los permisos de acceso público aplicados a objetos o buckets agregados recientemente, y evitará la creación de nuevas ACL de acceso público para buckets y objetos existentes. Esta configuración no cambia los permisos existentes que permiten acceso público a los recursos de S3 mediante ACL.
- Bloquear el acceso público a buckets y objetos concedido a través de cualquier lista de control de acceso (ACL)**
S3 ignorará todas las ACL que conceden acceso público a buckets y objetos.
- Bloquear el acceso público a buckets y objetos concedido a través de políticas de bucket y puntos de acceso públicas nuevas**
S3 bloqueará las nuevas políticas de buckets y puntos de acceso que concedan acceso público a buckets y objetos. Esta configuración no afecta a las políticas ya existentes que permiten acceso público a los recursos de S3.
- Bloquear el acceso público y entre cuentas a buckets y objetos concedido a través de cualquier política de bucket y puntos de acceso pública**
S3 ignorará el acceso público y entre cuentas en el caso de buckets o puntos de acceso que tengan políticas que concedan acceso público a buckets y objetos.

Desactivar el bloqueo de todo acceso público puede provocar que este bucket y los objetos que contiene se vuelvan públicos

AWS recomienda que active la opción para bloquear todo el acceso público, a menos que se requiera acceso público para casos de uso específicos y verificados, como el alojamiento de sitios web estáticos.

Reconozco que la configuración actual puede provocar que este bucket y los objetos que contiene se vuelvan públicos.

El hecho de haber habilitado ACL y quitar lo de público no convierte en Public nuestro bucket. Debemos crear una política de acceso a los elementos del bucket para que cuando alguien realice un GetObject, se le permita el acceso.



AWS Policy Generator

The AWS Policy Generator is a tool that enables you to create policies that control access to **Amazon Web Services (AWS)** services. For more information about creating policies, see [key concepts in Using AWS Identity and Access Management](#). Here are some steps to get started:

Step 1: Select Policy Type

A Policy is a container for permissions. The different types of policies you can create are an [IAM Policy](#), an [S3 Bucket Policy](#), a [VPC Endpoint Policy](#), and an [SQS Queue Policy](#).

Select Type of Policy S3 Bucket Policy

Step 2: Add Statement(s)

A statement is the formal description of a single permission. See [a description of elements](#) that you can use in statements.

Effect Allow Deny

Principal *

Use a comma to separate multiple values.

AWS Service Amazon S3

All Services

Use multiple statements to add permissions for more than one service.

Actions 1 Action(s) Selected

All Actions ('*')

Amazon Resource Name (ARN)

GetMultiRegionAccessPointRoutes
 GetObject
 GetObjectAcl
 GetObjectAttributes
 GetObjectLegalHold

`{BucketName}/{Key}`

A statement is the formal description of a single permission. See [a description of elements](#) that you can use in statements.

Effect Allow Deny

Principal *

Use a comma to separate multiple values.

AWS Service Amazon S3

All Services ('*')

Use multiple statements to add permissions for more than one service.

Actions 1 Action(s) Selected

All Actions ('*')

Amazon Resource Name (ARN)

`s3:::bucket-static-web-pgss/`

ARN should follow the following format: `arn:aws:s3:::{BucketName}/{Key}`
Use a comma to separate multiple values.

Add Conditions (Optional)

Add Statement

You added the following statements. Click the button below to Generate a policy.

Principal(s)	Effect	Action	Resource	Conditions
*	Allow	s3:GetObject	arn:aws:s3:::bucket-static-web-pgss/*	None

Policy JSON Document

Click below to edit. To save the policy, copy the text below to a text editor.
Changes made below will not be reflected in the policy generator tool.

```
{  
  "Id": "Policy1745927517342",  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "Stmt1745927498759",  
      "Action": [  
        "s3:GetObject"  
      ],  
      "Effect": "Allow",  
      "Resource": "arn:aws:s3:::bucket-static-web-pgss/*",  
      "Principal": "*"  
    }  
  ]  
}
```

Editar la política del bucket Información

Política de bucket

La política del bucket, escrita en JSON, proporciona acceso a los objetos almacenados en el bucket. Las políticas de bucket no se aplican a las operaciones de administración.

ARN del bucket

arn:aws:s3:::bucket-static-web-pgss

Política

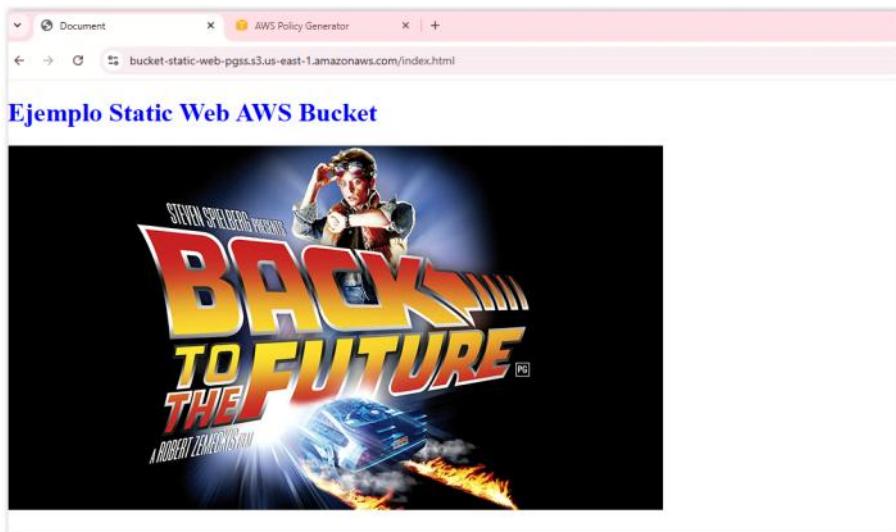
```
1▼ {  
2   "Id": "Policy1745927517342",  
3   "Version": "2012-10-17",  
4   "Statement": [  
5     {  
6       "Sid": "Stmt1745927498759",  
7       "Action": [  
8         "s3:GetObject"  
9       ],  
10      "Effect": "Allow",  
11      "Resource": "arn:aws:s3:::bucket-static-web-pgss/*",  
12      "Principal": "*"  
13    }  
14  ]  
15 }
```

Una vez que hemos comprobado que nuestro bucket es público y los recursos que subamos también, es el momento de generar una página HTML y apuntaremos a la imagen que tenemos subida en el bucket.

Creamos una página llamada index.html

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
  <meta charset="UTF-8">  
  <meta name="viewport" content="width=device-width, initial-scale=1.0">  
  <title>Document</title>  
</head>  
<body>  
  <h1 style="color: blue">Ejemplo Static Web AWS Bucket</h1>  
    
</body>  
</html>
```

Una vez creada la página, subimos nuestro fichero HTML al servidor de bucket y podremos visualizar el acceso.



Podemos visualizar la página sin problemas, pero en realidad esto no es un Hosting ni se le parece, ya que nuestra URL no deja de ser visible en todo momento para cada fichero:

www.marca.com/

No hemos configurado nuestro bucket para que sea un sitio Web Static. Debemos configurarlo para que nos genere una Web amigable y que podamos acceder desde una URL como Hosting.

Entramos dentro del bucket y en **Propiedades**

Alojamiento de sitios web estáticos

Utilice este bucket para alojar un sitio web o redireccionar solicitudes. [Más información](#)

Editar

- ⓘ Recomendamos usar AWS Amplify Hosting para el alojamiento de sitios web estáticos
Implemente rápidamente un sitio web rápido, seguro y confiable con AWS Amplify Hosting. Obtenga más información sobre [Amplify Hosting](#) o [consulte sus aplicaciones de Amplify existentes](#)

[Crear la aplicación de Amplify](#)

Alojamiento de sitios web estáticos de S3

Deshabilitada

Editar alojamiento de sitios web estáticos [Información](#)

Alojamiento de sitios web estáticos

Utilice este bucket para alojar un sitio web o redireccionar solicitudes. [Más información](#)

- Desactivar
 Habilitar

Tipo de alojamiento

- Alojar un sitio web estático
Utilice el punto de enlace del bucket como dirección web. [Más información](#)
 Redirigir las solicitudes de un objeto
Redirija las solicitudes a otro bucket o dominio. [Más información](#)

ⓘ Para que sus clientes puedan obtener acceso al contenido en el punto de enlace del sitio web, debe hacer que todo el contenido sea legible públicamente
acceso público de S3 del bucket. Para obtener más información, consulte [Utilizar Bloquear acceso público de Amazon S3](#)

Documento de índice

Especifique la página predeterminada o de inicio del sitio web.

index.html

Documento de error - opcional

Este se devuelve cuando se produce un error.

error.html

Reglas de redirecciónamiento: opcionales

Redireccione las reglas, escritas en JSON, para redirigir automáticamente las solicitudes de páginas web de contenido específico. [Más información](#)

Alojamiento de sitios web estáticos

Utilice este bucket para alojar un sitio web o redireccionar solicitudes. [Más información](#)

- ⓘ Recomendamos usar AWS Amplify Hosting para el alojamiento de sitios web estáticos
Implemente rápidamente un sitio web rápido, seguro y confiable con AWS Amplify Hosting. Obtenga más información sobre [Amplify Hosting](#) o [consulte sus aplicaciones de Amplify existentes](#)

Alojamiento de sitios web estáticos de S3

Habilitada

Tipo de alojamiento

Alojamiento de buckets

Punto de enlace de sitio web del bucket

Al configurar su bucket como sitio web estático, el sitio web estará disponible en el punto de enlace del sitio web específico de la región de AWS del bucket. [Más información](#)

<http://bucket-static-web-pgss.s3-website-us-east-1.amazonaws.com>

Y ya podemos comprobar cómo genera la URL para el Web Site como Hosting



Por último, si tuviéramos que permitir accesos a nuestro Web Site, también tenemos la posibilidad de habilitar y utilizar CORS como si de un App Service de Azure fuera.

Para habilitar CORS, se realiza desde la pestaña **Permisos** de nuestro Bucket

Uso compartido de recursos entre orígenes (CORS)

La configuración CORS, escrita en JSON, define una manera para que las aplicaciones web de los clientes cargadas en un dominio interactúen con los recursos de un dominio diferente. [Más información](#)

No hay configuraciones que mostrar.

[Copiar](#)

Tendríamos que incluir un acceso CORS mediante JSON. Ejemplo:

```
[  
  {  
    "AllowedHeaders": [  
      "*"  
    ],  
    "AllowedMethods": [  
      "PUT",  
      "POST",  
      "DELETE"  
    ],  
    "AllowedOrigins": [  
      "http://www.example.com"  
    ],  
    "ExposeHeaders": [  
      "x-amz-server-side-encryption",  
      "x-amz-request-id",  
      "x-amz-id-2"  
    ],  
    "MaxAgeSeconds": 3000  
  }  
]
```

Eliminamos el bucket creado actualmente

UTILIZAR S3 DESDE NET CORE

Para poder programar cualquier servicio en AWS es necesario tener un usuario que tenga permisos en el equipo de trabajo. Si hablamos de permisos, estamos hablando de IAM

Para trabajar con programación en nuestro equipo será necesario utilizar varios elementos:

- 1) Usuario IAM con permisos sobre **AmazonS3FullAccess**, o un grupo con dichos permisos. Este usuario IAM no es necesario que tenga acceso a la Consola de AWS.
- 2) **AWS Command Line Interface**. Esto es la consola para acceder a los recursos de AWS. Es muy útil y utilizado en elementos como EC2.

<https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

Vamos a comenzar creando un nuevo grupo en IAM con permisos para acceder a nuestro servicio S3

Creamos un nuevo grupo llamado **grupo-s3** con los siguientes permisos:

<input type="checkbox"/>	<input checked="" type="checkbox"/>  AmazonDMSRedshift...	Administrada por AWS	Nin
<input checked="" type="checkbox"/>	<input type="checkbox"/>  AmazonS3FullAccess	Administrada por AWS	Poli

El siguiente paso que vamos a realizar es creamos un nuevo usuario llamado **puent** al que incluiremos en el Grupo.

Este usuario NO tendrá acceso a la Consola, debemos hacerlo de otra forma para que acceda mediante **AWS CLI**.

Especificar los detalles de la persona

Detalles de la persona

Nombre de usuario

El nombre de usuario puede tener un máximo de 64 caracteres. Caracteres válidos: A-Z, a-z, 0-9, and + - . @ _ - (guion)

Proporcione acceso de usuario a la consola de administración de AWS: opcional
Si proporciona acceso a la consola a una persona, se trata de un [práctica recomendada](#) para administrar su acceso en IAM Identity Center.

i Si está creando acceso mediante programación a través de claves de acceso o credenciales específicas de servicios para AWS CodeCommit o Amazon Keypairs, puede generarlos después de crear este usuario de IAM. [Más información](#)

[Cancelar](#)

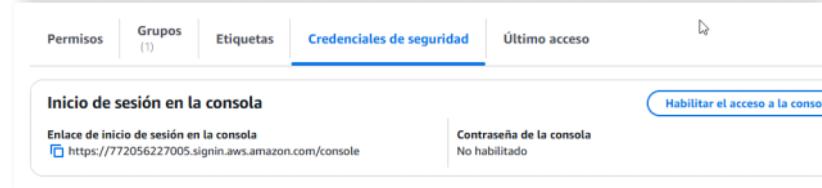
[Siguiente](#)



Actualmente esta persona no tiene ninguna posibilidad de acceder a AWS.

Debemos generar unas credenciales de seguridad para que el usuario pueda acceder mediante CLI y solamente

Tendrá acceso al recurso de S3.



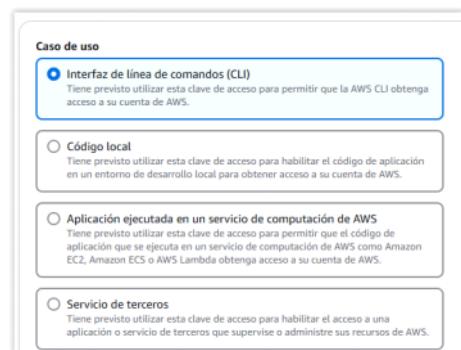
Claves de acceso (0)

Utilice las claves de acceso para enviar llamadas mediante programación a AWS desde AWS CLI, Herramientas de AWS para PowerShell, AWS SDK o llamadas directas a la API de AWS. Puede tener un máximo de dos claves de acceso (activas o inactivas) a la vez. [Más información](#)

No hay claves de acceso. Como práctica recomendada, evite el uso de credenciales a largo plazo, como las claves de acceso. En su lugar, utilice herramientas que proporcionen credenciales a corto plazo. [Más información](#)

[Crear clave de acceso](#)

[Crear clave de acceso](#)



Establecer el valor de etiqueta de descripción - opcional [Información](#)

La descripción de esta clave de acceso se adjuntará a esta persona como una etiqueta y se mostrará junto con la clave de acceso.

Valor de etiqueta de descripción

Describa el objetivo de esta clave de acceso y dónde se utilizará. Una buena descripción lo ayudará a rotar esta clave de acceso con confianza más adelante.

Máximo de 256 caracteres. Los caracteres permitidos son letras, números, espacios representables en UTF-8 y: _ : / = + - @

[Cancelar](#)

[Anterior](#)

[Crear clave de acceso](#)

Clave de acceso

Si pierde u olvida la clave de acceso secreta, no podrá recuperarla. En su lugar, cree una nueva clave de acceso y deje inactiva la antigua.

Clave de acceso Clave de acceso secreta

Mostrar

Con estas claves que acabamos de generar ya podemos programar en nuestro equipo local, siempre que nos validemos con este usuario. Solamente tendremos permisos de acceso con este usuario a S3.

Para validarnos en el equipo necesitamos varios elementos:

- 1) AWS Access Keys
- 2) AWS Secret Access Key
- 3) Default Region: us-east-1
- 4) Format: json

[Estados Unidos \(Norte de Virginia\) ▾](#)

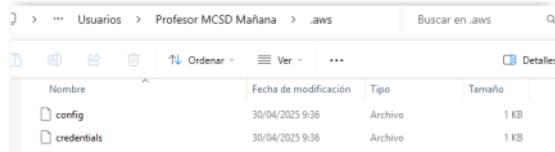
Abrimos cmd y ejecutamos el siguiente comando:

aws configure

```
C:\> aws configure
AWS Access Key ID [None]: AKIA3HQRDDS6WLJL3PYK
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json
```

Acabamos de dar de alta un usuario de AWS en nuestro Windows del aula.
 Las credenciales se quedan almacenadas en el equipo, nosotros lo que haremos es eliminar los usuarios que vayamos
 Creando dentro de AWS.
 Siempre, por si acaso, eliminamos las credenciales en Windows si no hemos terminado la práctica.
 Las credenciales van por usuario, pero no siempre es así.

Las credenciales están en nuestro equipo y en nuestro usuario en una carpeta llamada .aws



Vamos a comenzar trabajando con un bucket, creamos un nuevo bucket llamado **bucket-puente-pgs**
 Y lo vamos a poner como **public**.

ACL habilitadas
 Los objetos de este bucket pueden ser propiedad de otras cuentas de AWS. El acceso a este bucket y sus objetos se puede especificar mediante ACL.

Configuración de bloqueo de acceso público para este bucket

Se concede acceso público a los buckets y objetos a través de listas de control de acceso (ACL), políticas de buqueo que bloquee el acceso público a todos sus buckets y objetos, active Bloquear todo el acceso público. Esta configuración activa Bloquear todo el acceso público, pero, antes de aplicar cualquiera de estos ajustes, asegúrese de que el acceso público a los buckets u objetos, puede personalizar la configuración individual a continuación para cada uno.

Bloquear todo el acceso público
 Activar esta configuración equivale a activar las cuatro opciones que aparecen a continuación. Cada uno de los siguientes

Effect Allow Deny

Principal

AWS Service All Services (*)

Actions 1 Action(s) Selected All Actions (*)

Amazon Resource Name (ARN) d. You must enter a valid ARN.

Step 2: Add Statement(s)

A statement is the formal description of a single permission. See a [description of elements](#) that you can use in statements.

Effect Allow Deny

Principal

AWS Service All Services (*)

Actions 1 Action(s) Selected All Actions (*)

Amazon Resource Name (ARN) ARN should follow the following format: arn:aws:s3:::\${BucketName}/\${KeyName}. Use a comma to separate multiple values.

Add Conditions (Optional)

Add Statement

Principal(s)	Effect	Action	Resource	Conditions
* *	Allow	s3:GetObject	arn:aws:s3:::bucket-puente-pgs/*	None

Una vez que tenemos el bucket, es el momento de programar.

Creamos una aplicación **MvcNetCoreAWSS3**

 Microsoft.IdentityModel.Tokens by AzureAD, Microsoft, 2,548 downloads 8.9.0
 Includes types that provide support for SecurityTokens, Cryptographic operations: Signing, Verifying Signatures, Encryption.

 AWSSDK.Extensions.NETCore.Setup by awsdotnet, 155M downloads 4.0.0
 Extensions for the AWS SDK for .NET to integrate with .NET Core configuration and dependency injection frameworks.

 AWSSDK.S3 by awsdotnet, 366M downloads 4.0.0
 Amazon Simple Storage Service (Amazon S3), provides developers and IT teams with secure, durable, highly-scalable object storage.

Vamos a recibir el nombre del bucket a partir del **appsettings.json**

Creamos una carpeta llamada **Services** y una clase llamada **ServiceStorageS3**

SERVICESTORAGE53

```
public class ServiceStorageS3
{
    //VAMOS A RECIBIR EL NOMBRE DEL BUCKET
    //A PARTIR DEL APPSETTINGS.JSON
    private string BucketName;

    //LA CLASE/INTERFACE PARA TRABAJAR CON LOS BUCKETS
    //SE LLAMA IAmazonS3 Y VAMOS A RECIBIRLA MEDIANTE
    //INYECCION
    private IAmazonS3 ClientS3;

    public ServiceStorageS3(IConfiguration configuration
        , IAmazonS3 clientS3)
    {
        this.BucketName = configuration.GetValue<string>
            ("AWS:BucketName");
        this.ClientS3 = clientS3;
    }

    //COMENZAMOS CREANDO UN METODO PARA SUBIR FICHEROS
    public async Task<bool> UploadFileAsync
        (string fileName, Stream stream)
    {
        PutObjectRequest request = new PutObjectRequest
        {
            Key = fileName,
            BucketName = this.BucketName,
            InputStream = stream
        };
        //PARA TRABAJAR SE UTILIZA LA CLASE IAmazonS3
        //CON UNA PETICION DE PUTOBJECT
        PutObjectResponse response = await
            this.ClientS3.PutObjectAsync(request);
        if (response.HttpStatusCode == HttpStatusCode.OK)
        {
            return true;
        }
        else
        {
            return false;
        }
    }

    public async Task<bool> DeleteFileAsync
        (string fileName)
    {
        DeleteObjectResponse response = await
            this.ClientS3.DeleteObjectAsync
            (this.BucketName, fileName);
        if (response.HttpStatusCode == HttpStatusCode.OK)
        {
            return true;
        }
        else
        {
            return false;
        }
    }

    //METODO PARA RECUPERAR TODOS LOS FICHEROS DEL BUCKET
    //AUNQUE NO TENGAMOS VERSIONES, DEBEMOS INDICAR LAS VERSIONES
    //EN EL RECORRIDO DE LOS FICHEROS
    public async Task<List<string>> GetVersionsFileAsync()
    {
        ListVersionsResponse response = await
            this.ClientS3.ListVersionsAsync(this.BucketName);
        //EXTREMOS TODAS LAS MEYS DE NUESTROS FICHEROS, ES DECIR,
        //EL NOMBRE DE TODOS LOS FICHEROS. POR DEFECTO NOS
        //DEVUELVE LA ULTIMA VERSION
        List<string> fileNames =
            response.Versions.Select(x => x.Key).ToList();
        return fileNames;
    }
}
```

El siguiente paso es crear un controlador llamado **AWSFilesController**

AWSFILESCONTROLLER

```
public class AWSFilesController : Controller
{
    private ServiceStorageS3 service;

    public AWSFilesController(ServiceStorageS3 service)
    {
        this.service = service;
    }

    public async Task<IActionResult> Index()
    {
        List<string> filesS3 = await
            this.service.GetVersionsFileAsync();
        return View(filesS3);
    }

    public IActionResult UploadFile()
    {
        return View();
    }

    [HttpPost]
    public async Task<IActionResult> UploadFile
        (IFormFile file)
    {
        using (Stream stream = file.OpenReadStream())
        {
            await this.service.UploadFileAsync
                (file.FileName, stream);
        }
        return RedirectToAction("Index");
    }

    public async Task<IActionResult> DeleteFile(string filename)
    {
        await this.service.DeleteFileAsync(filename);
        return RedirectToAction("Index");
    }
}
```

Incluimos el nombre del bucket dentro de **appsettings.json**

```
        },
        "AllowedHosts": "*",
        "AWS": {
            "BucketName": "bucket-puente-pgs"
        }
    ]
```

Resolvemos las dependencias dentro de **Program**

Cuando utilizamos servicios de AWS, se utiliza el método **AddAWSService**

PROGRAM

```

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddAWSService<IAmazonS3>();
builder.Services.AddTransient<ServiceStorageS3>();
builder.Services.AddControllersWithViews();

var app = builder.Build();

```

Y podemos comprobar la funcionalidad

Para la URL, tenemos en AWS la ruta a nuestro File, que siempre será la misma:

Información general sobre el objeto

Propietario	URI DE S3
paco.garcia.serrano.3213	s3://bucket-puente-pgs/2.jpg
Región de AWS	Nombre de recurso de Amazon (ARN)
EE.UU. Este (Norte de Virginia) us-east-1	arn:aws:s3:::bucket-puente-pgs
Última modificación	Etiqueta de entidad (Etag)
30 Apr 2025 9:51:48 AM CEST	47d298bb6a0f0e90faeadb9cfba9fb8
Tamaño	URL del objeto
37.3 KB	https://bucket-puente-pgs.s3.us-east-1.amazonaws.com/2.jpg
Tipo	
jpg	
Clave	
2.jpg	

```

<li class="list-group-item">
    @name
    
        Delete
    </a>
</li>

```

 Home Privacy AWS S3 Service

AWS Storage S3

[Upload File AWS](#)

	Delete
	Delete

Con esta forma, nuestras URLs son estáticas, si cambiamos de Bucket o de servicio
Tenemos que estar retocando todas las vistas.

VERSION 2

Con inyección en las Vistas, lo que haremos será recuperar la URL.

Incluimos, dentro de `appsettings.json` nuestra URL de acceso al Bucket

APPSETTINGS.JSON

```

    },
    "AllowedHosts": "*",
    "AWS": {
        "BucketName": "bucket-puente-pgs",
        "BucketUrl": "https://bucket-puente-pgs.s3.us-east-1.amazonaws.com/"
    }
}

```

Para utilizar `IConfiguration` en nuestras vistas, creamos el objeto mediante `@inject`

_VIEWIMPORTS.CSHTML

```

@using MvcNetCoreAWSS3
@using MvcNetCoreAWSS3.Models
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers
@inject IConfiguration configuration

```

Por último, recuperamos la URL dentro de `Index.cshtml`

INDEX.CSHTML

```

@model List<string>

#{@
    string bucketUrl =
        configuration.GetValue<string>("AWS:BucketUrl");
}

<h1>AWS Storage S3</h1>

@foreach (string name in Model)
<li class="list-group-item">
    @name
    
    <a href="#" asp-controller="AWSFiles">View</a>

```

También podemos recuperar ficheros si fueran privados.
Es muy simple, solamente debemos aplicar `GetObjectRequest`

Probamos la funcionalidad como si el Bucket fuera privado.

SERVICESSTORAGE52

```

//METODO PARA RECUPERAR UN FICHERO SI NO FUERA
//PUBLICO NUESTRO BUCKET
0 references
public async Task<Stream> GetPrivateFileAsync(string fileName)
{
    GetObjectResponse response = await
        this.ClientS3.GetObjectAsync
        (this.BucketName, fileName);
    return response.ResponseStream;
}

```

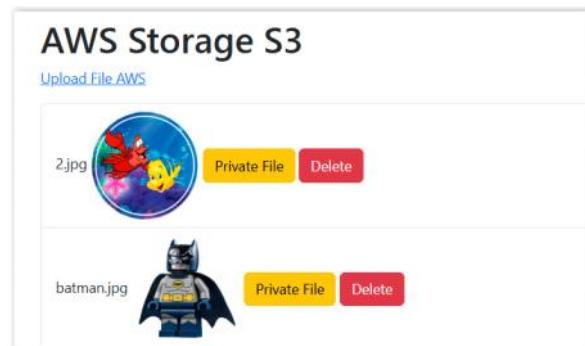
AWSFILESCONTROLLER

```

public async Task<IActionResult> PrivateFile(string filename)
{
    Stream stream = await
        this.service.GetPrivateFileAsync(filename);
    return File(stream, "image/png");
}

```

Y podremos visualizar la funcionalidad cómo si fueran imágenes privadas.



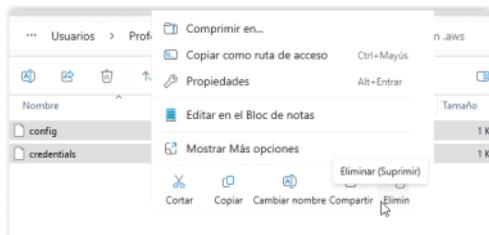
Por último, nos queda visualizar los permisos



Si quitamos al usuario `puente` del grupo `s3`...

Personas	Permisos	Access Advisor
Personas de este grupo (1/1)		
<input checked="" type="checkbox"/> Nombre de usuario	Grupos	Última acti...
<input checked="" type="checkbox"/> puente	1 Ninguno	hace 2 horas

Si eliminamos las claves...



An unhandled exception occurred while processing the request.

AmazonClientException: Failed to resolve AWS credentials. The credential providers used to search for credentials returned the

Exception 1 of 4: The environment variables AWS_ACCESS_KEY_ID/AWS_SECRET_ACCESS_KEY/AWS_SESSION_TOKEN were not
credentials.

Exception 2 of 4: The webIdentityTokenFile must be an absolute path. (Parameter 'webIdentityTokenFile')

Exception 3 of 4: Unable to find the "default" profile.

Exception 4 of 4: Failed to connect to EC2 instance metadata to retrieve credentials: Unable to get IAM security credentials from
Metadata Service..

Amazon.Runtime.Credentials.DefaultAWSCredentialsIdentityResolver.InternalGetCredentials()

Eliminamos todos los recursos de IAM y S3

AWS EC2

lunes, 5 de mayo de 2025 9:30

EC2 Elastic Cloud Computer. Son las máquinas virtuales dentro de AWS.

En este entorno Cloud no tenemos App Service, existe algo "parecido" pero funciona a partir De máquinas virtuales y, si lo hacemos mal, podríamos tener coste, es decir, no lo controlamos 100%

Si queremos montar un servicio en producción, tenemos que ir por este camino.

Los usuarios que van a desplegar las aplicaciones en las máquinas virtuales necesitan permisos EC2.

Dependiendo de los servicios que utilice nuestra App, debemos habilitar permisos propios en La máquina.

En la capa gratuita tenemos 750 horas gratuitas



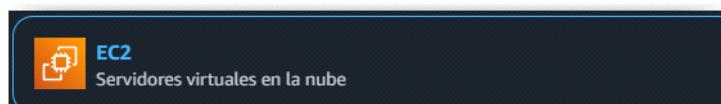
Nos van a cobrar por uso y capacidad. Existen máquinas gratuitas que entran en el plan Free.

Nota: Si tenemos la máquina creada y está apagada, NO COBRAN.
Siempre apagamos la máquina después de jugar con ella

Dentro de las máquinas se utilizan **AMI**, (Amazon Machine Image) que son el SO que contendrá Nuestra máquina. Podemos hasta crear nuestros propios AMI como si de Docker hablásemos, Pero en imágenes de equipos.

Las máquinas son creadas por zona geográfica. Seguimos trabajando en Virginia.

MAQUINA EC2 CON SERVIDOR WEB



A screenshot of the AWS EC2 Instances page. The top navigation bar includes 'Instancias', 'Información', 'Última actualización' (Hace less than a minute), 'Conectar', 'Estado de la instancia', 'Acciones', and a prominent blue 'Lanzar instancias' button. Below the navigation is a search bar with placeholder 'Buscar Instancia por atributo o etiqueta (case-sensitive)' and a dropdown for 'Todos los estados'. The main content area displays a message 'No hay instancias' and 'No tiene ninguna instancia en esta región'. A blue 'Lanzar instancias' button is centered in this area. At the bottom, there is a section titled 'Seleccione una instancia'.

Comenzamos creando una nueva máquina llamada **linux-web-server**

Nota: Dependiendo de la máquina virtual que seleccionemos, los comandos para Desplegar aplicaciones o conectar servicios dentro de Linux, son distintos.

Apto para la capa gratuita ▾

Inicio rápido

Amazon Linux
macOS
Ubuntu
Windows
Red Hat
SUSE Linux
Del > del

Buscar más AMI
Inclusión de AMI de AWS, Marketplace y la comunidad

Imágenes de máquina de Amazon (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type
ami-085386e29e44dacd7 (64 bits (x86)) / ami-00bcd7ae558b8179f (64 bits (Arm))
Virtualización: hvm Activado para ENA: true Tipo de dispositivo raíz: ebs

Apto para la capa gratuita ▾

▼ Tipo de instancia [Información](#) | [Obtener asesoramiento](#)

Tipo de instancia

t2.micro
Familia: t2 1 vCPU 1 GiB Memoria Generación actual: true
Bajo demanda Windows base precios: 0.0162 USD por hora
Bajo demanda Ubuntu Pro base precios: 0.0134 USD por hora
Bajo demanda SUSE base precios: 0.0116 USD por hora
Bajo demanda RHEL base precios: 0.026 USD por hora
Bajo demanda Linux base precios: 0.0116 USD por hora

Apto para la capa gratuita ▾

Todas las generaciones
[Comparar tipos de instancias](#)

Se aplican costos adicionales a las AMI con software preinstalado

En las máquinas virtuales necesitamos Keys para poder conectar a dichas máquinas mediante SSH.

Necesitamos unas claves públicas y privadas para trabajar.

Nota: Deberíamos guardar nuestras claves no solamente en el equipo dónde trabajemos, También subirlas a algún sitio. (One Drive)

Creamos un par de claves para acceder a la máquina llamadas

Crear par de claves

Nombre del par de claves
Con los pares de claves es posible conectarse a la instancia de forma segura.
keys-linux-web-server

El nombre puede incluir hasta 255 caracteres ASCII. No puede incluir espacios al principio ni al final.

Tipo de par de claves

RSA
Par de claves pública y privada cifradas mediante RSA

ED25519
Par de claves privadas y públicas cifradas ED25519

Formato de archivo de clave privada

.pem
Para usar con OpenSSH

.ppk
Para usar con PuTTY

⚠️ Cuando se le solicite, almacene la clave privada en un lugar seguro

[Cancelar](#) [Crear par de claves](#)

SECURITY GROUPS

El siguiente paso es crear grupos de seguridad.

Estos grupos son algo muy importante dentro de los servicios de comunicación en AWS.

Los grupos permiten conectar múltiples servicios entre sí y dentro de la misma red.

Un grupo podría pertenecer a múltiples servicios y, de esa forma, comunicar todos entre sí.

Tenemos una máquina EC2 con un MVC en su interior y tenemos otro EC2 con un API en su interior y deseamos que puedan comunicarse. Para poder comunicar estas máquinas simplemente tendríamos que introducirlas dentro del mismo grupo de seguridad.

Creamos un nuevo grupo llamado **grupo-linux-web-server**

▼ **Configuraciones de red** | [Información](#)

Red | [Información](#)
vpc-066662b1b43cc56c0

Subred | [Información](#)
Sin preferencias (subred predeterminada en cualquier zona de disponibilidad)

[Crear grupo de seguridad](#) | [Información](#)

Firewall (grupos de seguridad) | [Información](#)
Un grupo de seguridad es un conjunto de reglas de firewall que controlan el tráfico de la instancia. Agregue reglas para permitir que un tráfico específico llegue a la instancia.

[Crear grupo de seguridad](#) | [Información](#)

[Seleccionar un grupo de seguridad existente](#) | [Información](#)

Nombre del grupo de seguridad - obligatorio
grupo-linux-web-server

Este grupo de seguridad se agrega a todas las interfaces de red. El nombre no se puede editar después de crear el grupo de seguridad. La longitud máxima es de 255 caracteres. Caracteres válidos: a-z, A-Z, 0-9, espacios y _-:/()#,@[]+=&;!\$*

Descripción - obligatorio | [Información](#)
Grupo-para-primer-a-maquina-virtual

Abrimos el puerto **80** para poder comunicar con nuestra máquina mediante **HTTP**

direcciones IP conocidas.

[Agregar regla del grupo de seguridad](#)

▼ Regla del grupo de seguridad 2 (TCP, 80, Puerto 80 open) [Eliminar](#)

Tipo	Protocolo	Intervalo de puertos
HTTP	TCP	80
Tipo de origen	Origen	Descripción - opcional
Personalizada	<input type="text"/> Agregue CIDR, lista de prefijos o g	Puerto 80 open

⚠️ Las reglas con origen 0.0.0.0/0 permiten que todas las direcciones IP tengan acceso a la instancia. Le recomendamos que configure las reglas del grupo de seguridad para permitir el acceso únicamente desde direcciones IP conocidas.

[Agregar regla del grupo de seguridad](#)

Tipo | Información

HTTP

Protocolo | Información

TCP

Intervalo de puertos | Información

80

Tipo de origen | Información

Personalizada

Origen | Información

Agregue CIDR, lista de prefijos o grupo de seguridad

Descripción - opcional | Información

Puerto 80 open

0.0.0.0/0 X

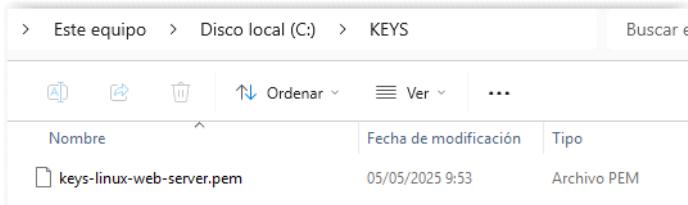
⚠️ Las reglas con origen 0.0.0.0/0 permiten que todas las direcciones IP tengan acceso a la instancia. Le recomendamos que configure las reglas del grupo de seguridad para permitir el acceso únicamente desde direcciones IP conocidas.

[Agregar regla del grupo de seguridad](#)

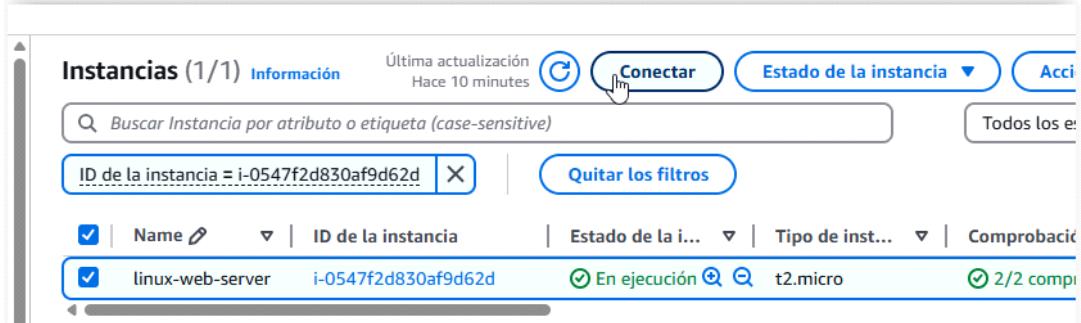
Una vez creada la máquina, podremos conectar cuando nos indique

	Name	ID de la instancia	Estado de la i...	Tipo de inst...	Comprobación de	Estado de la al:	Zona de dis
<input type="checkbox"/>	linux-web-server	i-0547f2d830af9d62d	En ejecución	t2.micro	2/2 comprobacion	Ver alarmas	us-east-1c

Vamos a copiar nuestro fichero PEM dentro de la carpeta C:\KEYS



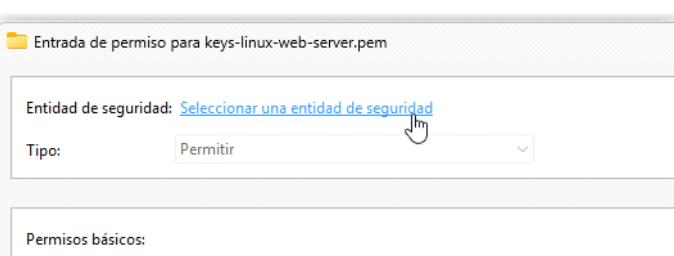
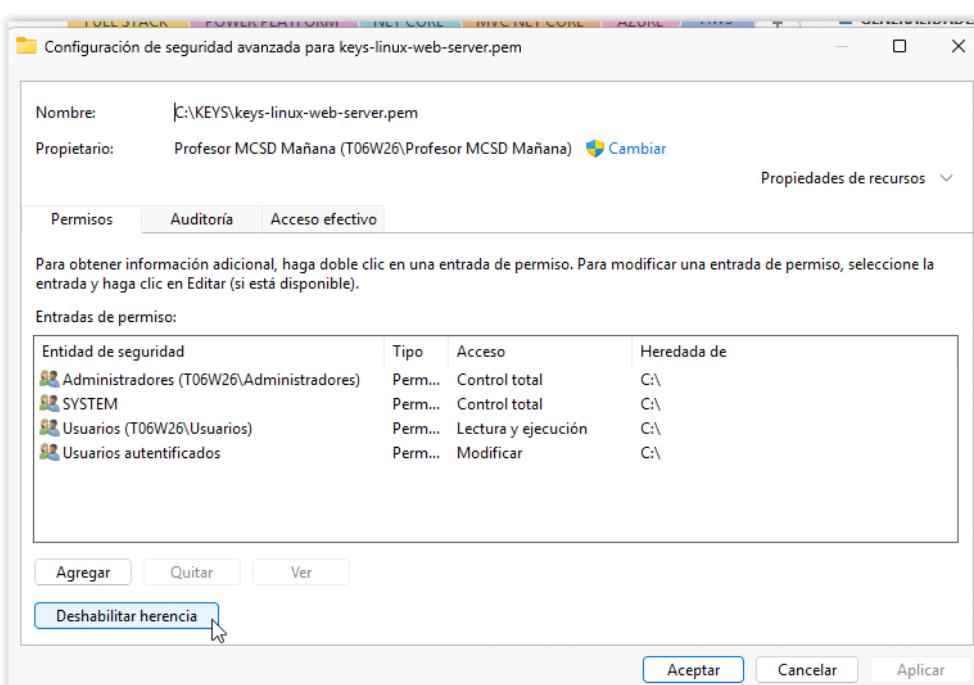
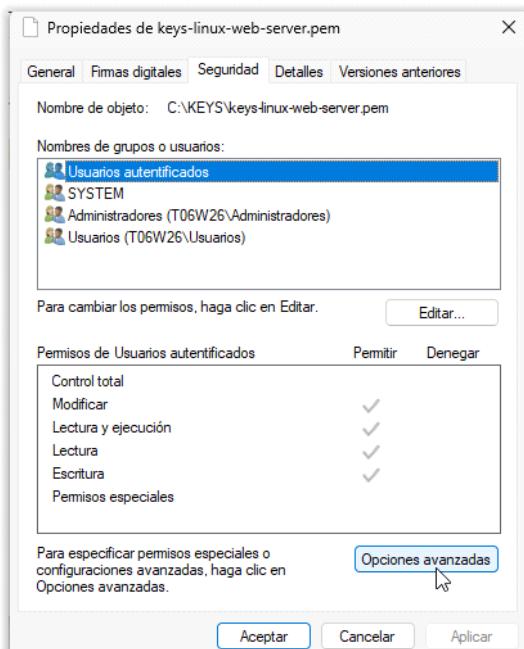
Para averiguar cómo podemos conectar a la máquina mediante el fichero pem, debemos entrar Dentro de la máquina y en la opción de Conectar

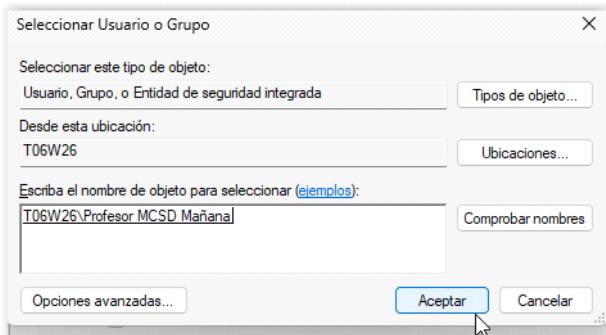


3. Ejecute este comando, si es necesario, para garantizar que la clave no se pueda ver públicamente.

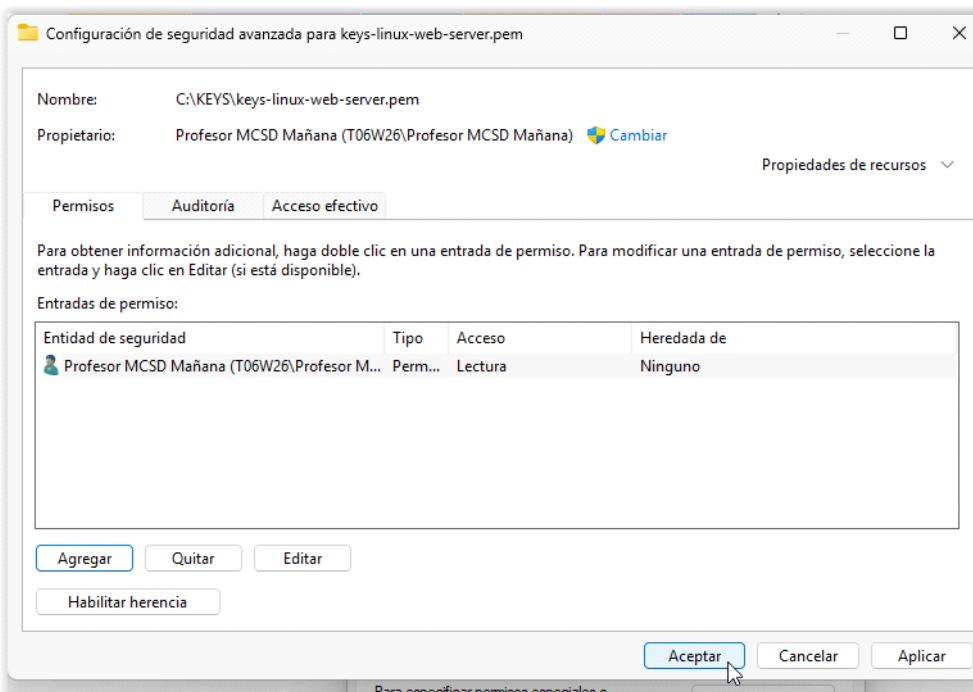
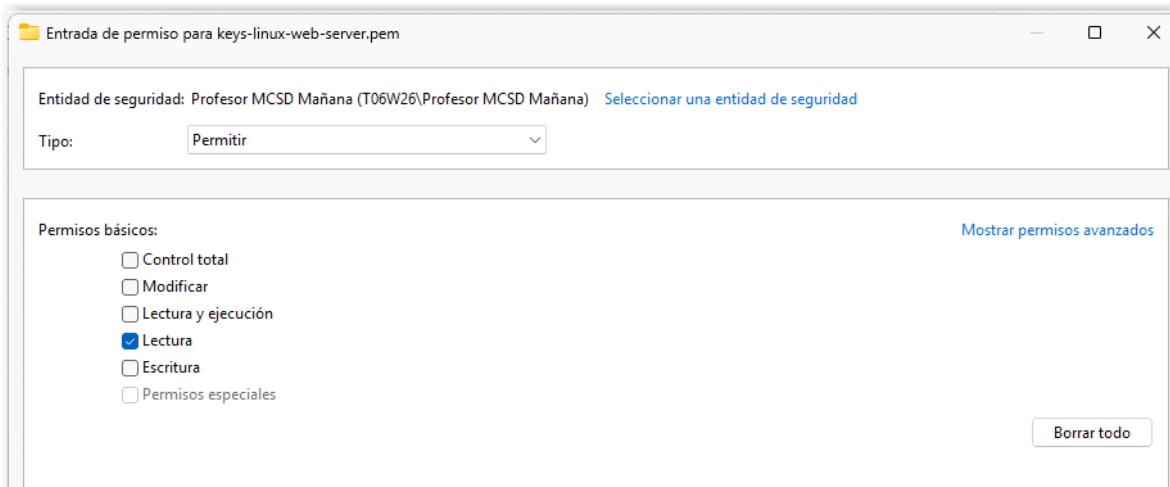
 chmod 400 "keys-linux-web-server.pem"

2) Modificar los permisos de forma "manual" en Windows (Siempre funciona)





Seleccionamos Solo Lectura



Y ya podremos conectarnos

```
c:\KEYS
λ ssh -i "keys-linux-web-server.pem" ec2-user@ec2-44-204-188-22.compute-1.amazonaws.com
      #_
      ~\_ ##### Amazon Linux 2
      ~~ \_#####\
      ~~   \###| AL2 End of Life is 2026-06-30.
      ~~     \#/ __
      ~~       V~' '-'>
      ~~~      / A newer version of Amazon Linux is available!
      ~~_. _/_/
      _/_/ _/_/ Amazon Linux 2023, GA and supported until 2028-03-15.
      _/m/ ' https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-172-31-82-194 ~]$
```

Lo que visualizamos es la IP privada con nuestro usuario.

Estamos en el sistema operativo **CentOS** que es el que utilizan las máquinas de AWS Con AMI.

En esta máquina se utiliza el instalador **yum** en lugar de **apt-get**

Vamos a instalar nuestro Web Server.

sudo su

```
[ec2-user@ip-172-31-82-194 ~]$ yum install httpd -y
```

Iniciamos el servicio de Apache

service httpd start

Indicamos que, al arrancar la máquina EC2, también active el servicio del server

```
[root@ip-172-31-82-194 ec2-user]# service httpd start
Redirecting to /bin/systemctl start httpd.service
[root@ip-172-31-82-194 ec2-user]# chkconfig httpd on
Note: Forwarding request to 'systemctl enable httpd.service'.
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/httpd.service.
```

Visualizar si tenemos el servicio activo

```
[root@ip-172-31-82-194 ec2-user]# service httpd status
Redirecting to /bin/systemctl status httpd.service
● httpd.service - The Apache HTTP Server
  Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
  Active: active (running) since Mon 2025-05-05 09:02:12 UTC; 1min 13s ago
    Docs: man:httpd.service(8)
  Main PID: 3693 (httpd)
    Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec: 0 B/sec"
  CGroup: /system.slice/httpd.service
```

Dentro del servidor Apache tenemos una carpeta dónde estarán nuestros ficheros del Servidor Web: **/var/www/html**

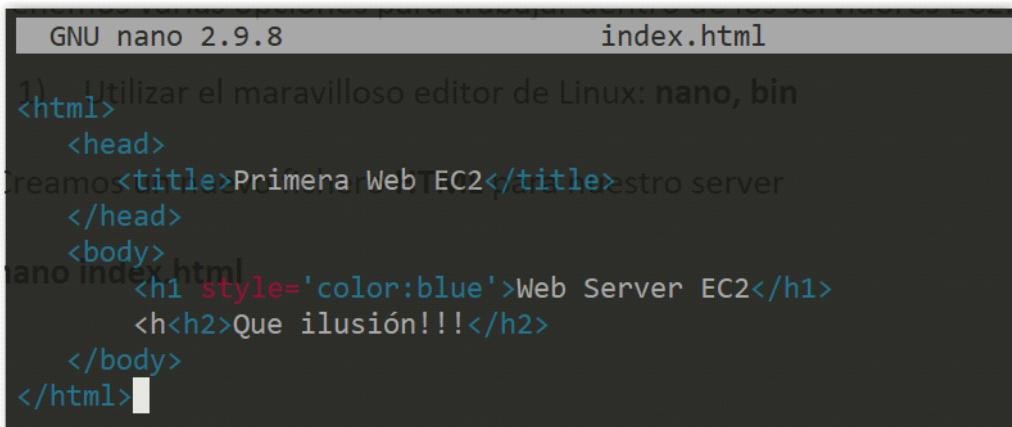
cd /var/www/html

Tenemos varias opciones para trabajar dentro de los servidores EC2.

- 1) Utilizar el maravilloso editor de Linux: **nano**, **vim**

Creamos un nuevo fichero **HTML** para nuestro server

nano index.html

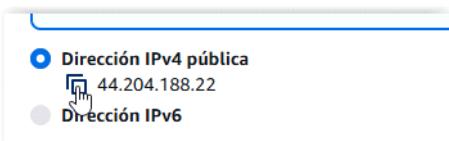


```
GNU nano 2.9.8                               index.html
1) Utilizar el maravilloso editor de Linux: nano, bin
<html>
  <head>
    <title>Primera Web EC2</title>
  </head>
  <body>
    <h1 style='color:blue'>Web Server EC2</h1>
    <h2>Que ilusión!!!</h2>
  </body>
</html>
```

Cuando lo tengamos, pulsamos sobre **CONTROL + X** e indicamos **yes**

Como anteriormente pudimos abrir el puerto **80** de la máquina, se supone que ya Tenemos acceso al servidor.

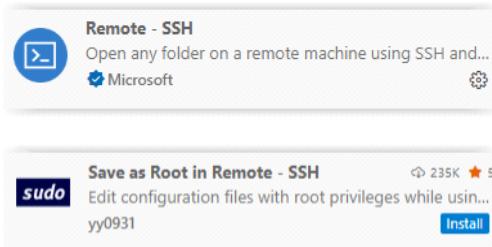
Nos conectamos por URL



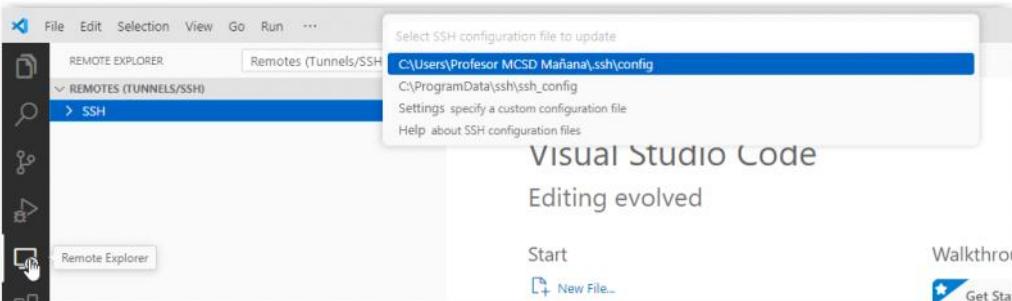
Ya tendremos nuestra máquina activa con la página Web Server



2) Podemos utilizar VS Code para editar ficheros en remoto dentro de nuestras máquinas



Pulsamos en la izquierda, sobre el nuevo ícono de **Remote Explorer** y abrimos el fichero Para la configuración.



Actualmente con la versión CentOS 7 no se puede, así que lo haremos con la versión 8 en Un futuro o con Ubuntu/Debian.

El siguiente paso es eliminar los recursos utilizados.

Las máquinas virtuales solamente consumen cuando están encendidas.

Si tenemos dos máquinas encendidas, las 750 horas corresponden al conjunto de todas las máquinas.

Recursos que tenemos que eliminar:

- 1) Máquina Virtual EC2.
- 2) Grupo de Seguridad
- 3) Las claves las vamos a mantener ya que se almacenan dentro de AWS.

Las máquinas EC2 tienen 3 estados:

- 1) **Encendido:** Consumiendo tiempo
- 2) **Terminado:** Este estado es cuando eliminamos una máquina. Primero se termina y la máquina estará ahí visible. En algún momento, AWS pasa el Recolector de basura y nuestra máquina desaparecerá.
- 3) **Apagado:** Inactiva y no consume tiempo

The screenshot shows the AWS EC2 Instances page. It displays a single instance named "linux-web-server" with ID "i-0547f2d830af9d62d" in the "En ejecución" (Running) state. A context menu is open over this instance, with the option "Terminar (eliminar) instancia" (Terminate (Delete)) highlighted.

Y pasará al estado de **Terminada**

The screenshot shows the AWS EC2 Instances page again. The same instance, "linux-web-server", is now listed in the "Terminada" (Terminated) state. The context menu is no longer visible.

Eliminamos también el grupo de seguridad

The screenshot shows the AWS Network & Security Groups page under the "Red y seguridad" (Networking & Security) section. It lists two security groups: "sg-007e8e1140e7be266" and "sg-0e7056c1ca62860a1".

The screenshot shows the AWS Network & Security Groups page. A context menu is open over the security group "sg-007e8e1140e7be266". The option "Eliminar grupos de seguridad" (Delete security group) is highlighted.

EBS - ELASTIC BLOCK STORE

martes, 6 de mayo de 2025 9:09

En términos generales, esto es simplemente utilizar discos duros para nuestras máquinas virtuales EC2.

Podemos compartir dichos discos duros en distintas máquinas, como si de un USB fuera.

Cuando creamos cualquier máquina EC2, nos crea un volumen raíz asociado a dicha máquina. Dicho volumen es

Llamado **EBS ROOT**

Tenemos dos formas de utilizar el servicio EBS:

- 1) **Root:** Disco duro principal de cada máquina, no se puede compartir y se elimina con la propia máquina
- 2) **Volumen de datos:** Son discos duros (podrían ser principales) y que podemos enchufar o desenchufar
Entre múltiples máquinas (No al mismo tiempo)

Nota: Para trabajar con múltiples volúmenes es necesario estar en la misma zona geográfica y, además, en la Misma zona dentro de la propia zona geográfica.

Tenemos 30 gb para crear discos duros

Vamos a crear una máquina virtual para poder compartir EBS mediante volumen de datos.
Al crear la máquina, los nombres de cada EBS son importantes debido a que para asociarlos o no, es necesario
Saber el volumen del disco.

Creamos una nueva máquina llamada **linux-ec2-ebs**

Nombre y etiquetas Información

Nombre

linux-ec2-ebs Agregar

Recientes **Inicio rápido**

Amazon Linux Del
macOS >
Ubuntu del
Windows del
Red Hat del
SUSE Linux del

Buscar más AMI
Inclusión de AMI de AWS, Marketplace y la comunidad

Imágenes de máquina de Amazon (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type Apto para la capa gratuita

ami-085386e29e4dacc7 (64 bits (x86)) / ami-00bcd7ae558b8179f (64 bits (Arm))
Virtualización: hvm Activado para ENA: true Tipo de dispositivo raíz: ebs

Crear par de claves

Nombre del par de claves
Con los pares de claves es posible conectarse a la instancia de forma segura.

keys-linux-tajamar X

El nombre puede incluir hasta 255 caracteres ASCII. No puede incluir espacios al principio ni al final.

Tipo de par de claves

RSA
Par de claves pública y privada cifradas mediante RSA

ED25519
Par de claves privadas y públicas cifradas ED25519

Formato de archivo de clave privada

.pem
Para usar con OpenSSH

.ppk
Para usar con PuTTY

Cancelar Crear par de claves

específico (vea la instancia).

Crear grupo de seguridad

Seleccionar un grupo de seguridad existente

Nombre del grupo de seguridad - *obligatorio*

grupo-linux-ec2-ebs

Este grupo de seguridad se agregará a todas las interfaces de red. El nombre no se puede editar después de crear el grupo de seguridad. La longitud máxima es de 255 caracteres. Caracteres válidos: a-z, A-Z, 0-9, espacios y _-:/()#@[]+=&.;!\$*

Descripción - *obligatorio* | [Información](#)

launch-wizard-1 created 2025-05-06T07:18:23.549Z

Reglas de grupos de seguridad de entrada

▼ Configurar almacenamiento [Información](#)

Avanzado

1x GiB Volumen raíz, No cifrado

i Los clientes que cumplen los requisitos de la capa gratuita pueden obtener hasta 30 GB de almacenamiento magnético o de uso general (SSD) de EBS X

[Agregar un nuevo volumen](#)

i Haga clic en Actualizar para ver la información de la copia de seguridad C
Las etiquetas que asigne determinan si alguna política de Data Lifecycle Manager realizará una

Una vez que nos ha creado la máquina EC2 podemos visualizar, dentro de Almacenamiento que tenemos una Unidad con el nombre **/dev/xvda**

i-04f76091af12312ef (linux-ec2-ebs)

Detalles Estado y alarmas Monitoreo Seguridad Redes **Almacenamiento** Etiquetas

▼ Detalles del dispositivo raíz

Nombre del dispositivo raíz <input checked="" type="checkbox"/> /dev/xvda	Tipo de dispositivo raíz EBS	Optimización para EBS desactivado
--	---------------------------------	--------------------------------------

▼ Dispositivos de bloques

Como os he comentado, dentro de la propia zona geográfica, nos crea la instancia dentro de una región, en mi caso
La región está creada en **us-east-1d**

Última actualización C Hace 1 minuto

Conectar **Estado de la instancia** Acciones

Nombre o etiqueta (case-sensitive) Todos los estados

1 de 1 Estado de la al: Zona de dispon... DNS de IPv4 pública Dirección IP

Verificar Ver alarmas + us-east-1d ec2-3-91-55-80.compute-1.amazonaws.com 3.91.5!

Vamos a crear un nuevo disco duro en la zona donde tenemos la máquina. (1d)

Entramos en **EBS** y en **volúmenes**

▼ Elastic Block Store

- Volúmenes** C
- Instantáneas
- Administrador del ciclo de vida

Crear volumen Información

Cree un volumen de Amazon EBS para asociarlo a cualquier instancia EC2 en la misma zona de disponibilidad.

Configuración del volumen

Tipo de volumen | [Información](#)

Magnético (estándar)

Tamaño (GiB) | [Información](#)

1

(Mín.: 1 GiB, máx.: 1024 GiB.)

IOPS | [Información](#)

No se aplica

Lo más importante es la ZONA

Zona de disponibilidad | [Información](#)

us-east-1d

ID de instantánea - *opcional* | [Información](#)

No crear un volumen a partir de una instantánea



Cifrado | [Información](#)

Utilice el cifrado de Amazon EBS como una solución de cifrado para los recursos de EBS asociados a las instancias EC2.

Cifrar este volumen

Etiquetas - *opcional* [Información](#)

Las etiquetas son marcas que se asignan a un recurso de AWS. Cada etiqueta consta de una clave y un valor opcional. Puede utilizarlas para los costos en AWS.

Clave

name

Valor - *opcional*

hdd-linux-ec2

[Agregar etiqueta](#)

Puede agregar 49 etiquetas más.

Tenemos un disco duro nuevo

Volúmenes (1/2) [Información](#)

Conjuntos de filtros guardados

[Elegir conjunto de filt...](#)

Buscar

	Name	ID de volumen	Tipo	Tamaño	IOPS
<input type="checkbox"/>	-	vol-01df6f30abef3fe88	gp2	8 GiB	100
<input checked="" type="checkbox"/>	hdd-linux-ec2	vol-0bd0e0516b418df3f	standard	1 GiB	-

La forma de utilizar este disco duro va a ser como volumen.

Debemos realizar un **Attach** dentro de la máquina para asociar el **EBS** nuevo

Volúmenes (1/2) [Información](#)

Conjuntos de filtros guardados

[Elegir conjunto de filt...](#)

Buscar

	Name	ID de volumen	Tipo	Tamaño
<input type="checkbox"/>	-	vol-01df6f30abef3fe88	gp2	8 GiB
<input checked="" type="checkbox"/>	hdd-linux-ec2	vol-0bd0e0516b418df3f	standard	1 GiB

ID de volumen: vol-0bd0e0516b418df3f (hdd-linux-ec2)

Last updated
less than a minute ago

[Acciones](#)

[Modificar volumen](#)

[Crear instantánea](#)

[Crear política de ciclo de vida de instantáneas](#)

[Eliminar volumen](#)

[Asociar volumen](#)

[Desasociar el volumen](#)

[Desasociar el volumen forzadamente](#)

Asociar volumen Información

Asocie un volumen a una instancia para utilizarlo de la misma forma en la que usaría una unidad de disco duro.

Detalles básicos

ID de volumen

vol-0bd0e0516b418df3f (hdd-linux-ec2)

Zona de disponibilidad

us-east-1d

Instancia | [Información](#)

Buscar ID de instancia o etiqueta de nombre



i-04f76091af12312ef
(linux-ec2-ebs) (running)



Detalles básicos

ID de volumen

vol-0bd0e0516b418df3f (hdd-linux-ec2)

Zona de disponibilidad

us-east-1d

Instancia | [Información](#)

i-04f76091af12312ef
(linux-ec2-ebs) (running)



Solo se muestran las instancias de la misma zona de disponibilidad que el volumen seleccionado.

Nombre del dispositivo | [Información](#)

/dev/sdf



Nombres de dispositivos recomendados para Linux: /dev/xvda para el volumen raíz. /dev/sd[f-p] para los volúmenes de datos.

Nuestra máquina tiene actualmente dos discos duros.

Como todo SO, uno de ellos está asociado, pero no es reconocible actualmente.
Debemos hacer que el disco EBS sea reconocible.

Entramos en nuestra máquina Linux y nos conectamos como root

sudo su

Visualizamos los discos duros **fdisk -l**

```
[ec2-user@ip-172-31-27-113 ~]$ sudo su
[ec2-user@ip-172-31-27-113 ~]# fdisk -l
Disk /dev/xvda: 8 GiB, 8589934592 bytes, 16777216 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes actualmente
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: B6468319-FCE5-46CE-ADBA-22AE3735410D
os en nuestra máquina Linux y nos conectamos como root
Device      Start    End  Sectors Size Type
/dev/xvda1     4096 16777182 16773087   8G Linux filesystem
/dev/xvda128   2048     4095      2048   1M BIOS boot
```

```
Disk /dev/xvdf: 1 GiB, 1073741824 bytes, 2097152 sectors
Units: sectors of 1 * 512 = 512 bytes
```

Vamos a particionar para que Linux reconozca nuestro disco.

fdisk /dev/xvdf

Escribimos **n**, después la partición **Primary p** y después pulsamos Enter a todo

Formateamos la nueva partición que acabamos de crear

mkfs -t ext4 /dev/xvdf

Vamos a montar nuestra nueva unidad de disco duro dentro **/mnt**

```
cd /mnt
```

Creamos un nueva carpeta llamada **VOL1**

```
mkdir VOL1
```

Montamos nuestro disco duro dentro de **VOL1**

```
mount /dev/xvdf /mnt/VOL1
```

```
[root@ip-172-31-27-113 ec2-user]# cd /mnt  
[root@ip-172-31-27-113 mnt]# mount /dev/xvdf /mnt/VOL1  
mount: /mnt/VOL1: mount point does not exist.  
[root@ip-172-31-27-113 mnt]# mkdir VOL1  
[root@ip-172-31-27-113 mnt]# mount /dev/xvdf /mnt/VOL1  
[root@ip-172-31-27-113 mnt]#
```

```
[root@ip-172-31-27-113 mnt]# df -h  
Filesystem      Size  Used Avail Use% Mounted on  
devtmpfs        468M   0  468M   0% /dev  
tmpfs          477M   0  477M   0% /dev/shm  
tmpfs          477M  468K  476M   1% /run  
tmpfs          477M   0  477M   0% /sys/fs/cgroup  
/dev/xvda1       8.0G  1.9G  6.2G  24% /  
tmpfs          96M   0   96M   0% /run/user/1000  
tmpfs          96M   0   96M   0% /run/user/0  
/dev/xvdf       974M  24K  907M   1% /mnt/VOL1  
[root@ip-172-31-27-113 mnt]#
```

Lo que hemos realizado "simplemente!" es dar formato a nuestro HDD y montar un volumen dentro Del SO de Linux para acceder a sus ficheros.

Vamos a crear un nuevo fichero para comprobar que todo es funcional.

```
cd VOL1
```

```
[root@ip-172-31-27-113 VOL1]# echo "Soy una maquina en Linux" > file1.txt  
[root@ip-172-31-27-113 VOL1]# ls  
file1.txt  lost+found
```

Lo que hemos realizado es algo temporal, cuando apaguemos la máquina e iniciemos, no estará nuestro /mnt/VOL1

Debemos montarlo de forma estática.

Para ello necesitamos el UUID del disco duro.

Desmontamos el disco duro

```
umount -l VOL1
```

Mediante **blkid** podemos visualizar el UUID de nuestro disco duro **/dev/xvdf**

Incluimos dentro de **fstab** el UUID montando dentro de la unidad **/mnt/VOL1**

```
nano /etc/fstab
```

```
GNU nano 2.9.8                               /etc/fstab                         Modified  
#  
UUID=912572f0-5d41-4612-a76d-a385f9e49fb5  /dev/xvdf      xfs    defaults,noatime  1   1  
UUID=90eeef492-03dd-4c12-aba1-4fc676c22d3c  /mnt/VOL1     ext4   defaults          0   0
```

Con esto ya hemos montado el disco duro para siempre.
Para comprobarlo, reiniciamos nuestra máquina.

Resumen de instancia de i-04f76091af12312ef (linux-ec2-ebs) [Información](#)

[Conectar](#) [Estado de la instancia ▲](#) [Acciones ▼](#)

Se ha actualizado hace 1 hora

ID de la instancia [i-04f76091af12312ef](#)

Dirección IPv4 -

Tipo de nombre de anfitrión -

Nombre DNS de IP privada (solo IPv4) -

Detener instancia

Iniciar instancia

Reiniciar instancia

Hibernar instancia

Terminar (eliminar) instancia

Última ejecución

Con el comando `df -h` podremos visualizar si nos ha creado de nuevo el VOL1

```
[root@ip-172-31-27-113 mnt]# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        468M   0    468M  0% /dev
tmpfs          477M   0    477M  0% /dev/shm
tmpfs          477M  412K  476M  1% /run
tmpfs          477M   0    477M  0% /sys/fs/cgroup
/dev/xvda1       8.0G  1.9G  6.2G  24% /
/dev/xvdf      974M  28K  907M  1% /mnt/VOL1
tmpfs          96M   0    96M  0% /run/user/1000
[root@ip-172-31-27-113 mnt]#
```

SNAPSHOT EC2

Una instantánea es una réplica de una imagen AMI EC2.

La instantánea también almacenará la información de los discos duros conectados.

Esta funcionalidad también nos permitiría incluir la máquina en el Market de AWS.

Vamos a crear un Snapshot de nuestra máquina y, posteriormente, eliminamos la máquina e intentamos Recuperarla.

▼ Elastic Block Store

- Volúmenes
- Instantáneas**
- Administrador del ciclo de vida

Crear instantánea [Información](#)

Cree una instantánea de un momento dado de un volumen de EBS y utilícela como referencia para nuevos volúmenes o para copia de seguridad de datos. Puede crear instantáneas aisladas individual o puede crear instantáneas de varios volúmenes a partir de todos los volúmenes asociados a una instancia.

Origen

Tipo de recurso | [Información](#)

Volumen

Cree una instantánea a partir de un volumen específico.

Instancia

Cree instantáneas de varios volúmenes a partir de una instancia.

ID de la instancia

La instancia desde la que se crean instantáneas de varios volúmenes.

i-04f76091af12312ef (linux-ec2-ebs)
us-east-1d



Volumenes - opcional Información

De forma predeterminada, todos los volúmenes asociados a la instancia se incluyen en el conjunto de instantáneas de datos específicos. También puede indicar si desea copiar las etiquetas de los volúmenes de origen en las instantáneas.

Excluir volúmenes

Indique si desea excluir el volumen raíz o volúmenes de datos específicos del conjunto de instantáneas.

Excluir el volumen raíz ( vol-01df6f30abef3fe88, No cifrado)

Excluir volúmenes de datos específicos

Copiar etiquetas del volumen de origen

Indique si desea copiar las etiquetas del volumen de origen en la instantánea.

Copiar etiquetas

Etiquetas Información

Las etiquetas son marcas que se asignan a un recurso de AWS. Cada etiqueta consta de una clave y un valor opcional. Puede utilizarlas para buscar los recursos y filtrarlos, o para los costos en AWS.

Clave

Name 

Valor - opcional

snapshot-ebs-ec2 

 Eliminar

 Agregar etiqueta

Puede agregar 49 etiquetas más.

Visualizaremos que nos ha creado dos elementos, uno por cada HDD que tengamos

Instantáneas (2) <small>Información</small>						Last updated  less than a minute ago	 Papelera de reciclaje	Acciones ▾	 Crear
De mi propiedad ▾		Buscar							
<input type="checkbox"/>	Name	ID de instantánea	Tamaño completo...	Tamaño de...	Descripción				
<input type="checkbox"/>	snapshot-ebs-...	snap-03b274a7bfc905b10	-	8 GiB	-				
<input type="checkbox"/>	snapshot-ebs-...	snap-0ccf827ecc58d5b2d	-	1 GiB	-				

El disco que nos interesa es **root (8gb)** que es quién tiene asociado el otro disco de HDD de 1gb.

Vamos a recuperar el disco raíz y, a su vez, se recuperará también el otro disco EBS

Terminamos nuestra máquina de Linux.

Instancias (1/1) Información Última actualización Hace about 1 hour  Conectar Estado de la instancia ▾ Acciones ▾

Buscar Instancia por atributo o etiqueta (case-sensitive)

<input checked="" type="checkbox"/>	Name 	ID de la instancia	Estado de la i...
<input checked="" type="checkbox"/>	linux-ec2-ebs	i-04f76091af12312ef	 En ejecución 

Detener instancia
Iniciar instancia
Reiniciar instancia
Hibernar instancia
Terminar (eliminar) instancia

▼ Eliminar volúmenes de EBS

Los volúmenes de EBS que no tengan la opción "Eliminar al terminar" establecida en "verdadero" se conservarán cuando se termine esta instancia. Estos volúmenes pueden generar costos de EBS por [Precios de Amazon EC2](#)

Para eliminar volúmenes asociados a esta instancia, vaya a la [Pantalla de volúmenes](#).

Los siguientes volúmenes no están configurados para eliminarse cuando se termine la instancia: vol-0bd0e0516b418df3f

Para confirmar que desea terminar las instancias, elija el botón de terminar que aparece a continuación. Las instancias que tengan la protección de terminación activada no se terminarán. La terminación de la instancia no se puede deshacer.

 Cancelar

 Terminar (eliminar)

El siguiente paso es crear otra máquina distinta con el SNAPSHOT del disco Raíz.

Instantáneas creadas correctamente: snap-0ccf827ecc58d5b2d, snap-03b274a7bfc905b10.

Instantáneas (1/2) Información		
Last updated less than a minute ago		
De mi propiedad	<input type="text"/> Buscar	Papelera de reciclaje
Name	ID de instantánea	Tamaño completo...
<input checked="" type="checkbox"/> snapshot-ebs-ec2	snap-03b274a7bfc905b10	1.91 GiB
<input type="checkbox"/> snapshot-ebs-ec2	snap-0ccf827ecc58d5b2d	52 MiB

ID de instantánea: snap-03b274a7bfc905b10 ([snapshot-ebs-ec2](#))

[Detalles](#) | [Configuración de instantáneas](#) | [Cana de almacenamiento](#) | [Etiquetas](#)

[Crear volumen a partir de una instantánea](#)
[Crear imagen a partir de una instantánea](#) (seleccionado)
[Copiar instantánea](#)
[Lanzar calculadora de tiempo de copias](#)
[Eliminar instantánea](#)
[Administrar etiquetas](#)
[Configuración de instantáneas](#)
[Archivar](#)

Crear imagen a partir de una instantánea [Información](#)

Cree una nueva imagen a partir de una instantánea tomada del volumen de dispositivo raíz de una instancia.

Configuración de imagen

ID de instantánea

snap-03b274a7bfc905b10 ([snapshot-ebs-ec2](#))

Nombre de la imagen

Un nombre descriptivo para la imagen.

ami-linux-ec2-ebs

De 3 a 128 caracteres. Los caracteres válidos son a-z, A-Z, 0-9, espacios y - _ / . @.

Descripción

Una descripción de la imagen.

Imagen con dos HDD

255 caracteres como máximo

Lo que hemos realizado es crear un nuevo AMI en nuestro servidor AWS.
Dicho AMI podríamos ponerlo en el Market si lo deseamos.

Todos los productos (1 filtrados, 1 sin filtrar)



ami-linux-ec2-ebs

ami-00f7f6aba2bad8498

Imagen con dos HDD

OwnerAlias: --Plataforma: --Arquitectura: x86_64 Propietario: 772056227005
Fecha de publicación: 2025-05-06 Tipo de dispositivo raíz: ebs
Virtualización: hvm Activado para ENA: Sí

[Seleccionar](#)

Resumen de imagen de ami-00f7f6aba2bad8498

[EC2 Image Builder](#)

[Acciones](#)

[Lanzar instancia a partir de una AMI](#)

ID de AMI

ami-00f7f6aba2bad8498

Tipo de imagen

machine

Detalles de la plataforma

Linux/UNIX

T

E

Nombre de AMI

ami-linux-ec2-ebs

ID de cuenta del propietario

772056227005

Arquitectura

x86_64

O

R

Nombre del dispositivo raíz

/dev/sda1

Estado

Disponible

Origen

772056227005/ami-linux-ec2-ebs

T

h

▼ Configuraciones de red [Información](#)

VPC : **obligatorio** | [Información](#)

vpc-066662b1b43cc56c0
172.31.0.0/16

(predeterminado) ▾



Subred | [Información](#)

subnet-003e77056d7264783

VPC: vpc-066662b1b43cc56c0 Propietario: 772056227005
Zona de disponibilidad: us-east-1d Tipo de zona: Zona de disponibilidad
Direcciones IP disponibles: 4091 CIDR: 172.31.16.0/20

Crear nueva subred ▾

Asignar automáticamente la IP pública | [Información](#)

Habilitar



Se aplican cargos adicionales cuando no se cumplen los límites del [nivel gratuito](#)

Grupos de seguridad comunes | [Información](#)

Seleccionar grupos de seguridad ▾

grupo-linux-ec2-ebs sg-0d8f570f17e103817 X

VPC: vpc-066662b1b43cc56c0

Compare reglas de grupo de seguridad

Los grupos de seguridad que agrega o elimina aquí se agregarán a todas las interfaces de red o se eliminarán de ellas.

► Configuración de red avanzada

Volumenes (1/2) [Información](#)

Conjuntos de filtros guardados

Elegir conjunto de filt... ▾

Buscar

Name	ID de volumen	Tipo	Tamaño
<input checked="" type="checkbox"/> hdd-linux-ec2	vol-0bd0e0516b418df3f	standard	1 GiB
<input type="checkbox"/> -	vol-01a08db46cf52ed0	gp3	8 GiB

ID de volumen: vol-0bd0e0516b418df3f (hdd-linux-ec2)

[Detalles](#)

[Comprobaciones de estado](#)

[Monitoreo](#)

[Etiquetas](#)

Last updated less than a minute ago



Acciones ▾

Cre...

Modificar volumen

Crear instantánea

Crear política de ciclo de vida de instantáneas

Eliminar volumen

Asociar volumen

Desasociar el volumen

Desasociar el volumen forzadamente

Administrar habilitación automática de E/S

Administrar etiquetas

Inyección de errores

Asociar volumen [Información](#)

Asocie un volumen a una instancia para utilizarlo de la misma forma en la que usaría una unidad de disco duro física no

Detalles básicos

ID de volumen

vol-0bd0e0516b418df3f (hdd-linux-ec2)

Zona de disponibilidad

us-east-1d

Instancia | [Información](#)

i-057ef55109bf4cffc
(linux-ec2-ebs-new) (running)



Solo se muestran las instancias de la misma zona de disponibilidad que el volumen seleccionado.

Nombre del dispositivo | [Información](#)

/dev/sdf

Lo probamos en un rato si es funcional o no...

VOLUMEN DE HDD A PARTIR DE SNAPSHOT

El siguiente paso es crear un EBS a partir de los Snapshot que tenemos.

Creamos un nuevo HDD de 1gb a partir del Snapshot, se supone que contendrá el fichero **file1.txt**

Instantáneas (1/2) [Información](#)

Last updated less than a minute ago [C](#) [Papelera de reciclaje](#) [Acciones ▾](#) [Crear](#)

Name	ID de instantánea	Tamaño complet...
<input type="checkbox"/> snapshot-ebs-ec2	snap-03b274a7bfc905b10	1.91 GiB
<input checked="" type="checkbox"/> snapshot-ebs-ec2	snap-0ccf827ecc58d5b2d	52 MiB

ID de instantánea: snap-0ccf827ecc58d5b2d (snapshot-ebs-ec2)

[Crear volumen a partir de una instantánea](#)

[Crear imagen a partir de una instantánea](#)

[Copiar instantánea](#)

[Lanzar calculadora de tiempo de copias](#)

[Eliminar instantánea](#)

[Administrar etiquetas](#)

[Configuración de instantáneas](#)

[Archivar](#)

Configuración del volumen

ID de instantánea
 snap-0ccf827ecc58d5b2d (snapshot-ebs-ec2)

Tipo de volumen | [Información](#)

Tamaño (GiB) | [Información](#)

 (Mín.: 1 GiB, máx.: 1024 GiB.)

IOPS | [Información](#)
 No se aplica

Rendimiento (MiB/s) | [Información](#)
 No se aplica

Zona de disponibilidad | [Información](#)

Etiquetas - opcional [Información](#)

Las etiquetas son marcas que se asignan a un recurso de AWS. Cada etiqueta consta de una clave y un valor opcional. Puede utilizarlas para buscar los recursos y filtrarlos, o para hacer de los costos en AWS.

Clave	Valor - opcional
<input type="text" value="Name"/>	<input type="text" value="hdd-ebs-1a"/> Eliminar

[Agregar etiqueta](#)
 Puede agregar 49 etiquetas más.

También creamos otro disco duro a partir del disco de 8gb que es quién contendrá la información del **FSTAB**

Instantáneas (1/2) [Información](#)

Last updated 1 minute ago [C](#) [Papelera de reciclaje](#) [Acciones ▾](#) [Crear](#)

Name	ID de instantánea	Tamaño complet...
<input checked="" type="checkbox"/> snapshot-ebs-ec2	snap-03b274a7bfc905b10	1.91 GiB
<input type="checkbox"/> snapshot-ebs-ec2	snap-0ccf827ecc58d5b2d	52 MiB

ID de instantánea: snap-03b274a7bfc905b10 (snapshot-ebs-ec2)

[Crear volumen a partir de una instantánea](#)

[Crear imagen a partir de una instantánea](#)

[Copiar instantánea](#)

[Lanzar calculadora de tiempo de copias](#)

[Eliminar instantánea](#)

[Administrar etiquetas](#)

[Configuración de instantáneas](#)

[Archivar](#)

Configuración del volumen

ID de instantánea

snap-03b274a7bfc905b10 (snapshot-ebs-ec2)

Tipo de volumen | [Información](#)

SSD de uso general (gp3)



Tamaño (GiB) | [Información](#)

8

(Mín.: 1 GiB, máx.: 16384 GiB.

IOPS | [Información](#)

3000

Mín.: 3000 IOPS, máx.: 16000 IOPS.

Rendimiento (MiB/s) | [Información](#)

125

Mín.: 125 MiB, máx.: 1000 MiB. Línea de base: 125 MiB/s.

Zona de disponibilidad | [Información](#)

us-east-1a



Restauración rápida de instantáneas | [Información](#)

No habilitado para la instantánea seleccionada

Cifrado

Utilice el cifrado de Amazon EBS como una solución de cifrado para los recursos de EBS asociados a las instancias EC2.

Cifrar este volumen

Etiquetas - *opcional* [Información](#)

Las etiquetas son marcas que se asignan a un recurso de AWS. Cada etiqueta consta de una clave y un valor opcional. Puede utilizarlas para buscar los recursos y filtrarlos, o pa de los costos en AWS.

Clave

Name

Valor - *opcional*

ssd-ebs-1a



[Eliminar](#)

[Agregar etiqueta](#)

Puede agregar 49 etiquetas más.

Creamos una nueva máquina llamada **ec2-ebs-eliminar**

Importante: La máquina debe estar dentro de la zona **us-east-1a**

Lanzar una instancia [Información](#)

Amazon EC2 le permite crear máquinas virtuales, o instancias, que se ejecutan en la nube de AWS. Comience rápidamente siguiendo los sencillos pasos que se indican a continuación.

Nombre y etiquetas [Información](#)

Nombre

ec2-ebs-eliminar

[Agregar etiquetas adicionales](#)



Recientes Mis AMI Inicio rápido

Amazon Linux macOS Ubuntu Windows Red Hat SUSE Linux Del > del

Buscar más AMI
Inclusión de AMI de AWS, Marketplace y la comunidad

Imágenes de máquina de Amazon (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type
ami-085386e29e44dacd7 (64 bits (x86)) / ami-00bdc7ae558b8179f (64 bits (Arm))
Virtualización: hvm Activado para ENA: true Tipo de dispositivo raíz: ebs

Apto para la capa gratuita ▾

▼ Configuraciones de red Información

VPC : **obligatorio** | Información

vpc-066662b1b43cc56c0 (predeterminado) ▾ C

Subred | Información

subnet-000d1117215ce99b3
VPC: vpc-066662b1b43cc56c0 Propietario: 772056227005
Zona de disponibilidad: us-east-1a Tipo de zona: Zona de disponibilidad
Direcciones IP disponibles: 4091 CIDR: 172.31.32.0/20

C Crear nueva subred ▾

Asignar automáticamente la IP pública | Información

Habilitar ▾

Se aplican cargos adicionales cuando no se cumplen los límites del nivel gratuito

específico llegue a la instancia.

Crear grupo de seguridad Seleccionar un grupo de seguridad existente

Grupos de seguridad comunes | Información

Seleccionar grupos de seguridad ▾

grupo-linux-ec2-ebs sg-0d8f570f17e103817 X
VPC: vpc-066662b1b43cc56c0

C Compare reglas de grupo de seguridad

Los grupos de seguridad que agrega o elimina aquí se agregarán a todas las interfaces de red o se eliminarán de ellas.

▶ Configuración de red avanzada

En esta nueva máquina, tenemos un disco duro raíz que está limpio.

Nosotros queremos el disco duro de 8gb que contiene la información de FSTAB

Detenemos la máquina de eliminar

Instancias (1/3) Información Última actualización Hace 2 minutes C Conectar Estado de la instancia ▾ Acciones ▾

Buscar Instancia por atributo o etiqueta (case-sensitive)

Name	ID de la instancia	Estado de la i...
linux-ec2-ebs	i-04f76091af12312ef	Terminada
linux-ec2-ebs-...	i-057ef55109bf4cffc	En ejecución
ec2-ebs-eliminar	i-0e5cdc7a757066fe6	En ejecución

Detener instancia Iniciar instancia Reiniciar instancia Hibernar instancia Terminar (eliminar) instancia

t2.micro 2/2 comprobador

Quitamos el disco duro principal que nos ha creado la máquina ec2-eliminar

Screenshot of the AWS CloudFormation console showing the stack **i-0e5cdc7a757066fe6** (ec2-ebs-eliminar). The stack status is **Detenida** (Stopped) with **t2.micro** instance type and **2/2 comprobaci** (2/2 checks) status.

Volumenes (1/5) Información

Name	ID de volumen	Tipo	Tamaño
ssd-ebs-1a	vol-0002c85ea85ebdbde	gp3	8 GiB
-	vol-0e466c553a5354443	gp2	8 GiB
hdd-ebs-1a	vol-013021a184ffe512c	standard	1 GiB

ID de volumen: vol-0e466c553a5354443

Acciones

- Modificar volumen
- Crear instantánea
- Crear política de ciclo de vida de instantáneas
- Eliminar volumen
- Asociar volumen
- Desasociar el volumen
- Desasociar el volumen forzadamente
- Administrar habilitación automática de E/S
- Administrar etiquetas
- Inyección de errores

Asociamos el volumen de 8gb del ssd con FSTAB

El volumen se ha desasociado correctamente.

Volumenes (1/5) Información

Name	ID de volumen	Tipo	Tamaño
ssd-ebs-1a	vol-0002c85ea85ebdbde	gp3	8 GiB
-	vol-0e466c553a5354443	gp2	8 GiB

ID de volumen: vol-0002c85ea85ebdbde (ssd-ebs-1a)

Acciones

- Modificar volumen
- Crear instantánea
- Crear política de ciclo de vida de instantáneas
- Eliminar volumen
- Asociar volumen
- Desasociar el volumen
- Desasociar el volumen forzadamente
- Administrar habilitación automática de E/S
- Administrar etiquetas
- Inyección de errores

Detalles básicos

ID de volumen

vol-0002c85ea85ebdbde (ssd-ebs-1a)

Zona de disponibilidad

us-east-1a

Instancia | Información

i-0e5cdc7a757066fe6
(ec2-ebs-eliminar) (stopped)



Solo se muestran las instancias de la misma zona de disponibilidad que el volumen seleccionado.

Nombre del dispositivo | Información

/dev/xvda

Nombres de dispositivos recomendados para Linux: /dev/xvda para el volumen raíz. /dev/sd[f-p] para los volúmenes de datos.

Nota: Los kernels de Linux más recientes pueden cambiar el nombre de sus dispositivos a /dev/xvdf through /dev/xvdp internamente, aunque el nombre introducido aquí (y mostrado en los detalles) sea /dev/sdf through /dev/sdp.

Asociamos también el HDD de 1gb que hemos creado a partir del Snapshot

Volumenes (1/5) [Información](#)

Last updated less than a minute ago [C](#) Acciones ▲ [Crear volumen](#)

Conjuntos de filtros guardados [Elegir conjunto de filt...](#) Buscar

-	Name	ID de volumen	Tipo	Tamaño
<input type="checkbox"/>	-	vol-0e466c553a5354443	gp2	8 GiB
<input checked="" type="checkbox"/>	hdd-ebs-1a	vol-013021a184ffe512c	standard	1 GiB
<input type="checkbox"/>	hdd-linux-ec2	vol-0bd0e0516b418df3f	standard	1 GiB

ID de volumen: vol-013021a184ffe512c (hdd-ebs-1a)

[Detalles](#) [Comprobaciones de estado](#) [Monitoreo](#) [Etiquetas](#)

[ID de volumen](#) [Tamaño](#) [Tipo](#) [Comprobación de estado](#)

Modificar volumen
Crear instantánea
Crear política de ciclo de vida de instantáneas
Eliminar volumen
Asociar volumen [Desasociar el volumen](#)
Desasociar el volumen forzadamente
Administrar habilitación automática de E/S
Administrar etiquetas
Inyección de errores

Asociar volumen [Información](#)

Asocie un volumen a una instancia para utilizarlo de la misma forma en la que usaría una unidad de disco duro física no

Detalles básicos

ID de volumen
 vol-013021a184ffe512c (hdd-ebs-1a)

Zona de disponibilidad
us-east-1a

Instancia | [Información](#)
i-0e5cdc7a757066fe6
(ec2-ebs-eliminar) (stopped) [C](#)

Solo se muestran las instancias de la misma zona de disponibilidad que el volumen seleccionado.

Nombre del dispositivo | [Información](#)
/dev/sdf

Nombres de dispositivos recomendados para Linux: /dev/xvda para el volumen raíz. /dev/sd[f-p] para los volúmenes de datos.

Vamos a probar si funciona ec2-eliminar

Nos conectamos a la máquina y ponemos sudo su

```
[ec2-user@ip-172-31-32-23 ~]$ sudo su
[root@ip-172-31-32-23 ec2-user]# cd /mnt
[root@ip-172-31-32-23 mnt]# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        468M   0    468M  0% /dev
tmpfs          477M   0    477M  0% /dev/shm
tmpfs          477M  412K  476M  1% /run
tmpfs          477M   0    477M  0% /sys/fs/cgroup
/dev/xvda1       8.0G  1.9G  6.2G  24% /
/dev/xvdf      974M  28K  907M  1% /mnt/VOL1
tmpfs          96M   0    96M  0% /run/user/1000
[root@ip-172-31-32-23 mnt]#
```

Y podremos comprobar que todo es funcional

```
[root@ip-172-31-32-23 VOL1]# ls
file1.txt  lost+found
[root@ip-172-31-32-23 VOL1]#
```

Probamos la siguiente funcionalidad que es la de EC2 del SNAPSHOT

<input checked="" type="checkbox"/> linux-ec2-ebs-new	i-057ef55109bf4cffc	En ejecución Q Q	t2.micro	2/2 comprobacion Ver alarmas +
---	---------------------	--	----------	--

Y podremos comprobar cómo funciona TODO!!!!

```
[ec2-user@ip-172-31-28-231 ~]$ sudo su
[root@ip-172-31-28-231 ec2-user]# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        468M    0  468M   0% /dev
tmpfs          477M    0  477M   0% /dev/shm
tmpfs          477M  412K  476M   1% /run
tmpfs          477M    0  477M   0% /sys/fs/cgroup
/dev/xvda1       8.0G  1.9G  6.2G  24% /
/dev/xvdf       974M  28K  907M   1% /mnt/VOL1
tmpfs          96M    0   96M   0% /run/user/1000
[root@ip-172-31-28-231 ec2-user]# cd /mnt/VOL1
[root@ip-172-31-28-231 VOL1]# ls
file1.txt  lost+found
[root@ip-172-31-28-231 VOL1]#
```

Por último, eliminamos todo.

- Máquinas virtuales: **ec2-eliminar, ec2-ebs**

<input checked="" type="checkbox"/> linux-ec2-ebs-new	i-057ef55109bf4cffc	En ejecución t2.micro	2/2 comprobacion Ver
<input checked="" type="checkbox"/> ec2-ebs-eliminar	i-0e5cdc7a757066fe6	En ejecución t2.micro	2/2 comprobacion Ver

- EBS volúmenes: Todos

Conjuntos de filtros guardados								
Elegir conjunto de filtros...								
Buscar								
	Name	ID de volumen	Tipo	Tamaño	IOPS	Rendimiento	ID de insta...	Creada
<input type="checkbox"/>	ssd-ebs-1a	vol-0002c85ea85ebdbde	gp3	8 GiB	3000	125	snap-03b274a...	2025/05/
<input type="checkbox"/>	-	vol-0e466c553a5354443	gp2	8 GiB	100	-	snap-09f4335...	2025/05/
<input type="checkbox"/>	hdd-ebs-1a	vol-013021a184ffe512c	standard	1 GiB	-	-	snap-0ccf827e...	2025/05/
<input type="checkbox"/>	hdd-linux-ec2	vol-0bd0e0516b418df3f	standard	1 GiB	-	-	-	2025/05/

- AMI: Nuestra AMI debemos desactivarla.

Imágenes de Amazon Machine Image (AMI) (1/1) Información

Papelera de reciclaje EC2 Image Builder Acciones ▾ Lanzar instancia a partir de una AMI

De mi propiedad ▼ Buscar AMI por atributo o etiqueta

Name	ID de AMI	Origen
<input checked="" type="checkbox"/> ami-linux-ec2-ebs	ami-00f7f6aba2bad8498	772056227005/ami-linux-ec2-ebs

Entramos en AMI desactivadas

Imágenes de Amazon Machine Image (AMI) (1/1) Información

Papelera de reciclaje EC2 Image Builder Acciones ▾ Lanzar instancia a partir de una AMI

Imágenes desactivadas ▼ Buscar AMI por atributo o et...

Name	Nombre de AMI
<input checked="" type="checkbox"/> ami-linux-ec2-ebs	ami-00f7f6aba2bad8498

ID de la AMI: ami-00f7f6aba2bad8498

Copiar AMI
Editar permisos de AMI
Solicitar instancias de spot
Administrar etiquetas
Anular registro de la AMI (se ha hecho)
Administrar la protección Desactivar la anulación del registro de la AMI
Cambiar descripción
Configurar lanzamiento rápido

- SnapShot: Todos

Instantáneas (2) [Información](#)

Last updated 1 minute ago [Reciclar](#) [Papelera de reciclaje](#) [Acción](#)

De mi propiedad ▾ Buscar

<input type="checkbox"/>	Name	ID de instantánea	Tamaño completo...	Tamaño de...	Descripción
<input type="checkbox"/>	snapshot-ebs-ec2	snap-03b274a7bfc905b10	1.91 GiB	8 GiB	-
<input type="checkbox"/>	snapshot-ebs-ec2	snap-0ccf827ecc58d5b2d	52 MiB	1 GiB	-

Por último, Security Groups

<input checked="" type="checkbox"/>	-	sg-0d8f570f17e103817	grupo-linux-ec2-ebs	vpc-066662b1b43cc56c0 [2]
-------------------------------------	---	----------------------	---------------------	---------------------------

SECURITY GROUPS

martes, 6 de mayo de 2025 13:19

Un grupo de seguridad indica la red dónde está alojada nuestra máquina o servicio y también podría actuar como Firewall.

Es una herramienta que nos permite abrir los puertos y realizar comunicaciones, por ejemplo, si tenemos un MariaDb dentro de un Linux, podemos abrir el puerto 3306 para que se conecten con nuestra base de datos.

Un grupo de seguridad puede ser compartido por múltiples máquinas/servicios.

Esta funcionalidad es importante porque AWS es un lio entre servicios, comunicaciones y permisos.

Lo primero son los permisos para trabajar con algún servicio: **IAM**

Lo segundo con las comunicaciones entre los diferentes servicios: **Security Groups**

Pongamos que tenemos el siguiente supuesto:

- **EC2-WEB-SERVER:** Servidor Web dedicado
- **EC2-BBDD:** Servidor de base de datos dedicado

Necesitamos poder comunicar las dos máquinas.

Si ponemos las dos máquinas dentro del mismo grupo de seguridad, compartirán la comunicación y la

Información entre sí.

Podríamos perfectamente, quitar el acceso público al servidor de BBDD y que se comuniquen mediante

Su IP privada, es decir, externamente estaría aislado el servidor de BBDD

Cualquier grupo de seguridad tiene dos reglas:

- **Entrada Inbound:** Tráfico entrante a nuestra máquina EC2
- **Salida Outbound:** Tráfico saliente de la máquina, es decir, quién se puede comunicar con la máquina

Para nuestro laboratorio vamos a montar tres máquinas con la siguiente arquitectura:

- **EC2-WEB-SERVER:** Servidor Web Aplicaciones
- **EC2-BBDD:** Servidor de base de datos
- **EC2-AISLADO:** Esta máquina es para las copias de seguridad y necesitamos que esté en un espacio
Aislado, que no tenga tráfico entrante, pero que las dos otras máquinas SI que puedan comunicar con ella.

Para empezar, cada máquina la vamos a montar en una región distinta: **us-east-1a, us-east-1c...**

Cada máquina tendrá un grupo de seguridad distinto.

- **EC2-WEB-SERVER:** us-east-1a, grupo-web-server
- **EC2-BBDD:** us-east-1b, grupo-bbdd
- **EC2-AISLADO:** us-east-1c, grupo-aislado

EC2-WEB-SERVER

▼ Configuraciones de red [Información](#)

VPC: **obligatorio** | [Información](#)

vpc-066662b1b43cc56c0 (predeterminado) ▾

Subred | [Información](#)

subnet-000d1117215ce99b3 [Crear nueva subred](#) ▾

VPC: vpc-066662b1b43cc56c0 Propietario: 772056227005
Zona de disponibilidad: us-east-1a Tipo de zona: Zona de disponibilidad
Direcciones IP disponibles: 4091 CIDR: 172.31.32.0/20

Asignar automáticamente la IP pública | [Información](#)

Habilitar

Se aplican cargos adicionales cuando no se cumplen los límites del nivel gratuito

EC2-BBDD

▼ Configuraciones de red [Información](#)

VPC: **obligatorio** | [Información](#)

vpc-066662b1b43cc56c0 (predeterminado) ▾

Subred | [Información](#)

subnet-0159958d113b9496e [Crear nueva subred](#) ▾

VPC: vpc-066662b1b43cc56c0 Propietario: 772056227005
Zona de disponibilidad: us-east-1b Tipo de zona: Zona de disponibilidad
Direcciones IP disponibles: 4091 CIDR: 172.31.0.0/20

Asignar automáticamente la IP pública | [Información](#)

Habilitar

Se aplican cargos adicionales cuando no se cumplen los límites del nivel gratuito

EC2-AISLADO

Configuraciones de red [Información](#)

VPC: **obligatorio** | [Información](#)

vpc-066662b1b43cc56c0	(predeterminado) ▾	
172.31.0.0/16		

Subred | [Información](#)

subnet-049540887ba2b91b6	
VPC: vpc-066662b1b43cc56c0 Propietario: 772056227005	
Zona de disponibilidad: us-east-1c Tipo de zona: Zona de disponibilidad	
Direcciones IP disponibles: 4091 CIDR: 172.31.80.0/20	

Asignar automáticamente la IP pública | [Información](#)

Habilitar	
-----------	--

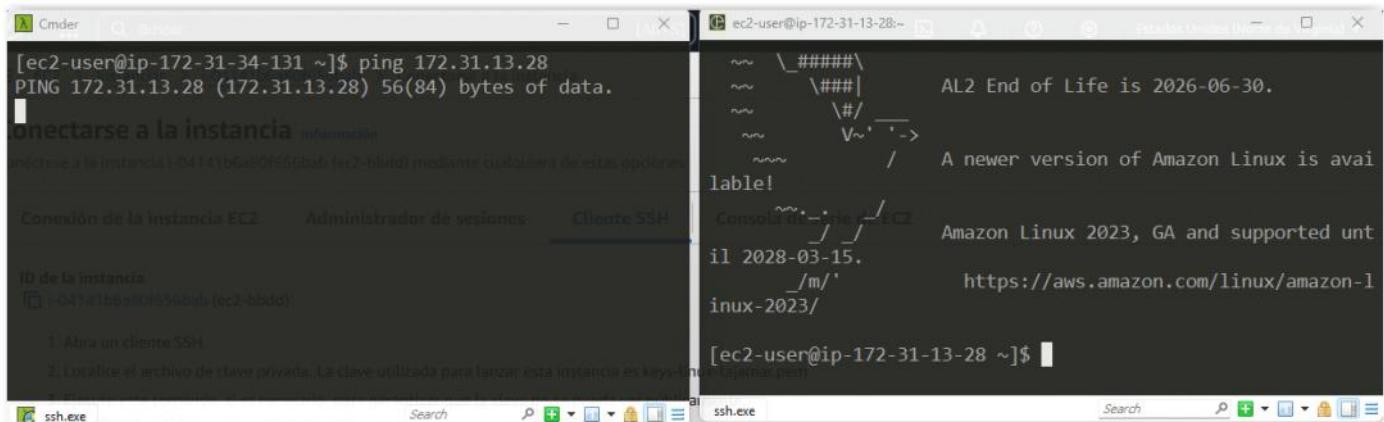
Vamos a visualizar la comunicación entre las máquinas.
Queremos comunicar nuestro Web Server con la BBDD dedicada.

Abrimos dos ventanas de CMDER para comunicar con cada máquina.

Izquierda: Web Server

Derecha: BBDD

Realizamos **ping** sobre cada máquina para comprobar la comunicación.
Podremos comprobar que NO tienen comunicación entre sí.



Para poder comunicar las dos máquinas, necesitamos incluirlas dentro de un mismo **Security Group**

Para ello, vamos a crear manualmente un nuevo grupo de seguridad llamado **grupo-virginia**

Detalles básicos

Nombre del grupo de seguridad [Información](#)

grupo-virginia

El nombre no se puede editar después de su creación.

Descripción [Información](#)

Grupo para Web Server y BBDD

VPC [Información](#)

vpc-066662b1b43cc56c0

Agregamos nuestras máquinas al nuevo grupo de Virginia

Instancias (1/5) Información			Última actualización Hace less than a minute	Conectar	Estado de la instancia ▾	Acciones ▾	Lanzar inst.
<input type="text" value="Buscar Instancia por atributo o etiqueta (case-sensitive)"/>							
<input type="checkbox"/> Name ▾			ID de la instancia	Estado de la i... ▾			
<input checked="" type="checkbox"/> ec2-bbdd			i-04141b6a80f656bab	En ejecución			
<input type="checkbox"/> linux-ec2-ebs-new			i-057ef55109bf4cffc	Terminada			
<input type="checkbox"/> ec2-ebs-eliminar			i-0e5cdc77f70cc6fc	Terminada			
<input type="checkbox"/> ec2-web-server			i-00cac3f	Cambiar grupos de seguridad Obtener la contraseña de Windows Modificar rol de IAM			
i-04141b6a80f656bab (ec2-bbdd)							
			Pol. de IAM	ID del propietario		Hora de lanzamiento	

Cambiar grupos de seguridad

Obtener la contraseña de Windows

Modificar rol de IAM

Conectar
Ver detalles
Administrar el estado de la instancia
Configuración de la instancia ▶
Redes ▶
Seguridad ▶
Imagen y plantillas ▶
Monitoreo y solución de problemas ▶

Estado de l...
Ver alarma
Ver alarma
Ver alarma
Ver alarma

Cambiar grupos de seguridad Información

Amazon EC2 evalúa todas las reglas de los grupos de seguridad seleccionados para controlar el tráfico entrante y saliente de la instancia. Puede utilizar esta ventana para agregar y eliminar grupos de seguridad.

Detalles de la instancia

- grupo-bbdd (sg-00f5a5b62e531b871)
- grupo-web-server (sg-0757af483048d213e)
- grupo-virginia (sg-094ee40e02ec77e02)
- grupo-aislado (sg-0aa22bb35aa305ca3)
- default (sg-0e7056c1ca62860a1)

8

Seleccionar grupos de seguridad

Agregar grupo de seguridad

Los grupos de seguridad asociados a la interfaz de red (eni-03b06372540b9dd38)

Grupos de seguridad asociados

Agregue uno o varios grupos de seguridad a la interfaz de red. También puede eliminar grupos de seguridad.

sg-094ee40e02ec77e02

X

Agregar grupo de seguridad

Los grupos de seguridad asociados a la interfaz de red (eni-03b06372540b9dd38)

ID de grupo de seguridad	Nombre del grupo de seguridad	Descripción	ID del propietario	
sg-00f5a5b62e531b871	grupo-bbdd	launch-wizard-1 created 2025-05-06T11:47:34.805Z	772056227005	<button>Eliminar</button>
sg-094ee40e02ec77e02	grupo-virginia	Grupo para Web Server y BBDD	772056227005	<button>Eliminar</button>

[Cancelar](#)

[Guardar](#)

Grupos de seguridad asociados

Agregue uno o varios grupos de seguridad a la interfaz de red. También puede eliminar grupos de seguridad.

sg-094ee40e02ec77e02

X

Agregar grupo de seguridad

Los grupos de seguridad asociados a la interfaz de red (eni-030d5c77245fd2ae4)

ID de grupo de seguridad	Nombre del grupo de seguridad	Descripción	ID del propietario	
sg-0757af483048d213e	grupo-web-server	launch-wizard-1 created 2025-05-06T11:46:06.854Z	772056227005	<button>Eliminar</button>
sg-094ee40e02ec77e02	grupo-virginia	Grupo para Web Server y BBDD	772056227005	<button>Eliminar</button>

Aunque estén dentro del mismo grupo de seguridad, con eso no basta, podemos comprobar que no tienen
Comunicación entre sí todavía.

Necesitamos poder realizar comunicación entre las máquinas
Para ello, necesitamos comunicar el Web Server con la base de datos.

Debemos crear una regla de entrada en el grupo-bbdd para admitir la comunicación del grupo de Virginia

Abrimos el grupo de seguridad de **grupo-bbdd**

i-04141b6a80f656bab (ec2-bbdd)

- sg-094ee40e02ec77e02 (grupo-virginia)
- sg-00f5a5b62e531b871 (grupo-bbdd)

Editar reglas de entrada Información

Las reglas de entrada controlan el tráfico entrante que puede llegar a la instancia.

Reglas de entrada <small>Información</small>	ID de la regla del grupo de seguridad	Tipo	Información	Protocolo	Intervalo de puertos	Origen	Información	Descripción: opcional
	sgr-00cbf0caa9d848303	SSH	Información	TCP	22	Pers...	0.0.0.0/0	Información
	-	Todo el tráfico	Información	Todo	Todo	Pers...	grupo	Eliminar
							Utilizar: "grupo"	
							Grupos de seguridad	
							grupo-virginia sg-094ee40e02ec77e02	
							grupo-aislado sg-0aa22bb35aa305ca3	
							grupo-web-server sg-0757af483048d213e	

Agregar regla

Y ya podremos comprobar la comunicación entre la máquina Web Server y BBDD, pero no a la inversa

The screenshot shows two terminal windows. The left window, titled 'Cmdr', displays the command 'ping 172.31.13.28' and its output, which shows successful ping results from the instance to the database server. The right window, titled 'ec2-user@ip-172-31-13-28 ~]', shows the command 'ping 172.31.34.131' and its output, which shows successful ping results from the database server back to the web server instance.

```
[ec2-user@ip-172-31-34-131 ~]$ ping 172.31.13.28
PING 172.31.13.28 (172.31.13.28) 56(84) bytes of data.
64 bytes from 172.31.13.28: icmp_seq=1 ttl=255 time=1.40 ms
64 bytes from 172.31.13.28: icmp_seq=2 ttl=255 time=1.77 ms
64 bytes from 172.31.13.28: icmp_seq=3 ttl=255 time=1.46 ms
64 bytes from 172.31.13.28: icmp_seq=4 ttl=255 time=1.46 ms
64 bytes from 172.31.13.28: icmp_seq=5 ttl=255 time=0.767 ms
^C
---172.31.13.28 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006 ms
rtt min/avg/max/mdev = 0.767/1.373/1.774/0.333 ms
[ec2-user@ip-172-31-34-131 ~]$ 
```

```
[ec2-user@ip-172-31-13-28 ~]$ ping 172.31.34.131
PING 172.31.34.131 (172.31.34.131) 56(84) bytes of data.
[ec2-user@ip-172-31-13-28 ~]$ 
```

IAM EC2

miércoles, 7 de mayo de 2025 9:12

Vamos a intentar utilizar el servicio S3 dentro de una máquina EC2, para comprobar qué tendríamos que hacer.

En este ejemplo, no vamos a implementar ninguna aplicación dentro de EC2, es un supuesto en el caso que

Tuvieramos que utilizar una App que, a su vez, utilizase el servicio S3 o cualquier otro.

Vamos a crear un servicio S3 y una máquina virtual.

Mediante comandos de AWS lo que haremos será visualizar los S3 que tengamos almacenados.

El comando para visualizar los S3 que tenemos.

aws s3 ls

Creamos una nueva máquina llamada **ec2-s3-delete**

También creamos un par de buckets

Buckets de uso general (2) [Información](#) Todas las regiones de AWS

Los buckets son contenedores de datos almacenados en S3.

Buscar buckets por nombre

Nombre	Región de AWS
buckets3delete-pgs	EE.UU. Este (Norte de Virginia) us-east-1
otrobucket-pgs	EE.UU. Este (Norte de Virginia) us-east-1

Lo primero que vamos a realizar es visualizar el listado de los buckets desde nuestra máquina de Tajamar

Para poder visualizar los buckets desde nuestra máquina de Tajamar, necesitamos un usuario con las Políticas de Buckets activas.

IAM Administrar el acceso a los recursos de AWS

Creamos un grupo

Nombre de la política ▲ | Tipo

Nombre de la política	Tipo
AmazonDMSRedshiftS3Role	Administrada por AWS
AmazonS3FullAccess	Administrada por AWS

Creamos un usuario llamado **user-s3-delete**

Especificar los detalles de la persona

Detalles de la persona

Nombre de usuario **user-s3-delete**

El nombre de usuario puede tener un máximo de 64 caracteres. Caracteres válidos: A-Z, a-z, 0-9, and + . @ _ - (guion)

Proporcione acceso de usuario a la consola de administración de AWS: **opcional**

Si proporciona acceso a la consola a una persona, se trata de una práctica recomendada para administrar su acceso en IAM Identity Center.

Si está creando acceso mediante programación a través de claves de acceso o credenciales específicas de servicios para AWS Cognito o Amazon Keyspaces, puede generarlos después de crear este usuario de IAM. [Más información](#)

Grupos de personas (1/1)

Buscar

Nombre del grupo	Personas	Políticas adjuntas	Creaciones
grupo-delete-s3	0	AmazonS3FullAccess	202

Necesitamos los permisos de acceso para la persona que hemos creado.

Claves de acceso (0)

Utilice las claves de acceso para enviar llamadas mediante programación a AWS desde AWS CLI, Herramientas de AWS para P llamadas directas a la API de AWS. Puede tener un máximo de dos claves de acceso (activas o inactivas) a la vez. [Más información](#)

No hay claves de acceso. Como práctica recomendada, evite el uso de credenciales a largo plazo, como las claves de acceso que proporcionen credenciales a corto plazo. [Más información](#)

[Crear clave de acceso](#)

Caso de uso

Interfaz de línea de comandos (CLI)

Tiene previsto utilizar esta clave de acceso para permitir que la AWS CLI obtenga acceso a su cuenta de AWS.

Si ejecutamos la instrucción dentro de la consola de comandos, no tenemos permisos

```
C:\> aws s3 ls
An error occurred (AccessDeniedException) when calling the ListBuckets operation: Access denied
Identity and Access Management (IAM)
Unable to locate credentials. You can configure credentials by running "aws configure".
Created: 2023-05-05 10:55:22 UTC Last modified: 2023-05-05 10:55:23 UTC
Último inicio de sesión: 2023-05-05 10:55:23 UTC
```

Ejecutamos **aws configure** para validar a nuestro usuario en el sistema.

```
C:\> aws configure
AWS Access Key ID [None]: AKIA3HQRDDS63IJHXZVV
AWS Secret Access Key [None]: YZ6uZmuyccW1JvKnNzTyc0c44lymkE
Default region name [None]: us-east-1
Default output format [None]: json
```

A continuación, volvemos a ejecutar el comando **aws s3 ls**

```
C:\> aws s3 ls
2025-05-07 09:19:46 buckets3delete-pgs
2025-05-07 09:20:14 otrobucket-pgs
```

El siguiente paso es visualizar qué sucede con una máquina virtual EC2

Nos conectamos a nuestra máquina creada previamente.

```
[ec2-user@ip-172-31-18-39 ~]$ aws s3 ls
Unable to locate credentials. You can configure credentials by running "aws configure".
```

Tenemos dos opciones:

- 1) **Aws configure:** Podemos utilizar nuestro usuario con permisos tal y como hemos hecho en la máquina de Windows.
- 2) **Role:** Mediante un role podemos asignar permisos entre servicios para que sean visibles solamente entre determinados servicios de AWS.

Entramos en IAM y Roles

Seleccionar entidad de confianza [Información](#)

Tipo de entidad de confianza

Servicio de AWS

Permita que servicios de AWS como EC2, Lambda u otros realicen acciones en esta cuenta.

Cuenta de AWS

Permitir a las entidades de otras cuentas de AWS que le pertenezcan a usted o a un tercero realizar acciones en esta cuenta.

Identidad web

Permite a las personas federadas por el proveedor de identidad web externo especificado asumir este rol para realizar acciones en esta cuenta.

Federación SAML 2.0

Permitir que las personas federadas con SAML 2.0 a partir de un directorio corporativo realicen acciones en esta cuenta.

Política de confianza personalizada

Cree una política de confianza personalizada para permitir que otras personas realicen acciones en esta cuenta.

Caso de uso

Permita que un servicio de AWS, como EC2, Lambda u otros, realicen acciones en esta cuenta.

Servicio o caso de uso

EC2

Elija un caso de uso para el servicio especificado.

Caso de uso

EC2

Allows EC2 instances to call AWS services on your behalf.

Políticas de permisos (1/1045) [Información](#)

Elija una o varias políticas para adjuntarlas al nuevo rol.

Filtrar por Tipo

<input type="text"/> s3	<input type="button"/>	Todos los tipos	13 cc
<input type="checkbox"/> Nombre de la política	<input type="button"/>	▲ Tipo	▼
<input type="checkbox"/>  AmazonDMSRedshiftS3Role		Administrada por AWS	

Detalles del rol

Nombre del rol

Ingrese un nombre significativo para identificar a este rol.

role-ec2-allow-s3

64 Caracteres máximos. Utilice caracteres alfanuméricos y '+ = ,_@-'.

Descripción

Agregue una breve explicación para este rol.

Allows EC2 instances to call AWS services on your behalf.

Máximo de 1000 caracteres. Utilice letras (A-Z, a-z), números (0-9), tabulaciones, nuevas líneas o cualquiera de los siguientes:

Ya tenemos el Role.

[role-ec2-allow-s3](#)

Dicho Role debemos asociarlo a la máquina/s que deseemos que se comuniquen con el servicio S3

Entramos dentro de nuestra máquina EC2 en AWS Console.

Instancias (1/1) [Información](#)

Última actualización Hace less than a minute

[Conectar](#)

Estado de la instancia

Acciones

Buscar Instancia por atributo o etiqueta (case-sensitive)

Name 

ID de la instancia

Estado de la i...

Conectar

Ver detalles

Administrar el estado de la instancia

Configuración de la instancia

Redes

Seguridad

Imagen y plantillas

Monitoreo y solución de problemas

ec2-s3-delete i-06703ac620f1ee45c

En ejecución 

Cambiar grupos de seguridad

Obtener la contraseña de Windows

Modificar rol de IAM

i-06703ac620f1ee45c (ec2-s3-delete)

Modificar rol de IAM [Información](#)

Asocie un rol de IAM a la instancia.

ID de la instancia

i-06703ac620f1ee45c (ec2-s3-delete)

Rol de IAM

Seleccione un rol de IAM para asociarlo a la instancia o cree uno nuevo si no ha creado ninguno. El rol que seleccione sustituirá todos los roles que estén asociados actualmente a la instancia.

role-ec2-allow-s3

[Crear un nuevo rol de IAM](#)

[Cancelar](#)

[Actualizar rol de IAM](#)

Y ya podremos visualizar los buckets desplegados

```
[ec2-user@ip-172-31-18-39 ~]$ aws s3 ls
2025-05-07 07:19:46 buckets3delete-pgs
2025-05-07 07:20:14 otrobucket-pgs
[ec2-user@ip-172-31-18-39 ~]$
```

Eliminamos los recursos utilizados.

EC2 NET CORE

miércoles, 7 de mayo de 2025 9:52

Ya sabemos que somos cobayas, es decir, estamos en la versión 9 que todavía no es la Oficial y a ver como se comporta nuestro amigo AWS con ella.

Necesitamos una máquina EC2 con un servidor que nos permita acceso a trabajar con Nuestra app de Net Core.

Como ya vimos en Azure, debemos desplegar nuestra App dentro de algún puerto y Realizar una redirección desde un servidor a ese puerto.

Las opciones más comunes son las siguientes:

- **Apache:** Es un servidor Web y nos permite tener el puerto 80 y 443 abiertos para Redirigir nuestras apps de net core.
- **Nginx:** Es un servidor que no sirve para páginas Web, en realidad, actúa como un Proxy inverso recuperando peticiones a su servidor y llevando dichas peticiones hacia Otros servicios: Net Core, Node...
- **Docker:** Mediante Docker también podemos desplegar aplicaciones dentro de Máquinas EC2

Trabajaremos con una máquina EC2 con el Ami **AWS Linux 2**

Realizaremos los siguientes pasos:

- 1) Actualizar la máquina EC2
- 2) Instalar Web Server Apache
- 3) Framework Net Core
- 4) Instalar Git

Comenzamos creando una nueva máquina Ec2 llamada **ec2-net-core**

The screenshot shows the AWS CloudFormation console with the 'Recientes' tab selected. A new stack named 'ec2-net-core' is being created. The 'Imagenes de máquina de Amazon (AMI)' section shows the selection of 'Amazon Linux 2 AMI (HVM - Kernel 5.10, SSD Volume Type)'. The stack status is 'En ejecución' and it is 'Creado'.

The screenshot shows the 'Firewall (grupos de seguridad)' section of the CloudFormation stack configuration. It includes options to 'Crear grupo de seguridad' or 'Seleccionar un grupo de seguridad existente'. A group named 'grupo-ec2-net-core' is selected. The group is described as 'Este grupo de seguridad se agregaría a todas las interfaces de red. El nombre no se puede editar después de crear el grupo de seguridad. La longitud máxima es de 255 caracteres. Caracteres válidos: a-z, A-Z, 0-9, espacios y _-/0#:@[]^&{}\$^'.

Nos conectamos a nuestra máquina y comenzamos.

sudo su

En esta máquina el comando para trabajar es **yum**

yum update -y

Instalamos Apache

yum install httpd -y

Instalamos Git

yum install git -y

Estamos en un sistema operativo CentOs y debemos buscar las librerías que correspondan a la versión en la que estamos trabajando.

Respecto a librerías, hablamos de Net Core para Linux

<https://learn.microsoft.com/en-us/linux/packages>

El siguiente paso es indicar las librerías que debemos utilizar para descargar Net Core

Gracias Dani y Carolina!!!

Actualizar el sistema:

sudo yum update -y

Instalar las dependencias necesarias:

sudo yum install gcc-c++-devel libcurl-devel openssl-devel zlib-devel libuuid-devel

Descargar e ejecutar el script de instalación de .NET:

wget https://dot.net/v1/dotnet-install.sh -O dotnet-install.sh chmod +x dotnet-install.sh

dotnet-install.sh --channel 9.0

Agregar .NET al PATH:

Para que el comando dotnet esté disponible en tu terminal, añade la ruta al PATH.

Para la sesión actual:

export PATH=\$HOME/.dotnet:\$PATH

Para hacerlo permanente, añade la línea anterior al final de tu archivo **~/.bashrc**

~/bashrc profile y luego ejecuta:

source ~/bashrc

Verificar la instalación:

dotnet --version

wget https://dot.net/v1/dotnet-install.sh -O dotnet-install.sh
chmod +x dotnet-install.sh
./dotnet-install.sh --channel 9.0 --quality preview --install-dir /usr/share/dotnet

ls /usr/share/dotnet
sudo ln -s /usr/share/dotnet/dotnet /usr/bin/dotnet

dotnet --version

COMANDOS PERMANENTES

```
Carolina.Penalba.Carpa 10:46
wget https://dot.net/v1/dotnet-install.sh
chmod +x dotnet-install.sh
./dotnet-install.sh --channel 9.0 --quality preview --install-dir /usr/share/dotnet
ls /usr/share/dotnet
sudo ln -s /usr/share/dotnet/dotnet /usr/bin/dotnet
dotnet --version
```

wget https://dot.net/v1/dotnet-install.sh
chmod +x dotnet-install.sh
./dotnet-install.sh --channel 9.0 --quality preview --install-dir /usr/share/dotnet
ls /usr/share/dotnet
sudo ln -s /usr/share/dotnet/dotnet /usr/bin/dotnet
dotnet --version

Se supone que es temporal esto, incluimos esta instrucción (**Gracias Aaron**)

```

echo "export PATH=$HOME/.dotnet:$PATH" >> ~/.bashrcsource ~/.bashrc

Iniciamos el servicio de Apache.

service httpd start

Comprobamos su estado

service httpd status

Incluimos las rutas para la redirección desde el puerto 80 de Apache al puerto 5000 de
Net Core

sudo iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 5000

Es el momento de crear una nueva App y probar si todo funciona correctamente.

Creamos un nuevo directorio llamado netcore

mkdir netcore

cd netcore

Creamos una nueva aplicación MVC

dotnet new mvc

Lanzamos nuestra App en el puerto 5000

```

`dotnet run --urls "http://0.0.0.0:5000"`

Entramos en nuestra IP con HTTP y comprobamos si visualizamos la aplicación Net Core.

Veremos que NO podemos acceder a nuestra máquina por el puerto 80 porque no está abierto

Editar reglas de entrada Información

Las reglas de entrada controlan el tráfico entrante que puede llegar a la instancia.

ID de la regla del grupo de seguridad	Tipo	Información	Protocolo	Intervalo de puertos	Origen	Información	Descripción: opcional
sgr-09fb2c4270e8a265a	SSH		TCP	22	Pers...	<input type="text" value="0.0.0.0"/> X	
-	HTTP		TCP	80	Any...	<input type="text" value="0.0.0.0"/> X	
						<input type="text" value="0.0.0.0"/> X	

[Agregar regla](#)

⚠️ Las reglas cuyo origen es 0.0.0.0/0 ::/0 permiten a todas las direcciones IP acceder a la instancia. Recomendamos configurar reglas de grupo de seguridad para permitir el acceso solo a las direcciones IP autorizadas.

El siguiente paso que vamos a realizar es "jugar" un poco.

Realizamos la práctica de ayer con servidores dedicados pero, esta vez, de forma real.

Tendremos un Web Server (ya lo tenemos)
Tendremos una máquina BBDD dedicada.

Desplegaremos todo en producción y haremos que se comuniquen por su IP privada y
Nosotros no tendremos acceso a la base de datos desde fuera cuando estemos en
Producción.

Trabajaremos con Postgres

<https://www.postgresql.org/>

Pondremos todos el mismo password: **postgres**

Abrimos la aplicación pgAdmin4

Usuario: **postgres**



Vamos a trabajar con la base de datos HOSPITAL y la tabla DEPT

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer

- Servers (1)
 - postrest17
 - Databases
 - Login/Groups
 - Create Database...
 - Tables
 - Refresh

Activity Stat

Dashboard

Query history

```

1 CREATE TABLE "DEPT"
2 ("DEPT_NO" int primary key not null,
3 "DNOMBRE" char(50),
4 "LOC" char(50));
5
6 insert into "DEPT" values(10,'DISNEYLAND','PARIS');
7 insert into "DEPT" values(20,'TERRA MÍTICA','BENIDORM');
8 insert into "DEPT" values(30,'PORT AVENTURA','TABARNAIA');
9 insert into "DEPT" values(40,'PARQUE WARNER','MADRID');
10
11 select * from "DEPT";

```

```

CREATE TABLE "DEPT"
("DEPT_NO" int primary key not null,
"DNOMBRE" char(50),
"LOC" char(50));

```

```
insert into "DEPT" values(10,'DISNEYLAND','PARIS');
```

```
insert into "DEPT" values(20,'TERRA MITICA','BENIDORM');
insert into "DEPT" values(30,'PORT AVENTURA','TABARNA');
insert into "DEPT" values(40,'PARQUE WARNER','MADRID');
```

select * from "DEPT";

Comenzamos creando una nueva aplicación MVC para comunicarnos con nuestro Postgres

Llamamos a la aplicación MvcNetCoreAWSPostgresEC2

 Microsoft.EntityFrameworkCoreCore 9.0.4 by aspnet, dotnetframework, EntityFramework, Microsoft, 1.49B downloads
Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations. EF Core works with SQL Server, Azure SQL Database, SQLite, Azure Cosmos DB, MySQL, PostgreSQL...

 Microsoft.EntityFrameworkCore.Tools 9.0.4 by aspnet, dotnetframework, EntityFramework, Microsoft, 413M downloads
Entity Framework Core Tools for the NuGet Package Manager Console in Visual Studio.

 Npgsql.EntityFrameworkCore.PostgreSQL 9.0.4 by Npgsql, roji, 251M downloads
PostgreSQL/Npgsql provider for Entity Framework Core.

Sobre Models creamos una nueva clase llamada Departamento

DEPARTAMENTO

```
[Table("DEPT")]
public class Departamento
{
    [Key]
    [Column("DEPT_NO")]
    public int IdDepartamento { get; set; }
    [Column("NOMBRE")]
    public string Nombre { get; set; }
    [Column("LOC")]
    public string Localidad { get; set; }
}
```

Creamos una carpeta llamada Data y una clase llamada HospitalContext

HOSPITALCONTEXT

```
2 references
public class HospitalContext: DbContext
{
    0 references
    public HospitalContext(DbContextOptions<HospitalContext> options)
        : base(options) { }

    0 references
    public DbSet<Departamento> Departamentos { get; set; }
}
```

Creamos una carpeta llamada Repositories y una clase llamada RepositoryHospitales

REPOSITORYHOSPITALES

```
public class RepositoryHospitales
{
    private HospitalContext context;

    public RepositoryHospitales(HospitalContext context)
    {
        this.context = context;
    }

    public async Task<List<Departamento>> GetDepartamentosAsync()
    {
        return await this.context.Departamentos.ToListAsync();
    }

    public async Task<Departamento> FindDepartamentoAsync(int idDepartamento)
    {
        return await this.context.Departamentos
            .FirstOrDefaultAsync(x => x.IdDepartamento == idDepartamento);
    }

    public async Task InsertDepartamentoAsync(int id, string nombre, string lo
calledad)
    {
        Departamento departamento = new Departamento();
        departamento.IdDepartamento = id;
        departamento.Nombre = nombre;
        departamento.Localidad = localidad;
        await this.context.Departamentos.AddAsync(departamento);
        await this.context.SaveChangesAsync();
    }
}
```

Sobre Controllers creamos un nuevo controlador llamado DepartamentosController

DEPARTAMENTOSCONTROLLER

```
public class DepartamentosController : Controller
{
    private RepositoryHospitales repo;

    public DepartamentosController(RepositoryHospitales repo)
    {
        this.repo = repo;
    }

    public async Task<IActionResult> Index()
    {
        List<Departamento> departamentos =
            await this.repo.GetDepartamentosAsync();
        return View(departamentos);
    }

    public async Task<IActionResult> Details(int id)
    {
        Departamento departamento =
            await this.repo.FindDepartamentoAsync(id);
        return View(departamento);
    }

    public IActionResult Create()
    {
        return View();
    }

    [HttpPost]
    public async Task<IActionResult> Create(Departamento dept)
    {
        await this.repo.InsertDepartamentoAsync(dept.IdDepartamento, dept.Nomb
re, dept.Localidad);
        return RedirectToAction("Index");
    }
}
```

Incluimos la cadena de conexión dentro de appsettings.json

APPSETTINGS.JSON

```
{
    "Logging": {
        "LogLevel": {
            "Default": "Information",
            "System": "Information",
            "Microsoft": "Information"
        }
    }
}
```

```

    "Default": "Information",
    "Microsoft.AspNetCore": "Warning"
},
"AllowedHosts": "*",
"ConnectionStrings": {
    "Postgres": "Host=localhost;Port=5432;Username=postgres;Password=postgres;Database=hospital"
}
}

```

Resolvemos las dependencias dentro de **Program**

PROGRAM

```

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
string connectionString =
    builder.Configuration.GetConnectionString("Postgres");
builder.Services.AddTransient<RepositoryHospitales>();
builder.Services.AddDbContext<HospitalContext>
    (options => options.UseNpgsql(connectionString));
builder.Services.AddControllersWithViews();

```

Y podemos comprobar la funcionalidad correcta

IdDepartamento	Nombre	Localidad	
10	DISNEYLAND	PARIS	Details
20	TERRA MITICA	BENIDORM	Details
30	PORT AVENTURA	TABARNIA	Details
40	PARQUE WARNER	MADRID	Details
50	Volcano Bay	FLORIDA	Details

El siguiente paso es publicar en Github como **PUBLIC** nuestro proyecto.

SERVIDOR DEDICADO EC2 BASE DE DATOS

El siguiente paso es crear una máquina EC2 que contendrá un servidor Postgres para trabajar con él.

Primero tendremos el servidor en desarrollo y nos conectaremos a él desde nuestro equipo de Windows.

Posteriormente, pondremos el servidor en Producción y no podremos conectar desde Windows, solamente desde otra máquina EC2.

Creamos un nuevo EC2 llamado **ec2-bbdd-postgres** con Amazon Linux 2

Imágenes de máquina de Amazon (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type
ami-085386e29e44dacd7 (64 bits (x86)) / ami-00bdc7ae558b8179f (64 bits (Arm))
Virtualización: hvm Activado para ENA: true Tipo de dispositivo raíz: ebs

Comenzamos entrando en nuestra VM.

sudo su

Amazon no tiene abierto todo el perfil de descargas de SO Linux, por medidas de seguridad.

Existe un Market interno para poder instalar elementos como bases de datos de terceros.

sudo amazon-linux-extras | grep postgres

```
[root@ip-172-31-90-3 ec2-user]# sudo amazon-linux-extras | grep postgres
59 postgresql13 available [ =stable ]
63 postgresql14 available [ =stable ]
```

sudo amazon-linux-extras install postgresql14 vim epel

Una vez descargado, lo instalamos con **yum**

yum install postgresql-server postgresql-devel -y

Arrancamos el servicio de base de datos

/usr/bin/postgresql-setup --initdb

Habilitamos el servicio de Postgres

systemctl enable postgresql

Arrancamos el servicio

```
systemctl start postgresql
```

```
[root@ip-172-31-90-3 ec2-user]# systemctl enable postgresql
Created symlink from /etc/systemd/system/multi-user.target.wants/postgresql.service to /usr/lib/systemd/system/postgresql.service.
[root@ip-172-31-90-3 ec2-user]# systemctl start postgresql
[root@ip-172-31-90-3 ec2-user]# systemctl status postgresql
● postgresql.service - PostgreSQL database server
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2025-05-08 09:17:42 UTC; 10s ago
     Process: 3570 ExecStartPre=/usr/bin/postmaster -D /var/lib/pgsql/data
   Main PID: 3572 (postmaster)
      CGroup: /system.slice/postgresql.service
              ├─3572 /usr/bin/postmaster -D /var/lib/pgsql/data
              ├─3576 postgres: logger
              ├─3578 postgres: checkpointer
              ├─3579 postgres: stats collector
              ├─3580 postgres: wal writer
              ├─3581 postgres: writer
              ├─3582 postgres: writer
              ├─3583 postgres: writer
              └─3584 postgres: writer
```

El siguiente paso es crear nuestra base de datos dentro de Postgres y montar todo.

Actualmente no podemos hacerlo con **pgAdmin4** aunque abramos los puertos del EC2 debido a que TODAS las bases de datos vienen con la comunicación cerrada en Linux, solamente para sus SO.

Conectamos a postgres

```
sudo su - postgres
```

Ponemos un Password a nuestro usuario de Postgres

```
psql -c "alter user postgres with password 'postgres'"
```

Creamos una nueva base de datos llamada **hospital**

```
createdb hospital
```

Nos conectamos como el usuario **postgres**

```
psql -U postgres
```

Nos conectamos a la base de datos **hospital**

```
psql postgres=#\c hospital;
```

```
[root@ip-172-31-90-3 ec2-user]# sudo su - postgres
su: invalid option -- 'o'
Try 'su --help' for more information.
[root@ip-172-31-90-3 ec2-user]# sudo su - postgres
-bash-4.2$ psql -c "alter user postgres with password 'postgres'"
ALTER ROLE
-bash-4.2$ createdb hospital
-bash-4.2$ psql -U postgres
psql (14.17)
Type "help" for help.

postgres=# psql postgres=#\c hospital;
You are now connected to database "hospital" as user "postgres".
hospital-#
```

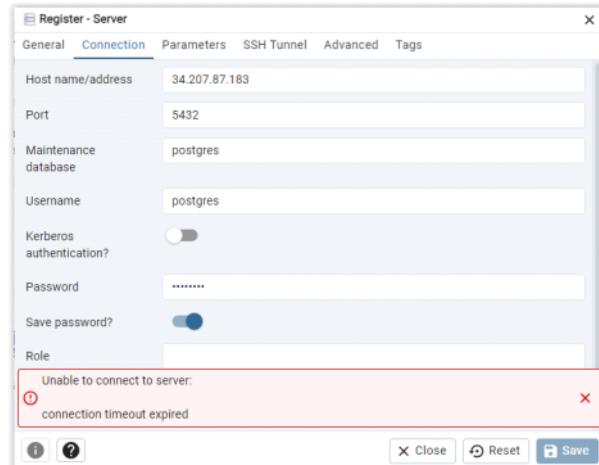
El siguiente paso es crear la tabla DEPT dentro de la máquina EC2.

```
hospital=# select * from "DEPT";
 DEPT_NO |          DNOMBRE          |          LOC
-----+-----+-----+
    10 | DISNEYLAND EC2           | PARIS
    20 | TERRA MITICA EC2         | BENIDORM
    30 | PORT AVENTURA EC2        | TABARNIA
    40 | PARQUE WARNER EC2        | MADRID
(4 rows)
```

Lo primero que queremos es probar nuestra aplicación MVC Net Core desde Windows y acceder al servidor EC2 que tenemos en la máquina del Postgres.

Vamos a conectar con **pgAdmin** y ver si tenemos acceso.

No podremos conectar porque No tenemos el puerto **5432** abierto en nuestro EC2



Abrimos el puerto para Postgres

Reglas de entrada						
ID de la regla del grupo de seguridad	Tipo	Protocolo	Intervalo de puertos	Origen	Descripción	
sgr-0d475ae2765bac11f	SSH	TCP	22	Persona...	0.0.0.0/0	
-	PostgreSQL	TCP	5432	Anywhere...	0.0.0.0/0	

La máquina ya acepta las conexiones a Postgres, pero Postgres es quién no acepta las conexiones todavía.

Debemos indicar que el servidor de BBDD acepte conexiones remotas.

Entramos en el directorio de configuración de Postgres

```
cd /var/lib/pgsql/data
```

Editamos el fichero de configuración de Postgres

nano postgresql.conf

```
# - Connection Settings -
listen_addresses = '*'  
#listen_addresses = 'localhost' # what IP address
# some connects
```

Editamos el fichero de clientes de escucha de PostgreSQL

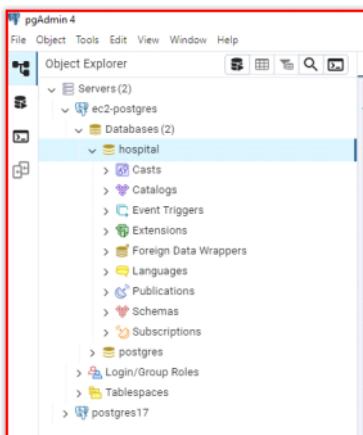
nano_pg_hba.conf

```
GNU nano 2.9.8                                     pg_hba.conf

# TYPE DATABASE USER ADDRESS METHOD
# "local" is for Unix domain socket connections only
local  all      all          peer
# IPv4 local connections:
host   all      all          127.0.0.1/32    ident
# IPv6 local connections:
host   all      all          ::1/128       ident
# Allow replication connections from localhost; by a user with the
# replication privilege.
local  replication all          peer
host   replication all          127.0.0.1/32    ident
host   replication all          ::1/128       ident
host   all      all          0.0.0.0/0     md5
```

Reiniciamos el servicio de Postgres

systemctl restart postgresql



El siguiente paso es comunicar nuestra app MVC con Postgres en el EC2.

Modificamos la cadena de conexión dentro de `appsettings.json`

```
        },
        "AllowedHosts": "*",
        "ConnectionStrings": {
            "Postgres": "Host=34.207.87.183;Port=5432;Username=postgres";
        }
    }
}
```

El siguiente paso es utilizar nuestro Web Server para conectar con la máquina EC2 de Base de datos.

Ponemos en producción nuestra nueva App.

Subimos a Github de nuevo la aplicación MVC.

Iniciamos nuestra máquina **ec2-net-core (casa)**

Iniciamos el servicio de Apache.

service httpd start

Comprobamos su estado

service httpd status

Incluimos las rutas para la redirección desde el puerto 80 de Apache al puerto **5000** de Net Core

sudo iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 5000

Es el momento de crear una nueva App y probar si todo funciona correctamente.

Creamos un nuevo directorio llamado **netcore**

mkdir netcore

cd netcore

Creamos una nueva aplicación MVC

dotnet new mvc

Lanzamos nuestra App en el puerto 5000

```
dotnet run --urls "http://0.0.0.0:5000"
```

Entramos en nuestra IP con HTTP y comprobamos si visualizamos la aplicación Net Core.

Veremos que NO podemos acceder a nuestra máquina por el puerto 80 porque no está abierto

COMANDOS IMPORTANTES CUANDO UTILIZAMOS IPTABLES

Comprobar que hemos puesto bien IPTABLES

sudo iptables -t nat -v -L -n --line-number

```
[root@ip-172-31-30-107 netcore]# sudo iptables -t nat -v -L -n --line-number
Chain PREROUTING (policy ACCEPT 1 packets, 40 bytes)
num  pkts bytes target     prot opt in   out    source         destination
 1      7  352 REDIRECT  tcp  --  eth0   *       0.0.0.0/0      0.0.0.0/0          tcp dpt:
80  redirect ports 5000

Chain INPUT (policy ACCEPT 8 packets, 392 bytes)
num  pkts bytes target     prot opt in   out    source         destination

Chain OUTPUT (policy ACCEPT 83 packets, 5713 bytes)
num  pkts bytes target    prot opt in   out    source         destination
```

Si nos hemos equivocado, eliminar el IPTABLES por su num

sudo iptables -t nat --delete PREROUTING 1

Vamos a intentar utilizar SSL, muchos servicios necesitan SSL para trabajar.

Instalamos el servicio de SSL

yum install mod_ssl -y

Deberemos crear un certificado de broma. Los certificados no son gratis.

Para tener un certificado, necesitamos una URL comprada. Existen servicios gratuitos que podemos utilizar, por ejemplo, DUCKDNS.

Entramos en la carpeta de certificados

cd /etc/pki/tls/certs

Generamos un nuevo certificado para nuestro servidor

sudo ./make-dummy-cert localhost.crt

Lo siguiente será añadir un nuevo iptables apuntando el puerto **443** hacia el puerto **5001** de Net Core

sudo iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 443 -j REDIRECT --to-port 5001

Por último, entramos de nuevo a nuestra App de Net Core

Lanzamos nuestra App mediante el puerto **443**, es decir, con HTTPS

dotnet run --urls "https://0.0.0.0:5001"

Abrimos el puerto HTTPS en nuestro Security Group

Editar reglas de entrada información

Las reglas de entrada controlan el tráfico entrante que puede llegar a la instancia.

Reglas de entrada		información		Protocolo		información		Intervalo de puertos		Origen		información	
ID de la regla del grupo de seguridad	Tipo	información		Protocolo		información		Intervalo de puertos		Origen		información	
sgr-09fb2c4270e8a263a	SSH			TCP				22		Persona...			
sgr-0be2abdd5f4e189e2	HTTP			TCP				80		Persona...			
-	HTTPS			TCP				443		Anywhe...			

Agregar regla

⚠️ Las reglas cuyo origen es 0.0.0.0/0 o ::/0 permiten a todas las direcciones IP acceder a la instancia. Recomendamos configurar reglas de grupo de seguridad para permitir el acceso único

El siguiente paso es desplegar nuestra aplicación de Postgres en la máquina EC2 del Web Server

Necesitaremos la URL del Git de nuestro proyecto

The screenshot shows a GitHub repository named 'MvcNetCoreAWSPostgresEC2'. The 'Code' dropdown menu is open, and the 'Clone' submenu is displayed. The URL 'https://github.com/serraguti/MvcNetCoreAWSPostgresEC2.git' is highlighted and has a 'Copy url to clipboard' button next to it. Other options in the submenu include 'Open with GitHub Desktop', 'Open with Visual Studio', and 'Download ZIP'.

Creamos una nueva carpeta

```
mkdir /var/core
```

```
cd /var/core
```

Clonamos nuestro Repo de Github mediante el siguiente comando

```
git clone URLgit
```

```
[root@ip-172-31-30-107 core]# git clone https://github.com/serraguti/MvcNetCoreAWSPostgresEC2.git
```

Entramos dentro de la carpeta hasta localizar el fichero csproj

```
dotnet restore
```

```
dotnet build
```

Por último, lanzamos nuestra App con HTTPS

```
dotnet run -urls "https://0.0.0.0:5001"
```

Y ya podremos visualizar nuestra App funcional

The screenshot shows a web application's index page. At the top, there is a navigation bar with icons for user profile, Home, Departamentos, Privacy, and a search bar. Below the navigation, the word 'Index' is displayed in large, bold, dark blue font. Underneath, there is a link 'Create New'. The main content is a table with four columns: 'IdDepartamento', 'Nombre', 'Localidad', and 'Details'. The table contains five rows of data:

IdDepartamento	Nombre	Localidad	Details
10	DISNEYLAND EC2	PARIS	<button>Details</button>
20	TERRA MITICA EC2	BENIDORM	<button>Details</button>
30	PORT AVENTURA EC2	TABARNIA	<button>Details</button>
40	PARQUE WARNER EC2	MADRID	<button>Details</button>
50	LEGOLAND	ORLANDO	<button>Details</button>

El último paso es proteger nuestro servidor de base de datos de forma que no se pueda entrar externamente.

¿Qué pasos debemos realizar?

- 1) No se pueda acceder de forma externa al servidor: Eliminamos la regla de entrada del grupo de base de datos.

Editar reglas de entrada Información

Las reglas de entrada controlan el tráfico entrante que puede llegar a la instancia.

Reglas de entrada Información

ID de la regla del grupo de seguridad	Tipo	Protocolo	Intervalo de puertos	Origen	Información	Descripción: opcional	Información
Información							
sgr-0d475ae2765bac11f	SSH	TCP	22	Persona...	<input type="text"/> 0.0.0.0/0	<input type="button" value="Eliminar"/>	<input type="button" value="Agregar regla"/>
sgr-02143879d353e5dd8	PostgreSQL	TCP	5432	Persona...	<input type="text"/> 0.0.0.0/0	<input type="button" value="Eliminar"/>	

⚠ Las reglas cuyo origen es 0.0.0.0/0 o ::/0 permiten a todas las direcciones IP acceder a la instancia. Recomendamos configurar reglas de grupo de seguridad para permitir el acceso únicamente desde direcciones IP conocidas.

- 2) Necesitamos que nuestro servidor Web Server acceda a la máquina EC2 dentro de AWS. ¿Qué hacemos?:

- a. Podriamos crear de nuevo la regla pero incluyendo solamente nuestro Web Server MVC.
- b. Podriamos comunicar todas las máquinas que deseásemos con la base de datos mediante un Security Group.

Vamos a utilizar la segunda opción, que se comuniquen mediante Security Groups

Creamos un nuevo Security Group llamado **grupo-seguridad-lumbago**

Crear grupo de seguridad Información

Un grupo de seguridad actúa como un firewall virtual para que la instancia controle el tráfico de entrada y salida. Para crear un nuevo grupo de seguridad, siga los pasos:

Detalles básicos

Nombre del grupo de seguridad Información

grupo-seguridad-lumbago

El nombre no se puede editar después de su creación.

Descripción Información

Comunicación entre máquinas con BBDD

VPC Información

vpc-066662b1b43cc56c0

Agregamos cada máquina que deseemos comunicar al grupo de Seguridad nuevo. Ec2-bbdd y ec2-net-core

Instancias (1/3) Información

Última actualización 🕒 Hace less than a minute							<input type="button" value="Conectar"/>	<input type="button" value="Estado de la instancia"/>	<input type="button" value="Acciones"/>	<input type="button" value="Lanzar"/>
	Name	ID de la instancia	Estado de la inst...	Tipo de inst...	Comprobación de...	Estado de la...				
<input checked="" type="checkbox"/>	ec2-net-core-casa	i-09dd09126f81ac848	En ejecución	t2.micro	2/2 comprobacion	<input type="button" value="Ver alarmas"/>				
<input type="checkbox"/>	ec2-net-core	i-0dbebf82e89791605	Detenida	t2.micro	-	<input type="button" value="Ver alarmas"/>				
<input type="checkbox"/>	ec2-bbdd-postgres	i-079e5e3549e1277e5	En ejecución	t2.micro	2/2 comprobacion	<input type="button" value="Ver alarmas"/>				

🕒 Hace less than a minute

Cambiar grupos de seguridad Información

Amazon EC2 evalúa todas las reglas de los grupos de seguridad seleccionados para controlar el tráfico entrante y saliente.

Detalles de la instancia

ID de la instancia

i-09dd09126f81ac848

Grupos de seguridad asociados

Agregue uno o varios grupos de seguridad a la interfaz de red. También puede eliminar grupos de seguridad.

Seleccionar grupos de seguridad

grupo-ec2-net-core { sg-04507b01b13e8b415 }

grupo-seguridad-bbdd (sg-0824e88e2bd2d4563)

default (sg-0e7056c1ca62860a1)

grupo-seguridad-lumbago (sg-0f2e48870b2059639)

Instancias (1/3) Información

Última actualización Hace less than a minute

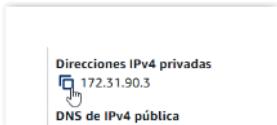
Name	ID de la instancia	Estado de la i...	Tipo de inst...	Comprobación de	Estado de la
ec2-net-core-casa	i-09dd09126f81ac848	En ejecución	t2.micro	2/2 comprobacion	Ver alarmas
ec2-net-core	i-0dbebf82e89791605	Detenida	t2.micro	-	Ver alarmas
ec2-bbdd-postgres	i-079e3e3549e1277e5	En ejecución	t2.micro	2/2 comprobacion	Ver alarmas

Acciones ▾

- Conectar
- Ver detalles
- Administrar el estado de la instancia
- Configuración de la instancia
- Redes
- Seguridad
- Imagen y plantillas
- Monitoreo y solución de problemas

Comprobamos si nos podemos conectar entre las máquinas.

Copiamos la IP privada de nuestra máquina de BBDD y hacemos ping



No existe comunicación entre las máquinas todavía

```
[root@ip-172-31-30-107 MvcNetCoreAWSPostgresEC2]# ping 172.31.90.3
PING 172.31.90.3 (172.31.90.3) 56(84) bytes of data.
```

En el grupo común debemos indicar que permita el tráfico entre las máquinas.

En el grupo de base de datos vamos a permitir el acceso del grupo lumbago.



Editar reglas de entrada Información

Las reglas de entrada controlan el tráfico entrante que puede llegar a la instancia.

Reglas de entrada Información

ID de la regla del grupo de seguridad	Tipo	Información	Protocolo	Intervalo de puertos	Origen	Información	Descripción: opcional	Información
sgr-0d475ae2765bac11f	SSH		TCP	22	Persona...	Q		
-	Todo el tráfico		Todo	Todo	Persona...	0.0.0.0/0 X		

Agregar regla

⚠️ Las reglas cuyo origen es 0.0.0.0/0 o ::/0 permiten a todas las direcciones IP acceder a la instancia. Recomendamos configurar reglas de grupo de seguridad más específicas.

0.0.0.0/16
 0.0.0.0/24
 0.0.0.0/32
 ::/0
 ::/16
 ::/32
 ::/48
 ::/64

Grupos de seguridad

- grupo-seguridad-lumbago | sg-0f2e48870b2059639
- grupo-ec2-net-core | sg-04507b01b13e8b415

Y ya tendremos acceso desde el Web Server hacia la base de datos.

```
[root@ip-172-31-30-107 MvcNetCoreAWSPostgresEC2]# ping 172.31.90.3
PING 172.31.90.3 (172.31.90.3) 56(84) bytes of data.
64 bytes from 172.31.90.3: icmp_seq=1 ttl=255 time=1.21 ms
64 bytes from 172.31.90.3: icmp_seq=2 ttl=255 time=0.985 ms
64 bytes from 172.31.90.3: icmp_seq=3 ttl=255 time=1.56 ms
64 bytes from 172.31.90.3: icmp_seq=4 ttl=255 time=1.18 ms
64 bytes from 172.31.90.3: icmp_seq=5 ttl=255 time=0.813 ms
```

Probamos nuestra App de Net Core

No podemos conectar. Nuestra App está utilizando la IP pública y eso quiere decir que cualquiera puede acceder.

Hemos restringido ese acceso, por lo que debemos modificar la cadena de conexión a **IP privada**

`nano appsettings.json`

```
    "AllowedHosts": "/*",
    "ConnectionStrings": {
        "Postgres": "Host=172.31.90.3;Port=5432;Username=postgres;Password"
    }
}
```

Y ya será funcional de nuevo solamente con acceso aislado a nuestras máquinas

Eliminamos las máquinas creadas

EC2 NET CORE 2023

viernes, 9 de mayo de 2025 9:17

Esta es la imagen AMI más nueva, en realidad, son todas iguales, un Linux con una máquina de motor de Amazon.

Es una copia de la máquina Ami Linux 2, pero trae en su interior instalado un **Fedora**, que es otra distribución de Linux.

Los comandos de instalación, en lugar de utilizar **yum**, utilizan el comando **dnf**

Si utilizamos un Linux Ubuntu/Debian, son **apt-get**

Vamos a crear una nueva máquina **Linux 2023**. Instalaremos un Net Core 9. Probaremos a instalar una nueva App y Lanzarla con HTTP.

Posteriormente, intentaremos utilizar HTTPS.

No tenemos **IPTABLES**

Vamos a utilizar un Servidor llamado **NGINX**. Dicho servidor se utiliza para redireccionar peticiones, si entra Una petición por algún lugar, enviamos dicha petición a otro lugar.

Comenzamos creando una nueva máquina EC2 llamada **ec2-viernes-papa**

The screenshot shows the AWS CloudFormation console interface. At the top, there's a navigation bar with tabs for 'Recientes' and 'Inicio rápido'. Below the navigation bar, there's a grid of icons representing different operating systems: Amazon Linux, macOS, Ubuntu, Windows, Red Hat, and SUSE Linux. To the right of the grid is a search bar labeled 'Buscar más AMI' and a note about including AWS Marketplace and community AMIs. The main area displays the 'Imágenes de máquina de Amazon (AMI)' section, specifically the 'AMI de Amazon Linux 2023' entry. This entry includes the AMI ID, a note that it's 'Apto para la capa gratuita', and details about its virtualization type (hvm), ENA support, and root device type (ebs). Below this, there are two sections for security groups. The first section, 'Regla del grupo de seguridad 2 (TCP, 80, 0.0.0.0/0)', has rules for HTTP (TCP port 80) from 0.0.0.0/0. The second section, 'Regla del grupo de seguridad 3 (TCP, 443, 0.0.0.0/0)', has a rule for HTTPS (TCP port 443) from 0.0.0.0/0. Both sections include fields for 'Tipo', 'Protocolo', 'Intervalo de puertos', 'Tipo de origen', 'Origen', and 'Descripción - opcional'.

Nos conectamos y vamos a instalar Net Core
Incluimos el repositorio para descargar Net Core 9.0

```
sudo wget -O /etc/yum.repos.d/microsoft-prod.repo
https://packages.microsoft.com/config/fedora/38/prod.repo
```

Instalamos Net Core 9.0

```
dnf install dotnet-sdk-9.0 -y
```

Para las redirecciones sin iptables necesitamos instalar **libicu**

```
dnf install -y libicu
```

El siguiente paso es instalar el servidor **nginx**

```
dnf install nginx -y
```

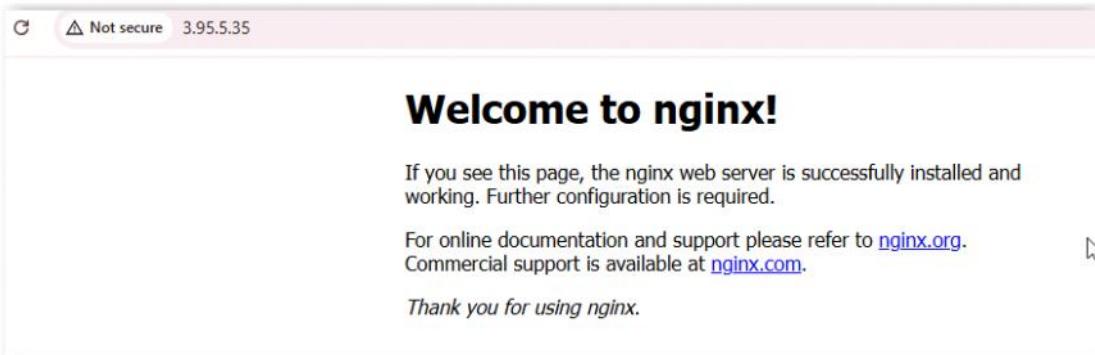
Habilitamos el nuevo servicio

```
systemctl enable nginx
```

Iniciamos el servicio

```
systemctl start nginx
```

Si ponemos nuestra IP pública, deberíamos visualizar el Servidor running



Cuando creamos una aplicación Net Core, necesitamos que el servidor Nginx nos lleve a nuestra URL del Proyecto Net Core.

Para ello, debemos configurar el servidor Nginx con la redirección de forma que, al entrar por el puerto 80, Nos lleve a la dirección del Servidor y puerto Net Core.

Entramos en la siguiente dirección de Linux

```
cd /etc/nginx
```

Editamos el fichero de configuración de **nginx**

```
nano nginx.conf
```

Necesitamos configurar la siguiente estructura:

```
server {
    listen      80;
    listen      [::]:80;
    server_name _;
    root        /usr/share/nginx/html;

    # Load configuration files for the default server block.
    include     /etc/nginx/default.d/*.conf;

    error_page 404 /404.html;
    location = /404.html {
    }
}
```

Documentación:

```
server_name your-domain.com www.your-domain.com;
```

```
location / {
proxy_pass http://127.0.0.1:your-app-port;
proxy_http_version 1.1;
proxy_set_header Upgrade $http_upgrade;
proxy_set_header Connection keep-alive;
proxy_set_header Host $host;
proxy_cache_bypass $http_upgrade;
}
```

Modificamos para entrar en nuestra aplicación Net Core en el puerto **5000**

```
location / {
proxy_pass http://127.0.0.1:5000;
proxy_http_version 1.1;
proxy_set_header Upgrade $http_upgrade;
proxy_set_header Connection keep-alive;
proxy_set_header Host $host;
proxy_cache_bypass $http_upgrade;
}
```

Este es el resultado

```

server {
    listen      80;
    listen      [::]:80;
server_name example.com, www.example.com;
location / {
Upgrade_Skip_Handoff;
proxy_pass http://127.0.0.1:5000;
proxy_http_version 1.1;
proxy_set_header Upgrade $http_upgrade;
proxy_set_header Connection keep-alive;
proxy_set_header Host $host;
proxy_cache_bypass $http_upgrade;
}

```

El siguiente paso es probar nuestra App Net Core.

```
cd /home/ec2-user
```

```
mkdir netcore
```

```
cd netcore
```

```
dotnet new mvc
```

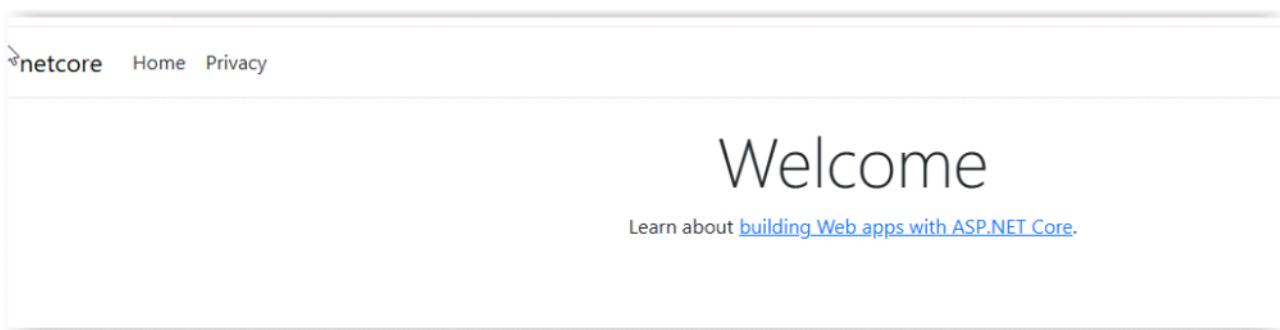
Reiniciamos nginx

```
systemctl restart nginx
```

Lanzamos nuestra app net core

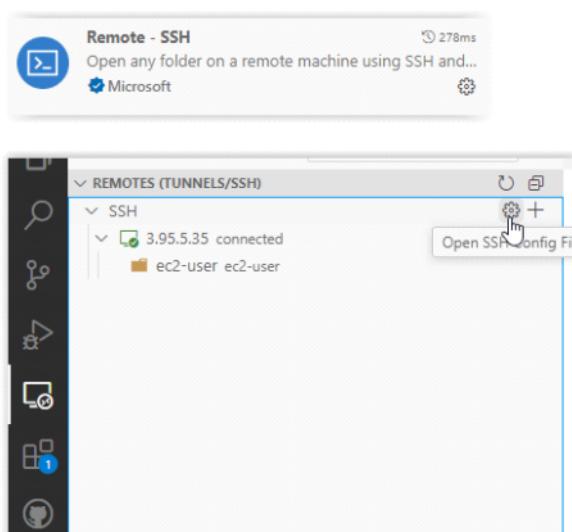
```
dotnet run --urls "http://0.0.0.0:5000"
```

Y podremos visualizar nuestra App perfectamente funcional



También podemos acceder a esta máquina mediante **Remote SSH** con VS Code.

Abrimos VS Code



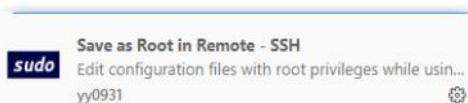
```

1 # Read more about SSH config files: https://linux.die.net/man/5/ssh\_config
2 Host 3.95.5.35
3   HostName 3.95.5.35
4   IdentityFile C:\KEYS\keys-linux-tajamar.pem
5   User ec2-user

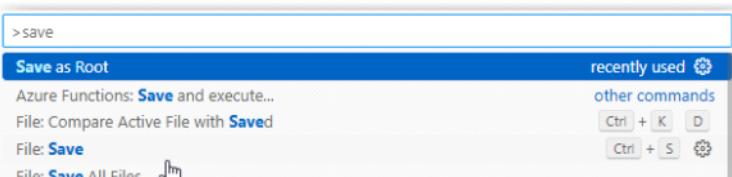
```

Y podremos conectarnos

Si queremos guardar ficheros en remoto, necesitamos a **sudo**, para hacerlo desde VS Code



Modificamos los ficheros que deseemos y, posteriormente, pulsamos **F1**, entramos en La paleta de **VS Code** y escribimos **Save as**



Proceso activo en EC2 Linux

Si necesitamos dejar activo nuestro Servicio de Net Core, es decir, tener el dotnet lanzado y Poder realizar más acciones sobre el SO.

Esta instrucción nos permite incluir un comando dentro de los servicios activos de Linux Para que siempre esté activo aunque salgamos de la máquina.

nohup dotnet run --urls "http://0.0.0.0:5000" > /dev/null 2>&1 &

Nos habrá dado un PID del proceso activo creado

```

[root@ip-172-31-82-24 netcore]# nohup dotnet run --urls "http://0.0.0.0:5000" > /dev/null 2>&1 &
[1] 26761
[root@ip-172-31-82-24 netcore]#

```

Con este comando podemos visualizar los procesos activos

ps -ef

```

root      26502      1  0 09:18 ?        00:00:00 nginx: master process /usr/sbin/nginx
nginx    26363  26362  0 09:18 ?        00:00:00 nginx: worker process
root      26507      2  0 09:20 ?        00:00:00 [kworker/0:1-cgroup_destroy]
root      26698  2434  0 09:24 ?        00:00:00 systemd-userwork: waiting...
root      26699  2434  0 09:24 ?        00:00:00 systemd-userwork: waiting...
root      26700  2434  0 09:24 ?        00:00:00 systemd-userwork: waiting...
root      26760      2  0 09:25 ?        00:00:00 [kworker/0:0-events_power_efficient]
root      26761  2535  3 09:25 pts/1    00:00:02 dotnet run --urls http://0.0.0.0:5000
root      26787  26761  1 09:25 pts/1    00:00:01 /home/ec2-user/netcore/bin/Debug/ne
root      26827  2535  0 09:26 pts/1    00:00:00 ps -ef
[root@ip-172-31-82-24 netcore]#

```

Con Kill y PID matamos el proceso que deseemos

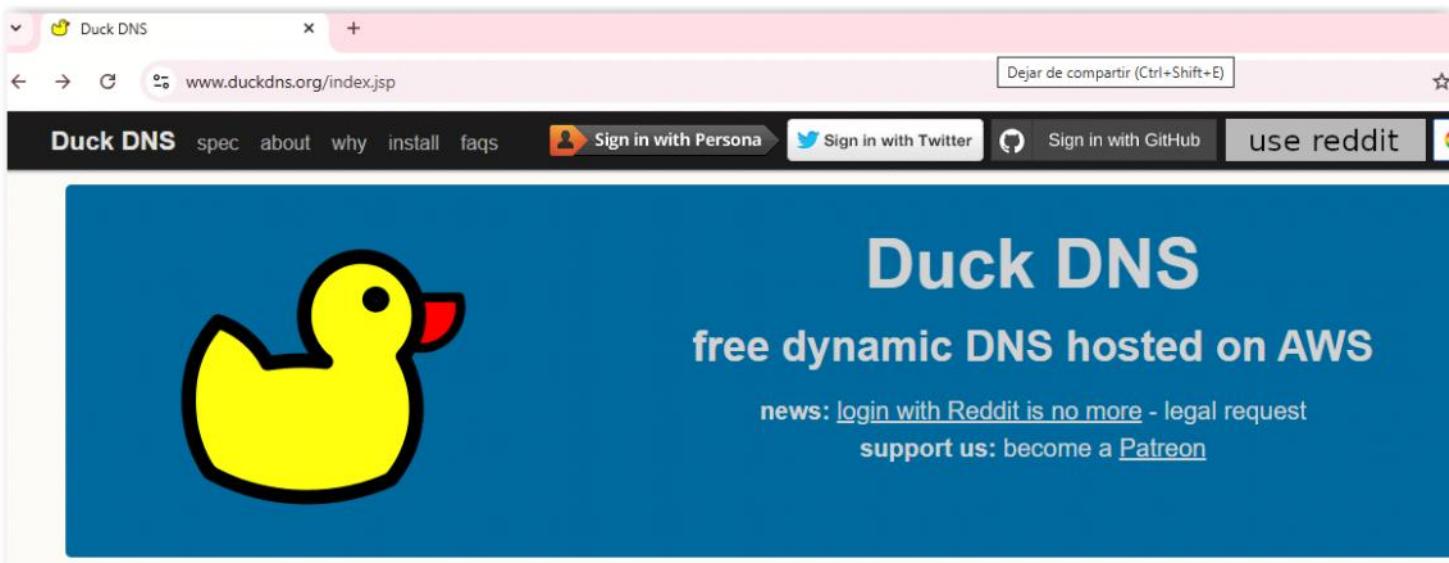
kill PID

Como ha comentado Amanda (muchas gracias!!!!) con -9 lo finaliza SI o SI

kill -9 PID

El siguiente paso que vamos a intentar es utilizar un dominio gratuito dentro de nuestro Servidor EC2.

Nos validamos dentro de **duckdns.org**



<https://www.duckdns.org/>

Lo que tenemos que hacer es crearnos un dominio, son 5 gratuitos

Nos creamos un dominio.

Copiamos nuestra IP pública del EC2 y cruzamos dedos...

Y podremos visualizar nuestra aplicación funcionando correctamente

Todavía no tenemos HTTPS, simplemente un dominio.

¿Qué tendríamos que hacer para tener un HTTPS?

- 1) Nginx llevarlo al 443 en lugar del 80
- 2) Nginx para poder utilizarlo con HTTPS necesita el certificado del dominio dónde Vamos a redireccionar.
- 3) Lanzar el servicio dotnet con https en el puerto 5001

EC2 DOCKER

Lunes, 12 de mayo de 2025 9:32

Vamos a desplegar una máquina virtual EC2 en la que implementaremos Docker y docker Compose para
Un docker de tipo Nginx Reverse proxy que apuntará a nuestra IP de la máquina y generar un certificado SSL
Con Letscript.

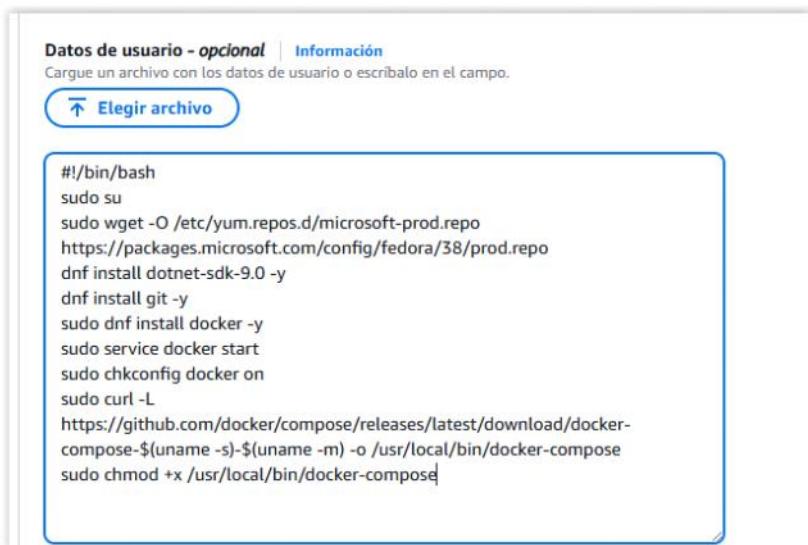
Comenzamos creando una nueva máquina llamada **ec2-docker**

Vamos a crear un grupo de seguridad y abrimos los siguientes puertos:

- HTTP: 80
- HTTPS: 443
- TCP Personalizado: 9000 (**Portainer**)
- TCP Personalizado: 81 (**Nginx**)
- TCP Personalizado: 5000 (**Net Core**)

Utilizamos el BASH nuevo que os he dejado...

[script_EC2_AWS_2023.txt](#)



Vamos a instalar las siguientes características:

- 1) Portainer: Es un administrador de imágenes docker de forma visual
- 2) Net Core
- 3) Nginx Reverse Proxy: Nuestro servidor virtual de redirecciones a la IP de nuestra máquina EC2

[How To configure Docker & Docker-Compose in AWS EC2 \[Amazon Linux 2023 AMI\] | by Freddy Manrique | Medium](#)



sudo curl -L [https://github.com/docker/compose/releases/latest/download/docker-compose-\\$\(uname -s\)-\\$\(uname -m\).tar.gz](https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m).tar.gz) -o /usr/local/bin/docker-compose

Docker-Compose

```
# docker-compose (latest version)
$-> sudo curl -L https://github.com/docker/compose/releases/latest/download/dock
# Fix permissions after download
$-> sudo chmod +x /usr/local/bin/docker-compose
# Verify success
$-> docker-compose version
```

Top h

Instalamos Portainer

```
docker pull portainer/portainer-ce:latest
```

Desplegamos Portainer

```
docker run -d -p 9000:9000 --restart always -v /var/run/docker.sock:/var/run/docker.sock
portainer/portainer-ce:latest
```

Entramos en Portainer y ponemos como password **Portainer12345** con el user **admin**

⚠ No es seguro 3.208.19.64:9000/#!/init/admin

New Portainer installation

Please create the initial administrator user.

Username: admin

Password: [REDACTED]

Confirm password: [REDACTED] ✓

⚠ The password must be at least 12 characters long. ✓

Create user

The screenshot shows the Portainer.io interface. On the left is a dark blue sidebar with navigation links: Home, Dashboard, Templates, Stacks, Containers, Images, Networks, Volumes, Events, Host, and Administration. The 'Containers' link is currently active. The main content area has a light blue header 'Environments' and a 'Home' section. Below it is a 'Latest News From Portainer' box. The 'Environments' section contains a table with one row for the 'local' environment. The 'local' environment row includes columns for Platform (local), Status (Up), Created (2025-05-14 12:38:51), Type (Standalone), Version (25.0.8), and Docker socket (/var/run/docker.sock). It also shows Group: Unassigned, No tags, Local, 0 stacks, 1 container (with status icons: green, red, yellow, orange), 1 volume, and 1 image (995.5 MB RAM).

Copiamos el `docker-compose.yml` de nginx

DOCKER-COMPOSE

```

services:
  nginx:
    image: jc21/nginx-proxy-manager:latest
    restart: always
    ports:
      # These ports are in format <host-port>:<container-port>
      - 80:80 # Public HTTP Port
      - 443:443 # Public HTTPS Port
      - 81:81 # Admin Web Port
      # Add any other Stream port you want to expose
      # - '21:21' # FTP
    environment:
      # Postgres parameters:
      DB_POSTGRES_HOST: dbpostgres
      DB_POSTGRES_PORT: "5432"
      DB_POSTGRES_USER: npm
      DB_POSTGRES_PASSWORD: password
      DB_POSTGRES_NAME: npm
      # Uncomment this if IPv6 is not enabled on your host
      DISABLE_IPV6: "true"
    volumes:
      - ./data:/data
      - ./letsencrypt:/etc/letsencrypt
    depends_on:
      - dbpostgres
  dbpostgres:
    image: postgres:latest
    restart: always
    environment:
      POSTGRES_USER: npm
      POSTGRES_PASSWORD: password
      POSTGRES_DB: npm
    volumes:
      - ./postgres:/var/lib/postgresql/data
  networks: {}

```

Name

This stack will be deployed using `docker compose`.

Build method

Web editor Upload Repository

Web editor

You can get more information about Compose file format in the [official documentation](#).

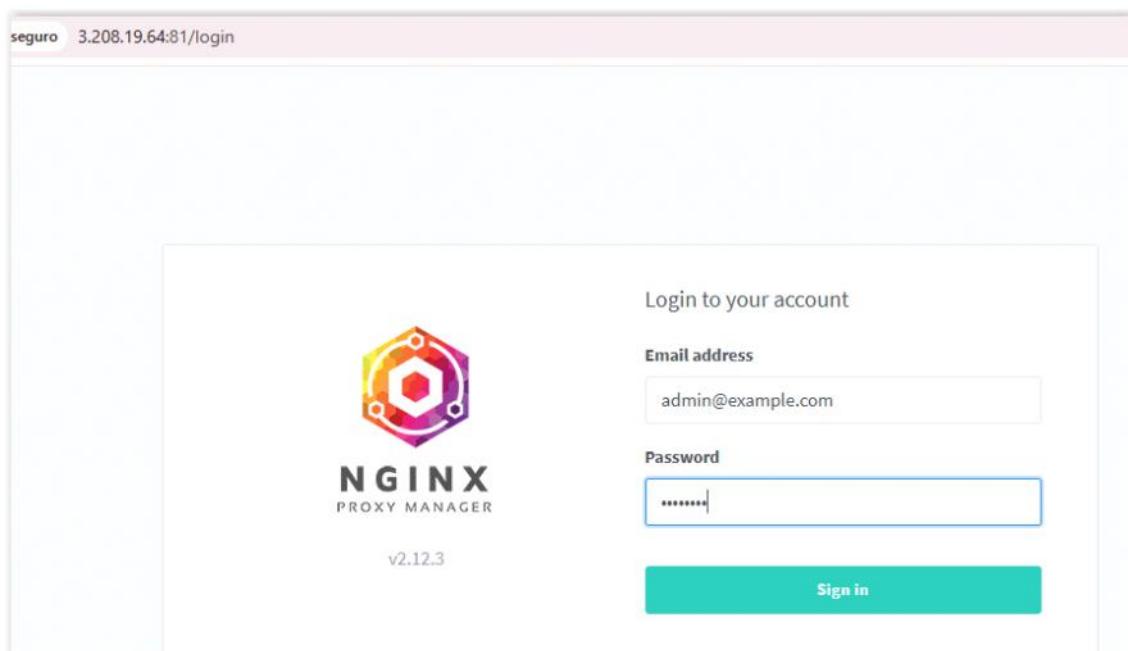
Define or paste the content of your docker compose file here

```
1 services:
2   nginx:
3     image: jc21/nginx-proxy-manager:latest
4     restart: always
5     ports:
6       # These ports are in format <host-port>:<container-port>
7       - 80:80 # Public HTTP Port
8       - 443:443 # Public HTTPS Port
9       - 91:91 # Admin Web Port
```

Hemos terminado con Portainer. Ahora entramos en **NGINX** con el puerto **81**

Email: **admin@example.com**

Password: **changeme**



Vamos a poner el mismo Password: **Portainer12345**

Debemos poner el correo de **DUCKDNS**

Edit User

Full Name *	Nickname *
Administrator	Admin
Email *	
pacoserranox@gmail.com	
<input type="button" value="Cancel"/> <input type="button" value="Save"/>	

Entramos en SSL Certificates

Abrimos la página de [duckdns.org](https://www.duckdns.org/)

<https://www.duckdns.org/>

Creamos un dominio cada uno....

Debemos poner la IP de la máquina donde deseemos trabajar. Nuestro MVC, IP pública.

domain	current ip	ipv6	changes
puente-zanahoria	3.208.19.64	<input type="button" value="update ip"/>	<input type="button" value="update ipv6"/>

El siguiente paso es configurar y solicitar un certificado SSL en nuestra máquina mediante NGINX.

Add Let's Encrypt Certificate

Domain Names *	
puente-zanahoria.duckdns.org	
⚠ These domains must be already configured to point to this installation	
Email Address for Let's Encrypt *	
pacoserranox@gmail.com	
<input checked="" type="checkbox"/> Use a DNS Challenge	
⚠ This section requires some knowledge about Certbot and its DNS plugins. Please consult the respective plugins documentation.	
DNS Provider *	
Please Choose...	

Después de un tiempo, debería crearnos un certificado



Dashboard Hosts Access Lists SSL Certificates Users Audit Log Settings

SSL Certificates

Search Certificate...



Add SSL Cert

NAME	CERTIFICATE PROVIDER	EXPIRES	STATUS
puente-zanahoria.duckdns.org Created: 14th May 2025	Let's Encrypt - DuckDNS	12th August 2025, 11:56 am	● Inactive

El siguiente paso es crear un HOST que utilizará nuestro duckdns.

Duckdns nos proporciona una dirección URL que contiene un servidor HTTPS válido.

Dicha URL debe direccionar internamente dentro de nuestra máquina.

Nosotros, en la máquina MVC vamos a lanzar la aplicación mediante **http** y al puerto **5000**



Dashboard Hosts Access Lists SSL Certificates Users Audit Log Settings

SSL Certific	Proxy Hosts
	Redirection Hosts
NAME puer	Streams 404 Hosts

Created: 14th May 2025

Search Certificate...



Add SSL Cert

New Proxy Host

X

↳ Details ⚒ Custom locations ⚒ SSL ⚒ Advanced

Domain Names *

puente-zanahoria.duckdns.org

Scheme *

http

Forward Hostname / IP *

3.208.19.64

Forward Port *

5000

Cache Assets

Block Common Exploits

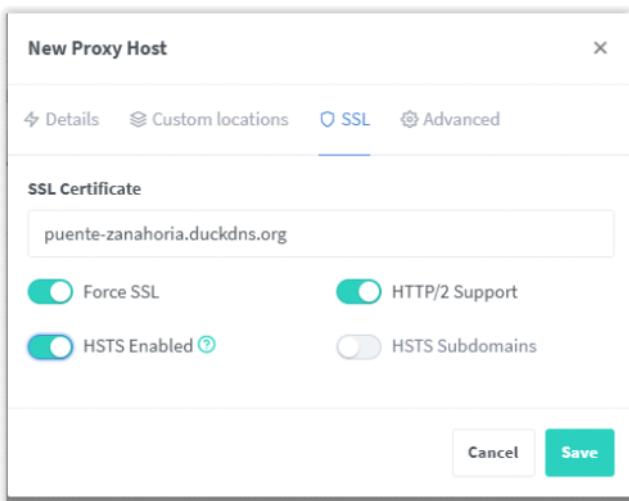
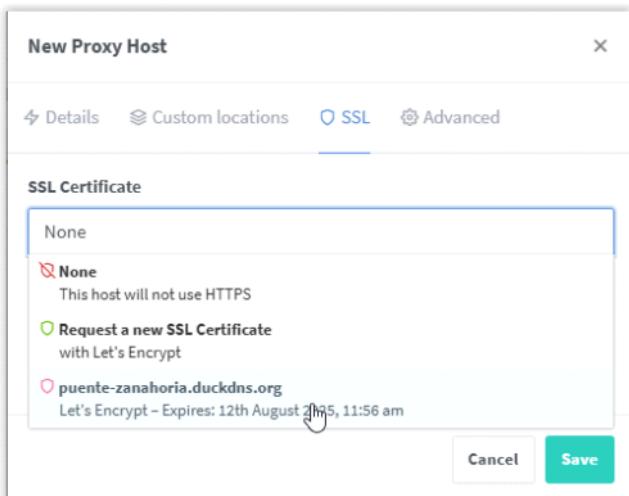
Websockets Support

Access List

Publicly Accessible

Cancel

Save



Ya tendremos nuestra IP HTTP hacia el dominio HTTPS.

Para probarlo, necesitamos el servicio MVC en la máquina.

Creamos un directorio, y ejecutamos **dotnet restore** y **build**

Lanzamos nuestro servicio con **HTTP** y puerto **5000**

```
dotnet run --urls "http://0.0.0.0:5000"
```

```
Build succeeded with 5 warning(s) in 11.1s
[root@ip-172-31-24-236 MvcNetCorePersonajesAWS]# dotnet run --urls "http://0.0.0.0:5000"
Using launch settings from /home/ec2-user/core/MvcNetCorePersonajesAWS/MvcNetCorePersonajesAWS/Properties/launchSettings.json...
Building...
warn: Microsoft.AspNetCore.DataProtection.KeyManagement.XmlKeyManager[35]
      No XML encryptor configured. Key {93388374-336c-417b-af56-1873085d7b5a} may be persisted to storage in an unencrypted form.
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://0.0.0.0:5000
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: /home/ec2-user/core/MvcNetCorePersonajesAWS/MvcNetCorePersonajesAWS
```

Probamos a conectar con la IP pública y el puerto **5000**

Y lo tenemos!!!!

IdPersonaje	Nombre
1	Koothrappali

PRACTICA LINUX

Necesito montar un servidor EC2 de base de datos dedicado con **MariaDb**

Instalar **MariaDb** en un Linux CentOS Amazon Linux 2.

Debemos crear la bbdd **hospital** e incluir la tabla **DEPT**

Modificamos nuestra App de Postgres del viernes para que se conecte a **MariaDb** y Comprobamos que en Local nos funciona.

Instalamos nuestra App de **MariaDb** en nuestro Web Server y vemos si Todo funciona en producción.

Instalamos Maria Db de Amazon

```
amazon-linux-extras install mariadb10.5 vim epel
```

```
yum install -y mariadb-server mariadb-devel
```

Iniciamos nuestro servidor

```
systemctl start mariadb
```

La primera vez, el usuario **root** no tiene password

Podemos utilizar una configuración de seguridad mediante un fichero
mysql_installation

Entramos como el usuario root

```
mariadb -u root -p
```

Vamos a crearnos un usuario con privilegios y será el que utilizaremos para Conectarnos desde Workbench o desde Net Core

```
CREATE USER 'adminmariia'@'localhost' IDENTIFIED BY 'mariadb';
```

Damos privilegios a dicho usuario

```
GRANT ALL PRIVILEGES ON *.* TO 'adminmariia'@'localhost';
```

Guardamos los cambios

```
FLUSH PRIVILEGES;
```

Salimos de Maria Db

```
exit;
```

Nos conectamos con el admin nuevo para ver si tiene permisos.

```
mariadb -u adminmariia -p
```

Ejecutamos algún comando **show databases;**

Creamos nuestra base de datos Hospital

```
create database hospital;
```

```
use hospital;
```

Ejecutamos el script de creación de DEPT

```
CREATE TABLE DEPT (
```

```

DEPT_NO int NOT NULL primary key,
DNOMBRE varchar(150),
LOC varchar(150)
);

insert into DEPT values (10, 'DISNEYLAND MARIA EC2', 'PARIS');
insert into DEPT values (20, 'TERRA MITICA MARIA EC2', 'BENIDORM');
insert into DEPT values (30, 'PORT AVENTURA MARIA EC2', 'TABARNIA');
insert into DEPT values (40, 'PARQUE WARNER MARIA EC2', 'MADRID');
commit;

```

Veremos que los datos ya están insertados



```

CREATE TABLE DEPT (
Database changed
MariaDB [hospital]> select * from DEPT;
+-----+-----+-----+
| DEPT_NO | DNOMBRE | LOC      |
+-----+-----+-----+
|      10 | DISNEYLAND MARIA EC2 | PARIS    |
|      20 | TERRA MITICA MARIA EC2 | BENIDORM |
|      30 | PORT AVENTURA MARIA EC2 | TABARNIA |
|      40 | PARQUE WARNER MARIA EC2 | MADRID   |
+-----+-----+-----+
4 rows in set (0.000 sec)

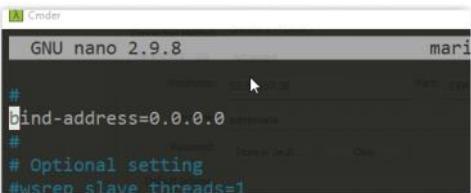
```

Debemos indicar, dentro del fichero de configuración de Maria Db que Admitiremos conexiones desde cualquier lugar externo.

```
cd /etc/my.cnf.d
```

Editamos el fichero de configuración

```
nano mariadb-server.cnf
```



```

# bind-address=0.0.0.0
# 
# Optional setting
#wsrep_slave_threads=1

```

Reiniciamos el Servidor de Maria Db

```
systemctl restart mariadb
```

```
systemctl status mariadb
```

Por último, entramos en Maria Db para permitir que nuestro usuario admite Peticiones externas (Utilizamos la IP del error de antes)

```
mariadb -u root -p
```

Ejecutamos la siguiente instrucción

```
GRANT ALL ON *.* to 'adminmaria'@'188.26.196.171' IDENTIFIED BY 'mariadb' WITH GRANT OPTION;
```

Probamos a conectar desde Workbench con nuestro usuario **adminmaria**

Y lo tenemos!!!!

EC2 RDS

martes, 13 de mayo de 2025 9:09

Relational Database Service

Es un servicio dedicado a base de datos.

No todas las bases de datos son gratuitas, debemos mirarlo bien.

La base de datos por excelencia en AWS se llama **Aurora**, que en realidad, tiene el mismo motor Que MySql/MariaDb.

Tipos de bases de datos:

- **Aurora**: La base de datos propia de AWS.
- **DynamoDb**: Es la base de datos NoSQL de AWS.
- **Amazon Elastic Cache**: Es la base de datos temporal de AWS, igual que en Azure, para Poder compartir sesiones entre dispositivos.
- **Amazon TimeStream**: Es una base de datos creada para grandes ficheros y trabajar Con Multimedia, pero no es gratis.

Vamos a trabajar con **MySql**.

Comenzaremos creando una nueva base de datos, posteriormente, comunicaremos con Dicha base de datos mediante Workbench.

Una vez que tengamos la base de datos, haremos un Api para consumir datos y lo desplegamos En una máquina EC2.

Nota: Tenemos 750 horas gratis para jugar con bases de datos. Es igual que las máquinas, Si tenemos dos bbdd, se suman sus horas.

Solamente consumen cuando están encendidas.

Super importante: La base de datos si se detiene, se vuelve a encender SOLA a la semana.

Los nombres de servidor, como en Azure, son UNICOS

Creamos un servidor con los siguientes datos:

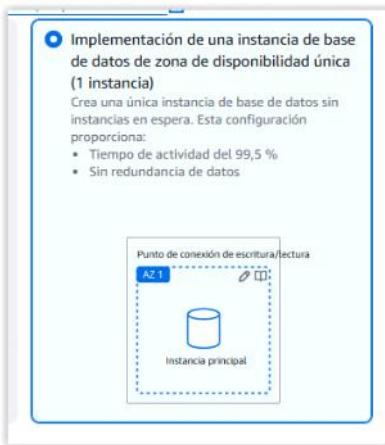
Servidor: **awsmysqlPGS**

Usuario: **adminsql**

Password: **Admin123**

The screenshot shows the AWS RDS MySQL creation wizard. At the top, it says "Aurora and RDS Servicio de bases de datos relacionales administrado". Below that, there's a section for "MySQL" with a logo. The main configuration area includes:

- Edición:** Comunidad de MySQL (selected)
- Versión del motor:** MySQL 8.0.41 (selected)
- Ocultar filtros:** Options for showing compatible versions with multi-AZ clusters or optimized writes.
- Plantillas:** Options for Production, Development/Test, or Free Tier.



Configuración

Identificador de instancias de bases de datos [Información](#)

Escriba un nombre para la instancia de base de datos. El nombre debe ser único en relación con todas las instancias de base de datos pertenecientes a su cuenta de AWS en la región de AWS actual.

El identificador de la instancia de base de datos no distingue entre mayúsculas y minúsculas, pero se almacena con todas las letras en minúsculas (como en "mydbinstance").
Restricciones: de 1 a 63 caracteres alfanuméricos o guiones. El primer carácter debe ser una letra. No puede contener dos guiones consecutivos. No puede terminar con un guion.

Nombre de usuario maestro [Información](#)

Escriba un ID de inicio de sesión para el usuario maestro de la instancia de base de datos.

1 a 16 caracteres alfanuméricos. El primer carácter debe ser una letra.

Administración de credenciales

Puede usar AWS Secrets Manager o administrar sus credenciales de usuario maestro.

Administrado en AWS Secrets Manager - **más seguro**
RDS genera una contraseña y la administra durante todo su ciclo de vida mediante AWS Secrets Manager.

Autoadministrado
Cree su propia contraseña o pida a RDS que cree una contraseña para que pueda administrarla.

Generar contraseña automáticamente

Amazon RDS puede generar una contraseña en su nombre, o bien puede especificar su propia contraseña.

Contraseña maestra [Información](#)

Seguridad de la contraseña [Muy débil](#)

Restricciones mínimas: al menos 8 caracteres ASCII imprimibles. No puede contener ninguno de los siguientes símbolos: / " @

Confirmar la contraseña maestra [Información](#)

Clase de instancia de base de datos [Información](#)

▼ Ocultar filtros

Mostrar las clases de instancia que admiten las escrituras optimizadas de Amazon RDS

Información

Las escrituras optimizadas de Amazon RDS mejoran el rendimiento de escritura hasta 2 veces sin costo adicional.

Incluir clases de generación anterior

Clases estándar (incluye clases m)

Clases optimizadas para memoria (incluye clases r y x)

Clases ampliables (incluye clases t)

db.t3.micro

2 vCPUs 1 GiB RAM Red: hasta 2085 Mbps



Almacenamiento

Tipo de almacenamiento [Información](#)

Los volúmenes de almacenamiento SSD de IOPS aprovisionadas (io2) ya están disponibles.

SSD de uso general (gp2)

Rendimiento de referencia determinado por el tamaño del volumen



Almacenamiento asignado [Información](#)

20

GiB

El valor de almacenamiento asignado debe ser de 20 GiB a 6144 GiB

► Configuración de almacenamiento adicional

▼ Configuración de almacenamiento adicional

Escalado automático de almacenamiento [Información](#)

Proporciona compatibilidad con el escalado dinámico para el almacenamiento de la base de datos en función de las necesidades de la aplicación.

Habilitar escalado automático de almacenamiento

Si se habilita esta característica, el almacenamiento podrá aumentar después de que se supere el umbral especificado.



Conectividad [Información](#)

Recurso de computación

Seleccione si desea configurar una conexión a un recurso de computación para esta base de datos. Al establecer una conexión, se cambiará automáticamente la configuración de conectividad para que el recurso de computación se pueda conectar a esta base de datos.

No se conecte a un recurso informático EC2

No configure una conexión a un recurso informático para esta base de datos. Puede configurar manualmente una conexión a un recurso informático más adelante.

Conectarse a un recurso informático de EC2

Configure una conexión a un recurso informático EC2 para esta base de datos.

Tipo de red [Información](#)

Para utilizar el modo de pila doble, asegúrese de asociar un bloque de CIDR IPv6 a una subred en la VPC que especifique.

IPv4

Sus recursos solo pueden comunicarse a través del protocolo de direcciones IPv4.

Modo de pila doble

Sus recursos pueden comunicarse a través de IPv4, IPv6 o ambos.

Acceso público [Información](#)

Sí

RDS asigna una dirección IP pública a la base de datos. Las instancias de Amazon EC2 y otros recursos fuera de la VPC pueden conectarse a la base de datos. Los recursos de la VPC también pueden conectarse a la base de datos. Elija uno o varios grupos de seguridad de VPC que especifiquen qué recursos pueden conectarse a la base de datos.

No

RDS no asigna una dirección IP pública a la base de datos. Solo las instancias de Amazon EC2 y otros recursos dentro de la VPC pueden conectarse a la base de datos. Elija uno o varios grupos de seguridad de VPC que especifiquen qué recursos pueden conectarse a la base de datos.

Grupo de seguridad de VPC (firewall) [Información](#)

Grupo de seguridad de VPC (firewall) [Información](#)

Elija uno o varios grupos de seguridad de VPC para permitir el acceso a su base de datos. Asegúrese de que las reglas del grupo de seguridad permitan el tráfico entrante adecuado.

Elegir existente

Elegir grupos de seguridad de VPC existentes

Crear nuevo

Crear un grupo de seguridad nuevo de VPC

Nuevo nombre del grupo de seguridad de VPC

grupo-mysql-rds

Zona de disponibilidad [Información](#)

Sin preferencia

Proxy de RDS

El proxy de RDS es un proxy de base de datos completamente administrado y de alta disponibilidad que mejora la escalabilidad, la resiliencia y la seguridad de las apli

Creación de un proxy de RDS [Información](#)

RDS crea automáticamente un rol de IAM y un secreto de Secrets Manager para el proxy. El proxy de RDS tiene costos adicionales. Para obtener más información, consulte [Precios del proxy de Amazon RDS](#).

Entidad de certificación - opcional [Información](#)

Al utilizar un certificado de servidor, se obtiene una capa adicional de seguridad al validar que la conexión se establece con una base de datos de Amazon. Para ello, se comprueba el certificado de servidor que se instala automáticamente en todas las bases de datos aprovisionadas.

rds-ca-rsa2048-g1 (predeterminado)

Vencimiento: May 26, 2061

Si no selecciona una entidad emisora de certificación, RDS elegirá una por usted.

▼ Configuración adicional

Puerto de la base de datos [Información](#)

Puerto TCP/IP que la base de datos usará para las conexiones de las aplicaciones.

3306

Autenticación de bases de datos

Opciones de autenticación de bases de datos [Información](#)

Autenticación con contraseña

Se autentica con las contraseñas de las bases de datos.

Autenticación de bases de datos con contraseña e IAM

Se autentica con las credenciales de usuario y la contraseña de las bases de datos a través de usuarios y roles de AWS IAM.

Autenticación Kerberos y con contraseña

Elija un directorio en el que desee permitir que los usuarios autorizados se autentiquen en esta instancia de base de datos a través de la autenticación Kerberos.

▼ Configuración adicional

Opciones de base de datos, cifrado activado, copia de seguridad activada, retroceder desactivado, mantener protección desactivado.

Opciones de base de datos

Nombre de base de datos inicial [Información](#)

Si no especifica un nombre de base de datos, Amazon RDS no crea una base de datos.

Grupo de parámetros de base de datos [Información](#)

▼

Grupo de opciones [Información](#)

▼

Copia de seguridad

Habilitar las copias de seguridad automatizadas.

Crea una instantánea de un momento dado de su base de datos

Cifrado

Habilitar el cifrado

Elija cifrar la instancia proporcionada. Los ID y alias de la clave maestra aparecen en la lista después de haberse creado mediante la consola de AWS Key Management Service. [Información](#)

Mantenimiento

Actualización automática de la versión secundaria [Información](#)

Habilitar actualización automática de versiones secundarias

La habilitación de la actualización automática de versión secundaria se actualizará automáticamente a nuevas versiones secundarias a medida que se vayan publicando. Las actualizaciones automáticas se realizan durante el periodo de mantenimiento de la base de datos.

Periodo de mantenimiento [Información](#)

Una vez creada la base de datos, nos genera un enlace para acceder

Conectividad y seguridad

Punto de enlace y puerto

Punto de enlace
awsmysqlpgs.c23akq60glxj.us-east-1.rds.amazonaws.com

Puerto
3306

Redes

Zona de disponibilidad
us-east-1c

VPC
vpc-066662b1b43cc56c0

Grupo de subredes
default-vpc-066662b1b43cc56c0

Subredes
subnet-003e77056d7264783
subnet-000d1117215ce99b3
subnet-05ecbc501e89a5eb5
subnet-038db9161528537a4
subnet-049540887ba2b91b6
subnet-0159958d113b9496e

Seguridad

Grupos de seguridad de la VPC
grupo-mysql-rds (sg-0430275b821c3557f)

Activo

Accesible públicamente

Sí

Entidad de certificación

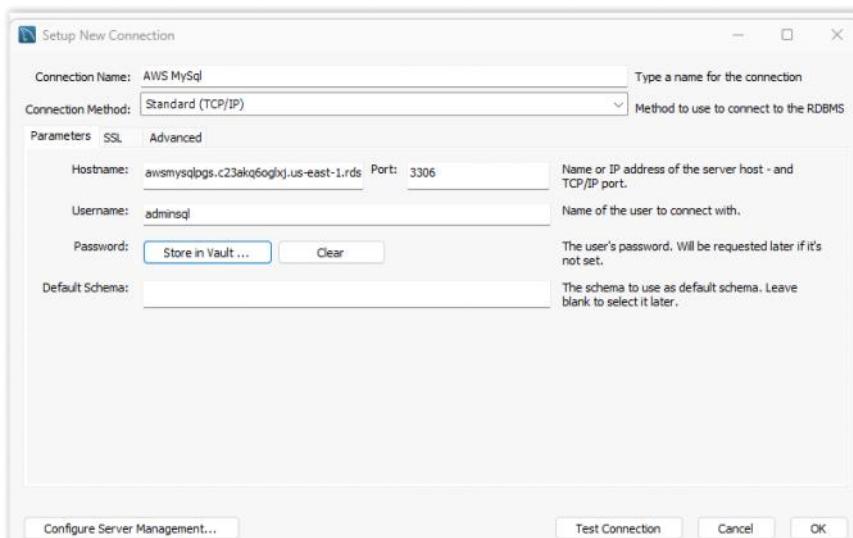
[Información](#)
rds-ca-rsa2048-g1

Fecha de la entidad de certificación

May 26, 2061, 01:34 (UTC+02:00)

Fecha de expiración del certificado de instancia de base

Abrimos **Workbench**



A diferencia de Azure, podremos comprobar que conectamos sin problemas.

Vamos a crear un Api para mostrar Personajes.

Creamos una nueva base de datos llamada **televisión** y copiamos el Script de personajes

Creamos una nueva aplicación llamada **ApiPersonajesAWS**

The screenshot shows the NuGet Package Manager interface with three installed packages listed:

- Microsoft.EntityFrameworkCore** by aspnet, dotnetframework, EntityF 9.0.4
Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations. EF Core works...
Version: 9.0.4
- Microsoft.EntityFrameworkCore.Tools** by aspnet, dotnetframework 9.0.4
Entity Framework Core Tools for the NuGet Package Manager Console in Visual Studio.
Version: 9.0.4
- MySql.EntityFrameworkCore** by MySQL, 8.99M downloads 9.0.3
MySql.EntityFrameworkCore adds support for Microsoft Entity Framework Core.
Version: 9.0.3

Sobre **Models** creamos una nueva clase llamada **Personaje**

PERSONAJE

```
[Table("PERSONAJES")]
0 references
public class Personaje
{
    [Key]
    [Column("IDPERSONAJE")]
    0 references
    public int IdPersonaje { get; set; }
    [Column("PERSONAJE")]
    0 references
    public string Nombre { get; set; }
    [Column("IMAGEN")]
    0 references
    public string Imagen { get; set; }
}
```

Sobre **Data** creamos una nueva clase llamada **PersonajesContext**

PERSONAJESCONTEXT

```
2 references
public class PersonajesContext: DbContext
{
    0 references
    public PersonajesContext(DbContextOptions<PersonajesContext> options)
        : base(options) { }
    0 references
    public DbSet<Personaje> Personajes { get; set; }
}
```

Sobre **Repositories** creamos una nueva clase llamada **RepositoryPersonajes**

```
public class RepositoryPersonajes
{
    private PersonajesContext context;
    public RepositoryPersonajes(PersonajesContext context)
    {
        this.context = context;
    }
    public async Task<List<Personaje>> GetPersonajesAsync()
    {
        return await this.context.Personajes.ToListAsync();
    }
    private async Task<int> GetMaxIdPersonajeAsync()
    {
        return await this.context.Personajes.MaxAsync(x => x.IdPersonaje) + 1;
    }
    public async Task CreatePersonajeAsync(string nombre, string imagen)
    {
        Personaje p = new Personaje();
        p.IdPersonaje = await this.GetMaxIdPersonajeAsync();
        p.Nombre = nombre;
        p.Imagen = imagen;
        await this.context.Personajes.AddAsync(p);
        await this.context.SaveChangesAsync();
    }
}
```

Sobre **Controllers** creamos un nuevo controlador llamado **PersonajesController**

PERSONAJESCONTROLLER

```
[Route("api/[controller]")]
[ApiController]
public class PersonajesController : ControllerBase
{
    private RepositoryPersonajes repo;

    public PersonajesController(RepositoryPersonajes repo)
    {
        this.repo = repo;
    }

    [HttpGet]
    public async Task<ActionResult<List<Personaje>>> GetPersonajes()
    {
        return await this.repo.GetPersonajesAsync();
    }

    [HttpPost]
    public async Task<ActionResult> Create(Personaje personaje)
    {
        await this.repo.CreatePersonajeAsync(personaje.Nombre, personaje.Image
n);
        return Ok();
    }
}
```

Cadena de conexión dentro de appsettings.json

APPSETTINGS.JSON

```
{
    "Logging": {
        "LogLevel": {
            "Default": "Information",
            "Microsoft.AspNetCore": "Warning"
        }
    },
    "AllowedHosts": "*",
    "ConnectionStrings": {
        " MySql": "server=awsmysqlpgs.c23akq6oglxj.us-east-1.rds.amazonaws.com;port=3306;user id=;password="
    }
}
```

Agregamos el siguiente Nuget



PROGRAM

```
using ApiPersonajesAWS.Data;
using ApiPersonajesAWS.Repositories;
using Microsoft.EntityFrameworkCore;
using Scalar.AspNetCore;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
// Learn more about configuring OpenAPI at https://aka.ms/aspnet/openapi
string connectionString = builder.Configuration.GetConnectionString(" MySql");
builder.Services.AddTransient<RepositoryPersonajes>();
builder.Services.AddDbContext<PersonajesContext>(
    options => options.UseMySQL(connectionString));
builder.Services.AddOpenApi();
builder.Services.AddControllers();

var app = builder.Build();

// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
}
app.MapScalarApiReference();
app.MapOpenApi();
app.UseHttpsRedirection();
app.MapControllers();
app.Run();
```

Configuramos Properties/Launchsettings.json

```

"http": {
  "commandName": "Project",
  "dotnetRunMessages": true,
  "launchBrowser": true,
  "launchUrl": "scalar/v1",
  "applicationUrl": "http://localhost:5041",
  "environmentVariables": {
    "ASPNETCORE_ENVIRONMENT": "Development"
  }
},
"https": {
  "commandName": "Project",
  "dotnetRunMessages": true,
  "launchBrowser": true,
  "launchUrl": "scalar/v1",
  "applicationUrl": "https://localhost:7157;http://localhost:5041",
  "environmentVariables": {
    "ASPNETCORE_ENVIRONMENT": "Development"
  }
}

```

Y veremos nuestra App trabajando correctamente

The screenshot shows a Swagger UI interface. On the left, there's a sidebar with a search bar and a 'Personajes' section containing two endpoints: `/api/Personajes` (GET) and `/api/Personajes` (POST). The main area is titled `/api/Personajes`. Under the title, it says 'Responses' and shows a successful response (200 OK). To the right, there's a 'Test Request' button and a preview of the JSON response body:

```

200
[
  {
    "idPersonaje": 1,
    "nombre": "string",
    "imagen": "string"
  }
]
OK

```

Una vez que tenemos nuestra aplicación funcional, es el momento de montar la App dentro de una máquina EC2 y ver qué más tendríamos que hacer (o no...)

Vamos a utilizar **BASH SCRIPT** que nos permite, mediante un Script poder generar la máquina. E incluir los comandos y librerías mientras nos genera la máquina.

Esta funcionalidad es muy sencilla, simplemente incluimos en la creación de la máquina Los comandos que deseemos para desplegar

Tenemos que modificar el Script para poner la versión Net Core 9.0

Voy a utilizar esta vez AMI 2

Creamos una nueva máquina llamada **ec2-api-personajes-rds**

The screenshot shows the AWS Lambda console. At the top, there are tabs for 'Recientes' and 'Inicio rápido'. Below these are several icons for different AMI distributions: Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, and a 'Del' icon. In the 'Inicio rápido' tab, there is a search bar labeled 'Buscar más AMI' and a note 'Inclusión de AMI de AWS, Marketplace y la comunidad'. Below this, there is a section titled 'Imágenes de máquina de Amazon (AMI)' with a list of AMI options. One item is highlighted: 'Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type'. This item has a note 'Apto para la capa gratuita' (Free Tier) and a dropdown arrow indicating more details.

Firewall (grupos de seguridad) | Información
Un grupo de seguridad es un conjunto de reglas de firewall que controlan el tráfico de la instancia. Agregue reglas para permitir que un tráfico específico llegue a la instancia.

Crear grupo de seguridad Seleccionar un grupo de seguridad existente

Nombre del grupo de seguridad - **obligatorio**
grupo-seguridad-api-personajes

Este grupo de seguridad se agregará a todas las interfaces de red. El nombre no se puede editar después de crear el grupo de seguridad. La longitud máxima es de 255 caracteres. Caracteres válidos: a-z, A-Z, 0-9, espacios y _-:/()#,@[]+=&{!\$*

Descripción - obligatorio | Información
launch-wizard-2 created 2025-05-13T08:56:09.280Z

Reglas de grupos de seguridad de entrada

▼ Regla del grupo de seguridad 1 (TCP, 22, 0.0.0.0/0) **Eliminar**

▼ Regla del grupo de seguridad 2 (TCP, 80, 0.0.0.0/0) **Eliminar**

Tipo Información	Protocolo Información	Intervalo de puertos Información
HTTP	TCP	80
Tipo de origen Información	Origen Información	Descripción - opcional Información
Personalizada	<input type="text"/> Agregue CIDR, lista de prefijos o g...	por ejemplo, SSH para Admin Desktop
	<input type="text"/> 0.0.0.0/0	X

▼ Regla del grupo de seguridad 3 (TCP, 443, 0.0.0.0/0) **Eliminar**

Tipo Información	Protocolo Información	Intervalo de puertos Información
HTTPS	TCP	443
Tipo de origen Información	Origen Información	Descripción - opcional Información
Personalizada	<input type="text"/> Agregue CIDR, lista de prefijos o g...	por ejemplo, SSH para Admin Desktop
	<input type="text"/> 0.0.0.0/0	X

Metadatos accesibles | Información
Habilitado

Punto de enlace IPv6 de metadatos | Información
Seleccionar

Versión de metadatos | Información
V1 y V2 (token opcional)

⚠ EC2 recomienda usar la versión 2 de los metadatos, a menos que se requiera explícitamente la versión 1 de los metadatos.

Límite de saltos de respuesta de metadatos | Información
1

Permitir etiquetas en metadatos | Información
Habilitar

Datos de usuario - opcional | Información
Cargue un archivo con los datos de usuario o escribalo en el campo.

Elegir archivo

```
#!/bin/bash
sudo su
yum install httpd -y
yum install git -y
wget https://dot.net/v1/dotnet-install.sh
chmod +x dotnet-install.sh
./dotnet-install.sh --channel 9.0 --quality preview --install-dir
/usr/share/dotnet
sudo ln -s /usr/share/dotnet/dotnet /usr/bin/dotnet
curl install_dotnet.sh
```

Subimos nuestra Api a Github en público

sudo su

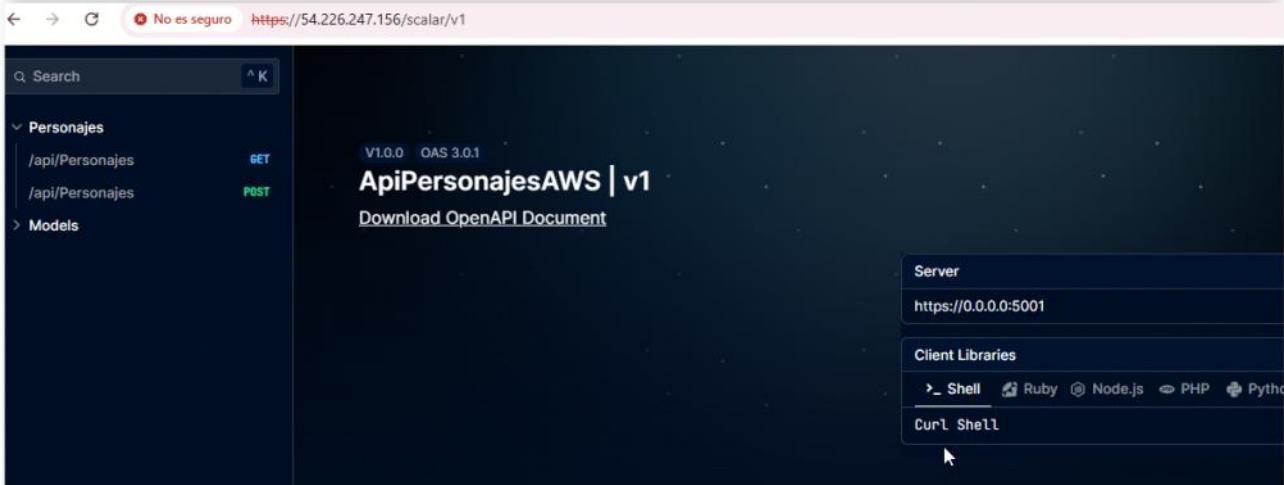
cd /etc/pki/tls/certs

```

sudo ./make-dummy-cert localhost.crt
Creamos una nueva carpeta para nuestro Net Core
cd /home/ec2-user
mkdir core
cd core
git clone URL.git
dotnet restore
dotnet build

Lanzamos nuestra App en HTTPS
dotnet run --urls "https://0.0.0.0:5001"
Y probamos nuestra App en producción

```



Debemos abrir el grupo de seguridad de RDS para que acepte las conexiones remotas de Cualquiera.
Por defecto, genera una IP para acceder desde el equipo donde hemos creado la base de datos.

ID de la regla del grupo de seguridad	Tipo	Protocolo	Intervalo de puertos	Origen	Descripción
sgr-0674dc1aa1c483aaf	MySQL/Aurora	TCP	3306	104.28.192.82/32	-

Editar reglas de entrada Información

Las reglas de entrada controlan el tráfico entrante que puede llegar a la instancia.

Reglas de entrada	Información
ID de la regla del grupo de seguridad	Información
Protocolo	Información
Intervalo de puertos	Información
Origen	Información
Descripción: opcional	

sg-0674dc1aa1c483aaf MySQL/Aurora TCP 3306 Any... 0.0.0.0/0 0.0.0.0/0 X

Agregar regla

Advertencia: Las reglas cuyo origen es 0.0.0.0/0 o ::/0 permiten a todas las direcciones IP acceder a la instancia. Recomendamos configurar reglas de grupo de seguridad para permitir el acceso únicamente desde direcciones IP conocidas.

Cancelar **Previsualizar los cambios** **Guardar reglas**

El siguiente paso que vamos a realizar es crear una App MVC Net Core para Consumir los personajes.

Creamos una nueva aplicación llamada **MvcNetCorePersonajesAWS**

Primero vamos a consumir el servicio mediante código Cliente JS.

INDEX.CSHTML

```

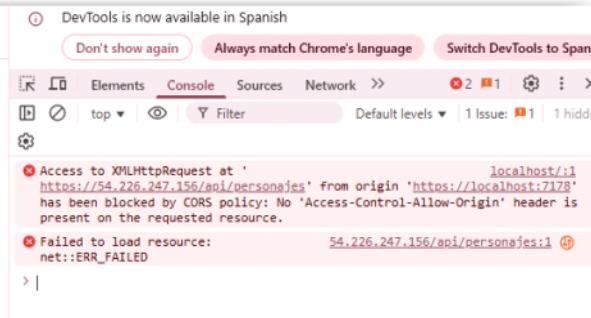
@{
    ViewData["Title"] = "Home Page";
}
@section Scripts {
    <script>
        $(document).ready(function() {
            $("#botonload").click(function() {
                let url = "https://54.226.247.156/api/personajes";
                $.get(url, function(data){
                    var html = "";
                    $.each(data, function(index, personaje){
                        html += "<li class='list-group-item'>";
                        html += personaje.nombre;
                        html += "<img src='" + personaje.imagen + "' style='"
                            + "width: 100px; height: 120px;'/>";
                        html += "</li>";
                    })
                    $("#capapersonajes").html(html);
                })
            })
        })
    </script>
}
<div class="text-center">
    <h1 class="display-4">Personajes AWS EC2</h1>
    <button id="botonload" class="btn btn-info">
        Load Api Personajes
    </button>
    <ul class="list-group" id="capapersonajes">
    </ul>
</div>

```

MvcNetCorePersonajesAWS Home Privacy

Personajes AWS EC2

[Load Api Personajes](#)



Tenemos dos formas de habilitar CORS:

- 1) En el servidor de acceso (Apache Httpd) mediante ficheros de configuración
- 2) Mediante código C# en el que indicaremos que habilite CORS para las peticiones

Abrimos el proyecto Api.

PROGRAM

```

using ApiPersonajesAWS.Data;
using ApiPersonajesAWS.Repositories;
using Microsoft.EntityFrameworkCore;
using Scalar.AspNetCore;
using System;

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddCors(p => p.AddPolicy("corsenabled", options =>
{
    options.WithOrigins("*").AllowAnyMethod().AllowAnyHeader();
}));

// Add services to the container.
// Learn more about configuring OpenAPI at https://aka.ms/aspnet/openapi
string connectionString = builder.Configuration.GetConnectionString("MySQL");
builder.Services.AddTransient<RepositoryPersonajes>();
builder.Services.AddDbContext<PersonajesContext>(
    (options => options.UseMySQL(connectionString)));
builder.Services.AddOpenApi();
builder.Services.AddControllers();

var app = builder.Build();

// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
}
app.MapScalarApiReference(opt =>
{
    opt.Title = "Scalar Personajes";
    opt.Theme = ScalarTheme.BluePlanet;
});
app.UseCors("corsenabled");
app.MapOpenApi();
app.UseHttpsRedirection();
app.MapControllers();
app.Run();

```

```

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddCors(p => p.AddPolicy("corsenabled", options =>
{
    options.WithOrigins("*").AllowAnyMethod().AllowAnyHeader();
}));

```

```
app.UseCors("corsenabled");
app.MapOpenApi();
app.UseHttpsRedirection();
app.MapControllers();
app.Run();
```

Volvemos a nuestra máquina del API actualizamos nuestra app:

`git pull`

Borramos la app y creamos la carpeta de nuevo

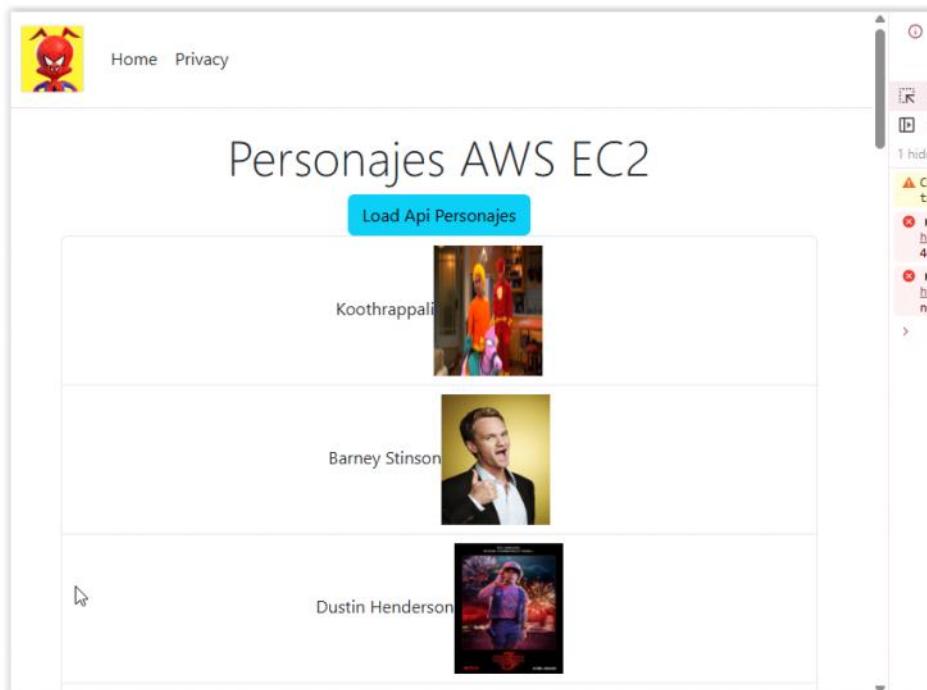
`rm -rf Api...`

```
[ec2-user@ip-172-31-20-152 ec2-user]# cd core
[ec2-user@ip-172-31-20-152 core]# rm -rf ApiPersonajesAWS2025/
[ec2-user@ip-172-31-20-152 core]# git clone https://github.com/serraguti/ApiPersonajesAWS2025.git
Cloning into 'ApiPersonajesAWS2025'...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (20/20), done.
remote: Total 27 (delta 4), reused 27 (delta 4), pack-reused 0 (from 0)
Receiving objects: 100% (27/27), 7.86 KiB | 7.86 MiB/s, done.
Resolving deltas: 100% (4/4), done.
[ec2-user@ip-172-31-20-152 core]# cd ApiPersonajesAWS2025/
[ec2-user@ip-172-31-20-152 ApiPersonajesAWS2025]# cd ApiPersonajesAWS
[ec2-user@ip-172-31-20-152 ApiPersonajesAWS]#
```

Ponemos en juego nuestra aplicación API con los procesos

`nohup dotnet run --urls "https://0.0.0.0:5001" > /dev/null 2>&1 &`

Y ya tendremos todo activo con cliente



A continuación, vamos a consumir el servicio mediante código Servidor.

Sobre el proyecto MVC, instalamos los siguientes Nuget

 **Microsoft.AspNet.WebApi.Client** by aspnet, Microsoft, 717M downloads
This package adds support for formatting and content negotiation to System.Net.Http.

 **Newtonsoft.Json** by dotnetfoundation, jamesnk, newtonsoft, 6.2B downloads
Json.NET is a popular high-performance JSON framework for .NET

Sobre **Models** creamos una nueva clase llamada **Personaje**

PERSONAJE

```

0 references
public class Personaje
{
    0 references
    public int IdPersonaje { get; set; }
    0 references
    public string Nombre { get; set; }
    0 references
    public string Imagen { get; set; }
}

```

Creamos una carpeta llamada **Services** y una clase llamada **ServiceApiPersonajes**

SERVICEAPIPERSONAJES

```

public class ServiceApiPersonajes
{
    private MediaTypeWithQualityHeaderValue Header;
    private string UrlApi;

    public ServiceApiPersonajes(IConfiguration configuration)
    {
        this.UrlApi = configuration.GetValue<string>
            ("ApiUrls:apiPersonajes");
        this.Header = new MediaTypeWithQualityHeaderValue("application/json");
    }

    public async Task<List<Personaje>> GetPersonajesAsync()
    {
        using (HttpClient client = new HttpClient())
        {
            string request = "api/personajes";
            client.DefaultRequestHeaders.Clear();
            client.DefaultRequestHeaders.Accept.Add(this.Header);
            HttpResponseMessage response = await
                client.GetAsync(this.UrlApi + request);
            if (response.IsSuccessStatusCode)
            {
                List<Personaje> personajes = await
                    response.Content.ReadAsAsync<List<Personaje>>();
                return personajes;
            }
            else
            {
                return null;
            }
        }
    }

    public async Task CreatePersonajeAsync(string nombre, string imagen)
    {
        using (HttpClient client = new HttpClient())
        {
            string request = "api/personajes";
            client.DefaultRequestHeaders.Clear();
            client.DefaultRequestHeaders.Accept.Add(this.Header);
            Personaje personaje = new Personaje
            {
                IdPersonaje = 0, Nombre = nombre, Imagen = imagen
            };
            string json = JsonConvert.SerializeObject(personaje);
            StringContent content = new StringContent
                (json, Encoding.UTF8, "application/json");
            HttpResponseMessage response =
                await client.PostAsync(this.UrlApi + request, content);
        }
    }
}

```

Incluimos la URL para acceder a nuestro Api en la máquina EC2

```

emastore.org/appsettings.json
[{"$id": "1", "Logging": {"LogLevel": {"Default": "Information", "Microsoft.AspNetCore": "Warning"}}, "AllowedHosts": "*", "ApiUrls": {"ApiPersonajes": "https://54.226.247.156/"}}
]

```

Resolvemos las dependencias dentro de Program

```

PROGRAM

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddTransient<ServiceApiPersonajes>();
builder.Services.AddControllersWithViews();

```

Sobre **Controllers** creamos un nuevo controlador llamado **PersonajesController**

PERSONAJESCONTROLLER

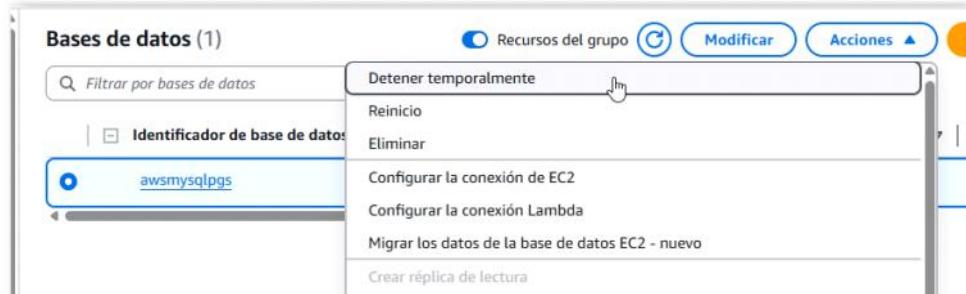
```
public class PersonajesController : Controller
{
    private ServiceApiPersonajes service;
    public PersonajesController(ServiceApiPersonajes service)
    {
        this.service = service;
    }
    public async Task<IActionResult> Index()
    {
        List<Personaje> personajes =
            await this.service.GetPersonajesAsync();
        return View(personajes);
    }
    public IActionResult Create()
    {
        return View();
    }
    [HttpPost]
    public async Task<IActionResult> Create(Personaje personaje)
    {
        await this.service.CreatePersonajeAsync(personaje.Nombre
            , personaje.Imagen);
        return RedirectToAction("Index");
    }
}
```

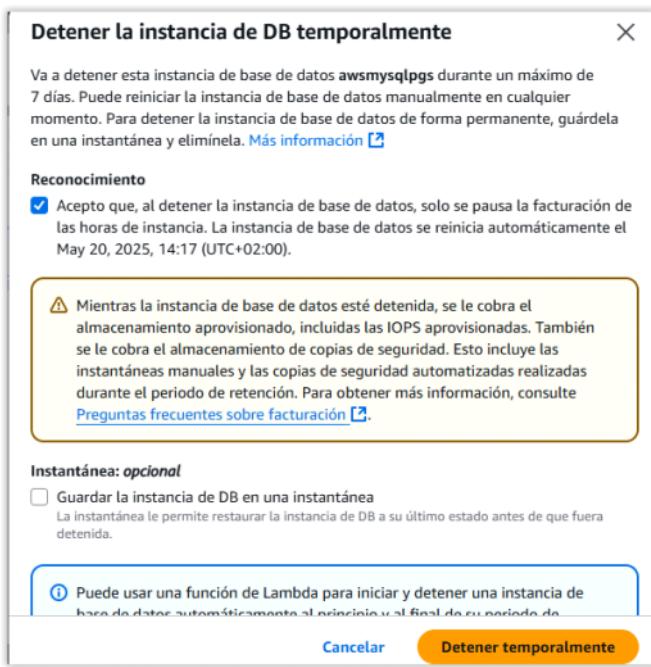
La petición para el servicio con SSL de "mentira":

```
public async Task<List<Personaje>> GetPersonajesAsync()
{
    using (HttpClientHandler handler = new HttpClientHandler())
    {
        handler.ServerCertificateCustomValidationCallback =
            (message, cert, chain, sslPolicies) =>
        {
            return true;
        };
        using (HttpClient client = new HttpClient(handler))
        {
            string request = "api/personajes";
            client.DefaultRequestHeaders.Clear();
            client.DefaultRequestHeaders.Accept.Add(this.Header);
            HttpResponseMessage response = await
                client.GetAsync(this.UrlApi + request);
            if (response.IsSuccessStatusCode)
            {
                List<Personaje> personajes = await
                    response.Content.ReadAsAsync<List<Personaje>>();
                return personajes;
            }
            else
            {
                return null;
            }
        }
    }
}
```

Con este proyecto, subimos a GitHub el MVC en **privado**

La base de datos la detenemos:





La máquina EC2 la detenemos.

Instancias (1/1) [Información](#)

Última actualización Hace less than a minute

Estado de la instancia ▲

Acciones ▾ **Lanzar**

Buscar Instancia por atributo o etiqueta

Name ID de la instancia

ec2-api-perso... i-0adc7c213e7

Detener instancia

Iniciar instancia

Reiniciar instancia

Hibernar instancia

Terminar (eliminar) instancia

Todos los estados

Tipo de instancia t2.micro

Comandos para mañana

dotnet run --urls "<https://0.0.0.0:5001>"

Poner el servicio en los procesos

nohup dotnet run --urls "<https://0.0.0.0:5001>" > /dev/null 2>&1 &

Guardar IP Tables de nuevo

sudo iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 443 -j REDIRECT --to-port 5001

Saber si tenemos IPTABLES

sudo iptables -t nat -v -L -n --line-number

Borrar IP Tables

sudo iptables -t nat --delete PREROUTING 1

Mirar los procesos

ps -ef

Eliminar proceso

kill PID

El siguiente paso es crear una máquina MVC para nuestra aplicación. Llamaremos a la máquina **ec2-personajes-mvc** y ponemos la máquina en el mismo grupo de Seguridad que el Api.

Utilizamos Bash Script.

Abrimos nuestro Github con el proyecto MVC privado

Necesitamos un Token que nos ofrece Github en nuestro proyecto para reconocer que somos nosotros
En los repositorios privados.

Entramos en **Settings** y en **Developer Settings**

Entramos en **Token Classic**

The screenshot shows the GitHub Developer Settings interface. On the left, there's a sidebar with 'GitHub Apps', 'OAuth Apps', 'Personal access tokens' (which is expanded), 'Fine-grained tokens', and 'Tokens (classic)'. The main area is titled 'Personal access tokens (classic)' with a sub-header 'Tokens you have generated that can be used to access the [GitHub API](#)'. It lists a token named 'Token de Viernes — repo' which expired on Sun, Jun 16 2024. There are two buttons: 'Generate new token' (fine-grained, repo-scoped) and 'Generate new token (classic)' (for general use). A note below explains that these tokens function like OAuth access tokens.

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

Token Miercoles

What's this token for?

Expiration

30 days (Jun 13, 2025)

The token will expire on the selected date

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input type="checkbox"/> repo:status	Access commit status
<input type="checkbox"/> repo_deployment	Access deployment status
<input type="checkbox"/> public_repo	Access public repositories
<input type="checkbox"/> repo:invite	Access repository invitations
<input type="checkbox"/> security_events	Read and write security events

Para poder utilizar el Token en nuestra línea de comandos:

git clone <https://TOKEN@github.com/URL.git>

```
[root@ip-172-31-94-54 ec2-user]# cd /etc/pki/tls/certs/personajesAWS.git
[root@ip-172-31-94-54 certs]# ls
ca-bundle.crt      localhost.crt      Makefile
ca-bundle.trust.crt make-dummy-cert  renew-dummy-cert
[root@ip-172-31-94-54 certs]# sudo ./make-dummy-cert localhost.crt
[root@ip-172-31-94-54 certs]# cd /home/ec2-user
[root@ip-172-31-94-54 ec2-user]# mkdir mvc > /dev/null
[root@ip-172-31-94-54 ec2-user]# cd mvc
[root@ip-172-31-94-54 mvc]# dotnet run --urls "https://0.0.0.0:5001" > /dev/null
```

git clone <https://TOKEN@github.com/serraguti/MvcNetCorePersonajesAWS.git>

Entramos dentro de nuestro proyecto y...

dotnet restore

dotnet build

dotnet run --urls "https://0.0.0.0:5001"

El siguiente paso, a sorprenderme...

Necesito un método para poder modificar los personajes en el Api

Necesito implementarlo dentro del MVC

Una vez que tengamos todo, quiero verlo en producción.

Vamos a solucionarlo con Docker, Portainer, Duckdns y un **Ami 2023**

Creamos una nueva máquina **ec2-mvc-2023-rds**

Recientes

Inicio rápido

Amazon Linux

macOS

Ubuntu

Windows

Red Hat

SUSE Linux

Del

Buscar más AMI

AMI de Amazon Linux 2023

ami-0953476d60561c955 (64 bits (x86), uefi-preferred) / ami-05a3e0187917e3e24 (64 bits (Arm), uefi)

Virtualización: hvm Activado para ENA: true Tipo de dispositivo raíz: ebs

Apto para la capa gratuita

Descripción

Vamos a crear un grupo de seguridad y abrimos los siguientes puertos:

- HTTP
- HTTPS
- TCP Personalizado: 9000 (**Portainer**)
- TCP Personalizado: 81 (**Nginx**)
- TCP Personalizado: 5000 (**Net Core**)

Utilizamos el BASH nuevo que os he dejado...

Datos de usuario - opcional | Información

Cargue un archivo con los datos de usuario o escribalo en el campo.

Elegir archivo

```
#!/bin/bash
sudo su
sudo wget -O /etc/yum.repos.d/microsoft-prod.repo
https://packages.microsoft.com/config/fedora/38/prod.repo
dnf install dotnet-sdk-9.0 -y
dnf install git -y
sudo dnf install docker -y
sudo service docker start
sudo chkconfig docker on
sudo curl -L
https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
```

Instalamos **Portainer**

`docker pull portainer/portainer-ce:latest`

Desplegamos Portainer

`docker run -d -p 9000:9000 --restart always -v /var/run/docker.sock:/var/run/docker.sock portainer/portainer-ce:latest`

Entramos en Portainer y ponemos como password **Portainer12345** con el user **admin**

No es seguro 3.208.19.64:9000/#!/init/admin

New Portainer installation

Please create the initial administrator user.

Username: admin

Password: [REDACTED]

Confirm password: [REDACTED] ✓

⚠ The password must be at least 12 characters long. ✓

Create user

The screenshot shows the Portainer.io interface. On the left is a dark sidebar with navigation links: Home, local (selected), Dashboard, Templates, Stacks, Containers, Images, Networks, Volumes, Events, Host, and Administration. The main content area has a light blue header "Environments" and a "Home" button. Below it is a section titled "Latest News From Portainer" with a message about Portainer 2.27.0 LTS. The "Environments" section has a search bar and filters for Platform, Connection, Status, Tags, and Groups. It shows one environment named "local" which is "Up" (green status). The "local" environment summary includes: Group: Unassigned, No tags, Local, 0 stacks, 1 container (status: 1 up, 0 down, 0 green, 0 red), 1 volume, 1 image, and 995.5 MB RAM.



Copiamos el `docker-compose.yml` de nginx

DOCKER-COMPOSE

```
services:
  nginx:
    image: jc21/nginx-proxy-manager:latest
    restart: always
    ports:
      # These ports are in format <host-port>:<container-port>
      - 80:80 # Public HTTP Port
      - 443:443 # Public HTTPS Port
      - 81:81 # Admin Web Port
      # Add any other Stream port you want to expose
      # - '21:21' # FTP
    environment:
      # Postgres parameters:
      DB_POSTGRES_HOST: dbpostgres
      DB_POSTGRES_PORT: "5432"
      DB_POSTGRES_USER: npm
      DB_POSTGRES_PASSWORD: password
      DB_POSTGRES_NAME: npm
      # Uncomment this if IPv6 is not enabled on your host
      DISABLE_IPV6: "true"
    volumes:
      - ./data:/data
      - ./letsencrypt:/etc/letsencrypt
    depends_on:
      - dbpostgres
  dbpostgres:
    image: postgres:latest
    restart: always
    environment:
      POSTGRES_USER: npm
      POSTGRES_PASSWORD: password
      POSTGRES_DB: npm
    volumes:
      - ./postgres:/var/lib/postgresql/data
  networks: {}
```

Name: nginx

This stack will be deployed using `docker compose`.

Build method:

- Web editor** (selected)
- Upload**
- Repository**

Web editor

You can get more information about Compose file format in the [official documentation](#).

① Define or paste the content of your docker compose file here

```

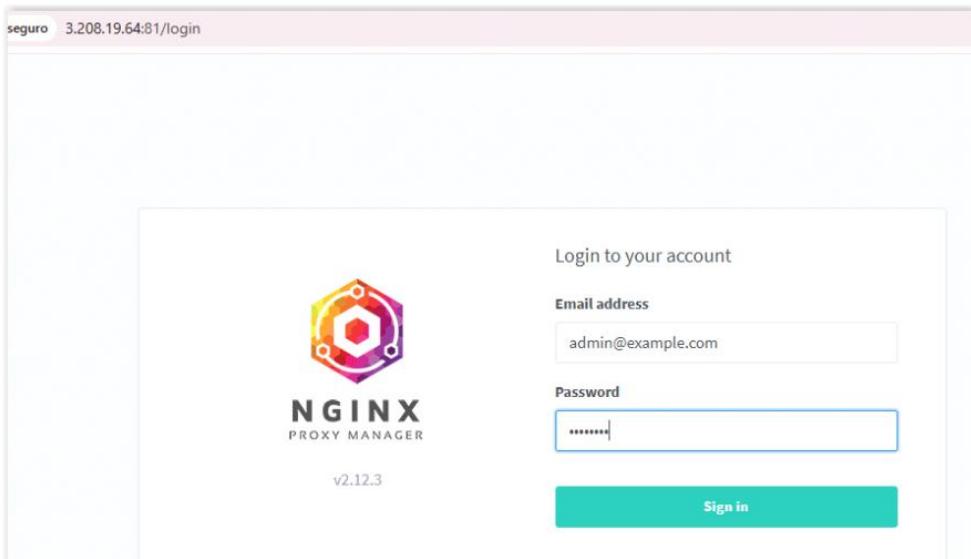
1 services:
2   nginx:
3     image: jc21/nginx-proxy-manager:latest
4     restart: always
5     ports:
6       # These ports are in format <host-port>:<container-port>
7       - 80:80 # Public HTTP Port
8       - 443:443 # Public HTTPS Port
9       - 81:81 # Admin Web Port

```

Hemos terminado con Portainer. Ahora entramos en NGINX con el puerto 81

Email: admin@example.com

Password: **changeme**



Vamos a poner el mismo Password: **Portainer12345**

Debemos poner el correo de **DUCKDNS**

Edit User

Full Name *	Nickname *
Administrator	Admin
Email *	
<input type="text" value="pacoserranox@gmail.com"/>	
<input type="button" value="Cancel"/> <input type="button" value="Save"/>	

Entramos en **SSL Certificates**

Abrimos la página de **duckdns.org**

<https://www.duckdns.org/>

Creamos un dominio cada uno....

Debemos poner la IP de la máquina donde deseemos trabajar. Nuestro MVC, IP pública.

domains	5/5	http://	sub domain	duckdns.org	add domain
domain	current ip	ipv6	change	status	actions
puente-zanahoria	<input type="text" value="3.208.19.64"/>	update ip	<input type="text" value="ipv6 address"/>	update ipv6	0 seconds

El siguiente paso es configurar y solicitar un certificado SSL en nuestra máquina mediante NGINX.

Add Let's Encrypt Certificate

Domain Names *

⚠ These domains must be already configured to point to this installation

Email Address for Let's Encrypt *

Use a DNS Challenge

⚠ This section requires some knowledge about Certbot and its DNS plugins. Please consult the respective plugins documentation.

DNS Provider *

Después de un tiempo, debería crearnos un certificado

SSL Certificates			
NAME	CERTIFICATE PROVIDER	EXPIRES	STATUS
puente-zanahoria.duckdns.org Created: 14th May 2025	Let's Encrypt - DuckDNS	12th August 2025, 11:56 am	● Inactive

El siguiente paso es crear un HOST que utilizará nuestro duckdns.

Duckdns nos proporciona una dirección URL que contiene un servidor HTTPS válido.

Dicha URL debe direccionar internamente dentro de nuestra máquina.

Nosotros, en la máquina MVC vamos a lanzar la aplicación mediante **http** y al puerto **5000**

Proxy Hosts			
NAME	CERTIFICATE PROVIDER	EXPIRES	STATUS
puente-zanahoria.duckdns.org Created: 14th May 2025	Let's Encrypt - DuckDNS	12th August 2025, 11:56 am	● Inactive

New Proxy Host

[Details](#) [Custom locations](#) [SSL](#) [Advanced](#)

Domain Names *

Scheme * **Forward Hostname / IP *** **Forward Port ***

http	3.208.19.64	5000
------	-------------	------

Cache Assets Block Common Exploits

Websockets Support

Access List

Publicly Accessible

[Cancel](#) [Save](#)

New Proxy Host

[Details](#) [Custom locations](#) [SSL](#) [Advanced](#)

SSL Certificate

None

None
This host will not use HTTPS

Request a new SSL Certificate
with Let's Encrypt

puente-zanahoria.duckdns.org
Let's Encrypt - Expires: 12th August 2025, 11:56 am

[Cancel](#) [Save](#)

New Proxy Host

[Details](#) [Custom locations](#) [SSL](#) [Advanced](#)

SSL Certificate

puente-zanahoria.duckdns.org

Force SSL HTTP/2 Support

HSTS Enabled ? HSTS Subdomains

[Cancel](#) [Save](#)

Ya tendremos nuestra IP HTTP hacia el dominio HTTPS.

Para probarlo, necesitamos el servicio MVC en la máquina.

Creamos un directorio, y ejecutamos **dotnet restore** y **build**

Lanzamos nuestro servicio con **HTTP** y puerto **5000**

dotnet run --urls "http://0.0.0.0:5000"

```
Build succeeded with 5 warning(s) in 11.1s,000
[root@ip-172-31-24-236 MvcNetCorePersonajesAWS]# dotnet run --urls "http://0.0.0.0:5000"
Using launch settings from /home/ec2-user/core/MvcNetCorePersonajesAWS/MvcNetCorePersonajesAWS/Properties/launchSettings.json...
Building...
warn: Microsoft.AspNetCore.DataProtection.KeyManagement.XmlKeyManager[35]
      No XML encryptor configured. Key {93388374-336c-417b-af56-1873085d7b5a} may be persisted to storage in an unencrypted form.
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://0.0.0.0:5000
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: /home/ec2-user/core/MvcNetCorePersonajesAWS/MvcNetCorePersonajesAWS
```

Probamos a conectar con la IP pública y el puerto **5000**

Y lo tenemos!!!!

The screenshot shows a web browser window with the following details:

- URL:** puente-zanahoria.duckdns.org/Personajes
- Page Title:** Index
- Content:** A table with two columns: IdPersonaje and Nombre. The table has one row with values 1 and Koothrappali.
- Navigation:** Links for Home, Personajes AWS, and Privacy are visible in the header.
- Icon:** A small superhero icon is present in the header area.

AWS ELASTIC BEANSTALK

Lunes, 19 de mayo de 2025 9:15

Es lo más parecido que tenemos a un App Service de Azure.

En realidad, es una trampa, tenemos que abrir bien los ojos y escuchar a **Paco**.

En realidad, este maravilloso App Service genera una serie de recursos para montarse, Es decir, al final, nos genera una máquina EC2, nos podría llegar a generar un RDS y Nosotros, si no nos fijamos, tan felices.

Tampoco tendremos control sobre la máquina que nos genera.

Genera los siguientes recursos:

- 1) **ELB**: Elastic Load Balancer: Sirve para crear URLs de tipo HTTPS para nuestros Servidores. Al crear una máquina, además de generar una IP pública, nos genera Una dirección HTTPS. Dicha dirección no está activa si no utilizamos ELB. No solo sería con esto, además podría generar un Route S3, que firma un certificado HTTPS.
- 2) **RDS**: Podríamos generar a la vez una base de datos.
- 3) **EC2**: Genera una máquina virtual asociada al Elastic Beanstalk.

Un ELB está formado por lo siguiente:

- 1) Aplicación para desplegar
- 2) Entorno de despliegue de la aplicación /Deployment
- 3) Servicios asociados: ELB, RDS, EC2...

Vamos a visualizar este nuevo servicio de forma "manual" desplegando una simple Página HTML, una imagen y en un servidor PHP.

Necesitamos una página HTML y una imagen e incluir todo lo necesario dentro de un **ZIP**.

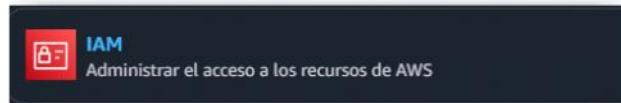
Dicho contenido podemos implementarlo de dos formas:

- 1) Mediante un **S3** y lo que se llama **Artifact** que es el Assembly del despliegue de una Aplicación.
- 2) Subiendo manual los elementos del **Artifact** a nuestro servicio.

Nombre	Fecha de modificación	Tipo	Tamaño
ELB.zip	19/05/2025 9:33	WinRAR ZIP archive	10 KE
fry.png	03/12/2024 11:08	Archivo PNG	8 KE
test.css	17/01/2025 14:05	CSSfile	1 KE
test.html	19/05/2025 9:30	Microsoft Edge H...	1 KE
test1.css	20/01/2025 9:55	CSSfile	3 KE

Necesitamos un **Role de IAM** con permisos para poder desplegar elementos dentro de **ELB** con **EC2**

Abrimos **IAM** y creamos un Role.



Seleccionar entidad de confianza Información

Tipo de entidad de confianza

Servicio de AWS

Permita que servicios de AWS como EC2, Lambda u otros realicen acciones en esta cuenta.

Identidad web

Permite a las personas federadas por el proveedor de identidad web externo especificado asumir este rol para realizar acciones en esta cuenta.

Política de confianza personalizada

Cree una política de confianza personalizada para permitir que otras personas realicen acciones en esta cuenta.

Cuenta de AWS

Permitir a las entidades de otras cuentas de AWS que le pertenezcan a usted o a un tercero realizar acciones en esta cuenta.

Federación SAML 2.0

Permitir que las personas federadas con SAML 2.0 a partir de un directorio corporativo realicen acciones en esta cuenta.

Caso de uso

Permita que un servicio de AWS, como EC2, Lambda u otros, realicen acciones en esta cuenta.

Servicio o caso de uso

EC2



Elija un caso de uso para el servicio especificado.

Caso de uso

EC2

Allows EC2 instances to call AWS services on your behalf.

EC2 Role for AWS Systems Manager

AWS Elastic Beanstalk Web Tier

AWS Elastic Beanstalk Worker Tier

AWS Elastic Beanstalk Multicontainer Docker

Asignar nombre, revisar y crear

Detalles del rol

Nombre del rol

Ingrese un nombre significativo para identificar a este rol.

ec2-elasticbeanstalk-role

64 Caracteres máximos. Utilice caracteres alfanuméricos y '+ =,_@-~'.

Descripción

Agregue una breve explicación para este rol.

Allows EC2 instances to call AWS services on your behalf.

Necesitamos otro Role con permisos de administración

Seleccionar entidad de confianza Información

Tipo de entidad de confianza

Servicio de AWS

Permita que servicios de AWS como EC2, Lambda u otros realicen acciones en esta cuenta.

Cuenta de AWS

Permitir a las entidades de otras cuentas de AWS que le pertenezcan a usted o a un tercero realizar acciones en esta cuenta.

Identidad web

Permite a las personas federadas por el proveedor de identidad web externo especificado asumir este rol para realizar acciones en esta cuenta.

Política de confianza personalizada

Cree una política de confianza personalizada para permitir que otras personas realicen acciones en esta cuenta.

Caso de uso

Permita que un servicio de AWS, como EC2, Lambda u otros, realicen acciones en esta cuenta.

Servicio o caso de uso

Elastic Beanstalk

Elija un caso de uso para el servicio especificado.

Caso de uso

Elastic Beanstalk - Compute

Allows your environment's EC2 instances to perform operations required for your application.

Elastic Beanstalk - Environment

Allows access to other AWS service resources that are required to create and manage environments.

Agregar permisos Información

Políticas de permisos (2) Información

El tipo de rol que ha seleccionado requiere la siguiente política.

Nombre de la política

▲ | Tipo

 AWSElasticBeanstalkEnhancedHealth

Administrada por AWS

 AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy

Administrada por AWS

Asignar nombre, revisar y crear

Detalles del rol

Nombre del rol

Ingrese un nombre significativo para identificar a este rol.

aws-elasticbeanstalk-service-role

64 Caracteres máximos. Utilice caracteres alfanuméricos y '+ =_,@-_-'.

Descripción

Agregue una breve explicación para este rol.



El siguiente paso es crear el servicio Elastic Beanstalk



Elastic Beanstalk

Ejecute y administre aplicaciones web

Configuración del entorno Información

Nivel de entorno Información

Amazon Elastic Beanstalk tiene dos tipos de niveles de entorno para admitir diferentes tipos de aplicaciones web.

Entorno de servidor web

Ejecute un sitio web, una aplicación web o una API web que atienda solicitudes HTTP. [Más información](#)

Entorno de trabajo

Ejecute una aplicación de proceso de trabajo que procese cargas de trabajo de ejecución prolongada bajo demanda o realice tareas de fondo.

Información de la aplicación Información

Nombre de aplicación

app-service-html-elb

La longitud máxima es de 100 caracteres.

► Etiquetas de aplicación (opcional)

Información del entorno Información

Elija el nombre, el subdominio y la descripción del entorno. No se pueden cambiar más adelante.

Nombre del entorno

App-service-html-elb-env

Debe tener entre 4 y 40 caracteres. El nombre solo puede contener letras, números y guiones. No puede comenzar ni terminar por un guion. Este nombre debe ser único dentro de una región de su cuenta.

Dominio

Deje en blanco el valor autogenerado

.us-east-1.elasticbeanstalk.com

[Verificar disponibilidad](#)

Descripción del entorno

Plataforma Información

Tipo de plataforma

Plataforma administrada

Plataformas publicadas y mantenidas por Amazon Elastic Beanstalk. [Más información](#)

Plataforma personalizada

Plataformas creadas y de su propiedad. Esta opción no está disponible si no tiene plataformas.

Plataforma

PHP

Ramificación de la plataforma

PHP 8.4 running on 64bit Amazon Linux 2023

Versión de la plataforma

4.6.1 (Recommended)

Código de aplicación Información

- Aplicación de ejemplo
 Versión existente
Versiones de la aplicación que ha cargado.
 Cargar el código
Cargue un paquete de código fuente desde su equipo o copie uno desde Amazon S3.

Etiqueta de versión

Nombre único para esta versión del código de la aplicación.

v1

Origen del código fuente. Tamaño máximo de 500 MB

- Archivo local

Cargar aplicación

Nombre del archivo: **ELB.zip**

El archivo debe tener un tamaño máximo de archivo inferior a 500 MB

- URL pública de S3

Valores preestablecidos Información

Comience a partir de un elemento preestablecido que coincida con su caso de uso o elija una configuración personalizada para anular los valores recomendados y utilice los valores predeterminados del servicio.

Elementos preestablecidos de configuración

- Instancia única (compatible con la capa gratuita)
 Instancia única (mediante instancia de spot)
 Alta disponibilidad
 Alta disponibilidad (con instancias de spot y bajo demanda)
 Configuración personalizada

Configuración del acceso al servicio Información

Acceso al servicio

Los roles de IAM, asumidos por Elastic Beanstalk como rol de servicio, y los perfiles de instancia de EC2 permiten a Elastic Beanstalk crear y administrar su entorno. Tanto el rol de IAM como el perfil de instancia deben estar asociados a políticas administradas de IAM que contengan los permisos necesarios.

[Más información](#)

Rol de servicio

Elija un rol de IAM para que Elastic Beanstalk asuma como rol de servicio. El rol de IAM debe tener las políticas administradas de IAM necesarias.

aws-elasticbeanstalk-service-role



[Crear rol](#)

Perfil de instancia de EC2

Elija un perfil de instancia de IAM con políticas administradas que permitan a las instancias de EC2 realizar las operaciones necesarias.

ec2-elasticbeanstalk-role



[Crear rol](#)

Par de claves de EC2: opcional

Seleccione un par de claves de EC2 para iniciar sesión de forma segura en sus instancias de EC2. [Más información](#)

keys-linux-tajamar



[Cancelar](#)

[Ir a revisión](#)

[Anterior](#)

[Siguiente](#)

Tipos de instancia

Agregue tipos de instancias para su entorno con el orden de lanzamiento que prefiera. La preferencia de orden solo se aplica a las instancias spot que utilizan la estrategia de asignación priorizada y optimizada para la capacidad. Le recomendamos que incluya al menos dos tipos de instancias.

1. t3.micro



2. t3.small



[Eliminar](#)

[Agregar tipo de instancia](#)

Configuración de actualizaciones, monitoreo y registros - *opcional* Información

▼ Monitoreo Información

Informes de estado

Los informes de estado mejorados proporcionan monitoreo gratuito de aplicaciones y sistemas operativos en tiempo real de las instancias y otros recursos personalizados. **EnvironmentHealth** se proporciona de forma gratuita con informes de estado mejorados. Se aplican cargos adicionales por cada métrica. Para obtener más información, consulte [Precios de Amazon CloudWatch](#).

Sistema

- Básico 
 Mejorado

Transmisión de secuencias de eventos de estado a CloudWatch Logs

Configure Elastic Beanstalk para transmitir eventos de estado del entorno a CloudWatch Logs. Puede establecer la retención hasta un máximo de diez años.

▼ Actualizaciones administradas de la plataforma Información

Active las actualizaciones administradas de la plataforma para aplicarlas de manera automática durante el periodo de mantenimiento que elija. La aplicación permanece disponible durante el proceso de actualización.

Actualizaciones administradas

- Activado 

Periodo de actualización semanal

Martes a las 22:54 UTC

Actualizar nivel

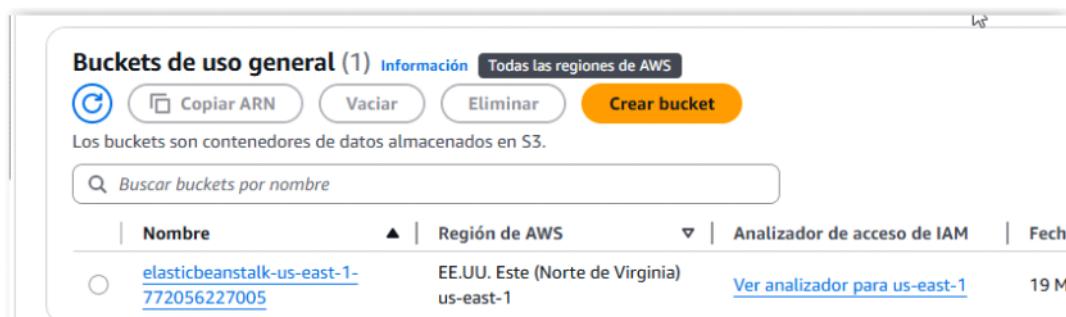
Menor y parche

Sustitución de instancia

Si la habilidad es desactivada, una sustitución de instancia no hace otras actualizaciones disponibles.

Una vez que se ha desplegado el servicio, veremos que también nos ha creado un Servicio S3 con el Artifact en su interior.

Dicho Bucket S3 no lo eliminaremos ya que es imprescindible para utilizar ELB.



Buckets de uso general (1) Información Todas las regiones de AWS

Copiar ARN Vaciar Eliminar Crear bucket

Los buckets son contenedores de datos almacenados en S3.

Buscar buckets por nombre

Nombre	Región de AWS	Analizador de acceso de IAM	Fecha de creación
elasticbeanstalk-us-east-1-772056227005	EE.UU. Este (Norte de Virginia) us-east-1	Ver analizador para us-east-1	19 May 2018

Dentro ELB podemos visualizar lo que nos ha generado.

Una vez que hemos visto las versiones y los elementos, vamos a Terminar nuestro Entorno

El siguiente paso es intentar utilizar una aplicación Net Core

Vamos a ir paso a paso.

Tendremos dos versiones:

- 1) Versión simple para desplegar un Net Core con un título y una triste imagen
- 2) Implementamos la aplicación para consumir datos de un RDS a ver qué sucede.

Vamos a hacerlo todo manual utilizando la Consola de AWS.

Configuración del entorno Información

Nivel de entorno Información

Amazon Elastic Beanstalk tiene dos tipos de niveles de entorno para admitir diferentes tipos de aplicaciones web.

Entorno de servidor web

Ejecute un sitio web, una aplicación web o una API web que atienda solicitudes HTTP. [Más información](#)

Entorno de trabajo

Ejecute una aplicación de proceso de trabajo que procese cargas de trabajo de ejecución prolongada bajo demanda o realice tareas de fondo.

Información de la aplicación Información

Nombre de aplicación

elastic-beanstalk-net-core

La longitud máxima es de 100 caracteres.

Plataforma Información

Tipo de plataforma

Plataforma administrada

Plataformas publicadas y mantenidas por Amazon Elastic Beanstalk. [Más información](#)

Plataforma personalizada

Plataformas creadas y de su propiedad. Esta opción no está disponible si no tiene plataformas.

Plataforma

.NET Core on Linux

Ramificación de la plataforma

.NET 9 running on 64bit Amazon Linux 2023

Versión de la plataforma

3.4.1 (Recommended)

Código de aplicación Información

Aplicación de ejemplo

Versión existente

Versiones de la aplicación que ha cargado.

Cargar el código

Cargue un paquete de código fuente desde su equipo o copie uno desde Amazon S3.

Valores preestablecidos Información

Comience a partir de un elemento preestablecido que coincide con su caso de uso o elija una configuración recomendados y utilice los valores predeterminados del servicio.

Elementos preestablecidos de configuración

Instancia única (compatible con la capa gratuita)

Instancia única (mediante instancia de spot)

Alta disponibilidad

Alta disponibilidad (con instancias de spot y bajo demanda)

Configuración personalizada

Acceso al servicio

Los roles de IAM, asumidos por Elastic Beanstalk como rol de servicio, y los perfiles de instancia de EC2 permiten a Elastic Beanstalk crear y administrar su entorno. Tanto el rol de IAM como el perfil de instancia deben estar asociados a políticas administradas de IAM que contengan los permisos necesarios.

[Más información](#)

Rol de servicio

Elija un rol de IAM para que Elastic Beanstalk asuma como rol de servicio. El rol de IAM debe tener las políticas administradas de IAM necesarias.

aws-elasticbeanstalk-service-role



[Crear rol](#)

Perfil de instancia de EC2

Elija un perfil de instancia de IAM con políticas administradas que permitan a las instancias de EC2 realizar las operaciones necesarias.

ec2-elasticbeanstalk-role



[Crear rol](#)

Par de claves de EC2: opcional

Seleccione un par de claves de EC2 para iniciar sesión de forma segura en sus instancias de EC2.

keys-linux-tajamar



[Crear](#)

Tipos de instancia

Agregue tipos de instancias para su entorno con el orden de lanzamiento que prefiera. La preferencia de spot que utilizan la estrategia de asignación priorizada y optimizada para la capacidad. Le recomendamos

1. t3.micro ▼ ▲ ▼
2. t3.small ▼ ▲ ▼ Eliminar

[Agregar tipo de instancia](#)

Mientras tanto, vamos a crear una nueva base de datos RDS.

Aurora and RDS

Servicio de bases de datos relacionales administrado

Crear base de datos [Información](#)

Elegir un método de creación de base de datos

Creación estándar

Puede definir todas las opciones de configuración, incluidas las de disponibilidad, seguridad, copias de seguridad y mantenimiento.

Creación sencilla

Utilice las configuraciones recomendadas. Algunas opciones de configuración se pueden cambiar después de crear la base de datos.

Opciones del motor

Tipo de motor [Información](#)

Aurora (MySQL Compatible)



Aurora (PostgreSQL Compatible)



MySQL



Plantillas

Elija una plantilla de ejemplo para adaptarla a su caso de uso.

Producción

Utilice los valores predeterminados para disfrutar de una alta disponibilidad y de un rendimiento rápido y constante.

Desarrollo y pruebas

Esta instancia se ha diseñado para su uso en desarrollo, fuera de un entorno de producción.

Capa gratuita

Utilice el nivel gratuito de RDS para desarrollar nuevas aplicaciones, probar aplicaciones existentes o adquirir experiencia práctica con Amazon RDS. [Información](#)

Configuración

Identificador de instancias de bases de datos [Información](#)

Escriba un nombre para la instancia de base de datos. El nombre debe ser único en relación con todas las instancias de base de datos para la región de AWS actual.

awsmysqlpaco

El identificador de la instancia de base de datos no distingue entre mayúsculas y minúsculas, pero se almacena con todas las letras en mayúsculas. Los caracteres permitidos incluyen letras, números y guiones. El primer carácter debe ser una letra. No puede contener dos guiones consecutivos.

Usuario: adminsql

Password: Admin123

Clases optimizadas para memoria (incluye clases r y x)

Clases ampliables (incluye clases t)

db.t3.micro

2 vCPUs 1 GiB RAM Red: hasta 2085 Mbps



Conectividad [Información](#)



Recurso de computación

Seleccione si desea configurar una conexión a un recurso de computación para esta base de datos. Al establecer una conexión, se cambiará automáticamente la configuración de conectividad para que el recurso de computación se pueda conectar a esta base de datos.

No se conecte a un recurso informático EC2

No configura una conexión a un recurso informático para esta base de datos. Puede configurar manualmente una conexión a un recurso informático más adelante.

Conectarse a un recurso informático de EC2

Configure una conexión a un recurso informático EC2 para esta base de datos.

Tipo de red [Información](#)

Para utilizar el modo de pila doble, asegúrese de asociar un bloque de CIDR IPv6 a una subred en la VPC que especifique.

IPv4

Sus recursos solo pueden comunicarse a través del protocolo de direcciones IPv4.

Modo de pila doble

Sus recursos pueden comunicarse a través de IPv4, IPv6 o ambos.

Nube privada virtual (VPC) [Información](#)

Elija la VPC. La VPC define el entorno de red virtual para esta instancia de DB.

Default VPC (vpc-066662b1b43cc56c0)

6 Subredes, 6 Zonas de disponibilidad



Solo se muestran las VPC con grupos de subredes de base de datos correspondientes.

Acceso público [Información](#)

Sí

RDS asigna una dirección IP pública a la base de datos. Las instancias de Amazon EC2 y otros recursos fuera de la VPC pueden conectarse a la base de datos. Los recursos de la VPC también pueden conectarse a la base de datos. Elija uno o varios grupos de seguridad de VPC que especifiquen qué recursos pueden conectarse a la base de datos.

No

RDS no asigna una dirección IP pública a la base de datos. Solo las instancias de Amazon EC2 y otros recursos dentro de la VPC pueden conectarse a la base de datos. Elija uno o varios grupos de seguridad de VPC que especifiquen qué recursos pueden conectarse a la base de datos.

Grupo de seguridad de VPC (firewall) [Información](#)

Elija uno o varios grupos de seguridad de VPC para permitir el acceso a su base de datos. Asegúrese de que las reglas del grupo de seguridad permitan el tráfico entrante adecuado.

Elegir existente

Elegir grupos de seguridad de VPC existentes

Crear nuevo

Crear un grupo de seguridad nuevo de VPC

Nuevo nombre del grupo de seguridad de VPC

grupo-rds-aws

Zona de disponibilidad [Información](#)

Sin preferencia



Copia de seguridad

- Habilitar las copias de seguridad automatizadas.
Crea una instantánea de un momento dado de su base de datos

Cifrado

- Habilitar el cifrado
Elija cifrar la instancia proporcionada. Los ID y alias de la clave maestra aparecen en la lista después de haberse creado mediante la Service. [Información](#)

Mantenimiento

Actualización automática de la versión secundaria [Información](#)

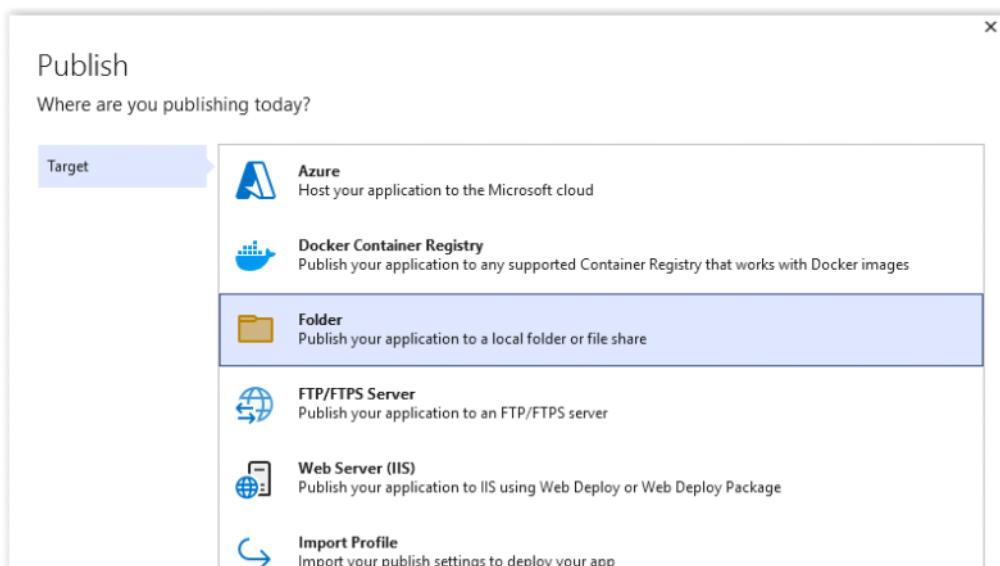
- Habilitar actualización automática de versiones secundarias
Al habilitar la actualización automática de la versión secundaria, se actualizará automáticamente la versión secundaria de la base de datos. Para conocer las limitaciones y obtener más información, consulte Actualizar automáticamente la versión secundaria del motor [documentación](#)

Creamos una nueva aplicación llamada **MvcAWSeriesELB**

Dicha aplicación, en su primera versión lo que hará será tener un triste título y una imagen.

Para publicar tenemos dos formas:

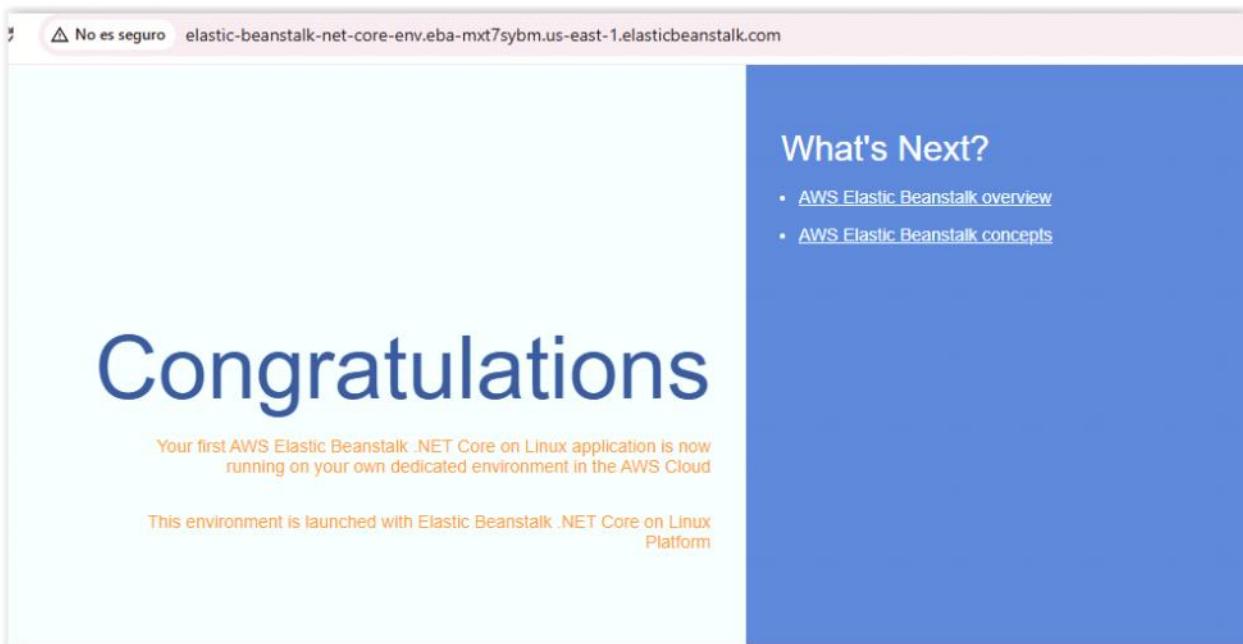
- 1) Manual con el Artifact directamente
- 2) Con Visual Studio Enterprise



Con los ficheros generados, debemos crear un **Artifact** con un ZIP

Nombre	Fecha de modificación	Tipo	Tamaño
wwwroot	19/05/2025 10:56	Carpeta de archivos	
appsettings.Development.json	19/05/2025 10:47	Archivo JSON	1 KB
appsettings.json	19/05/2025 10:47	Archivo JSON	1 KB
corev1.zip	19/05/2025 10:58	WinRAR ZIP archive	6.940 KB
MvcAWSeriesELB.deps.json	19/05/2025 10:56	Archivo JSON	1 KB
MvcAWSeriesELB.dll	19/05/2025 10:56	Extensión de la apl...	46 KB
MvcAWSeriesELB.exe	19/05/2025 10:56	Aplicación	142 KB
MvcAWSeriesELB.pdb	19/05/2025 10:56	Program Debug D...	34 KB
MvcAWSeriesELB.runtimeconfig.json	19/05/2025 10:56	Archivo JSON	1 KB
MvcAWSeriesELB.staticwebassets.endpo...	19/05/2025 10:56	Archivo JSON	827 KB
web.config	19/05/2025 10:56	Archivo CONFIG	1 KB

Ya deberíamos tener una App desplegada de prueba dentro de nuestro ELB



Entramos dentro del entorno y pulsamos sobre **Cargar e implementar**



- EF Microsoft.EntityFrameworkCore by aspnet, dotnetframework, EntityFramework, MySQL 9.0.5 Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations. EF Core works with SQL Server, Azure SQL...
- EF Microsoft.EntityFrameworkCore.Tools by aspnet, dotnetframework, EntityFramework 9.0.5 Entity Framework Core Tools for the NuGet Package Manager Console in Visual Studio.

MySQL MySQL.EntityFrameworkCore by MySQL, 9,07M downloads 9.0.3 MySQL.EntityFrameworkCore adds support for Microsoft Entity Framework Core.

Sobre **Models** creamos una clase llamada **Serie**

SERIE

```

[Table("SERIES")]
0 references
public class Serie
{
    [Key]
    [Column("IDSERIE")]
    0 references
    public int IdSerie { get; set; }
    [Column("SERIE")]
    0 references
    public string Nombre { get; set; }
    [Column("IMAGEN")]
    0 references
    public string Imagen { get; set; }
    [Column("ANYO")]
    0 references
    public int Anyo { get; set; }
}

```

Creamos una carpeta llamada **Data** y una clase llamada **SeriesContext**

SERIESCONTEXT

```

2 references
public class SeriesContext: DbContext
{
    0 references
    public SeriesContext(DbContextOptions<SeriesContext> options)
        :base(options) {}

    0 references
    public DbSet<Serie> Series { get; set; }
}

```

Sobre **Repositories** creamos una nueva clase llamada **RepositorySeries**

REPOSITORYSERIES

```

public class RepositorySeries
{
    private SeriesContext context;

    public RepositorySeries(SeriesContext context)
    {
        this.context = context;
    }

    public async Task<List<Serie>> GetSeriesAsync()
    {
        return await this.context.Series.ToListAsync();
    }

    private async Task<int> GetMaxIdSerieAsync()
    {
        return await this.context.Series.MaxAsync
            (x => x.IdSerie) + 1;
    }

    public async Task CreateSerieAsync(string nombre, string imagen
        , int anyo)
    {
        Serie serie = new Serie();
        serie.IdSerie = await this.GetMaxIdSerieAsync();
        serie.Nombre = nombre;
        serie.Imagen = imagen;
        serie.Anyo = anyo;
        await this.context.Series.AddAsync(serie);
        await this.context.SaveChangesAsync();
    }
}

```

Sobre **Controllers** creamos un nuevo controlador llamado **SeriesController**

SERIESCONTROLLER

```

public class SeriesController : Controller
{
    private RepositorySeries repo;

    public SeriesController(RepositorySeries repo)
    {
        this.repo = repo;
    }
}

```

```

    }

    public async Task<IActionResult> Index()
    {
        List<Serie> series = await this.repo.GetSeriesAsync();
        return View(series);
    }

    public IActionResult Create()
    {
        return View();
    }

    [HttpPost]
    public async Task<IActionResult> Create(Serie serie)
    {
        await this.repo.CreateSerieAsync(serie.Nombre, serie.Imagen
            , serie.Año);
        return RedirectToAction("Index");
    }
}

```

Incluimos la cadena de conexión en App Settings.json

```

{
    "Logging": {
        "LogLevel": {
            "Default": "Information",
            "Microsoft.AspNetCore": "Warning"
        }
    },
    "AllowedHosts": "*",
    "ConnectionStrings": {
        "MySql": "server=awsmysqlpaco.c23akq6oglxj.us-east-1.rds.amazonaws.com;port=3306;us"
    }
}

```

Resolvemos las dependencias dentro de Program

PROGRAM

```

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
string connectionString =
    builder.Configuration.GetConnectionString("MySql");
builder.Services.AddTransient<RepositorySeries>();
builder.Services.AddDbContext<SeriesContext>
    (options => options.UseMySQL(connectionString));
builder.Services.AddControllersWithViews();

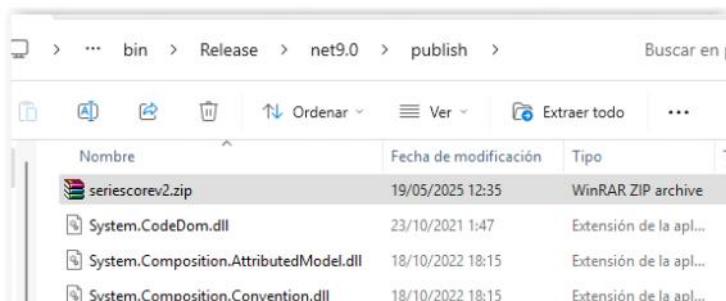
var app = builder.Build();

```

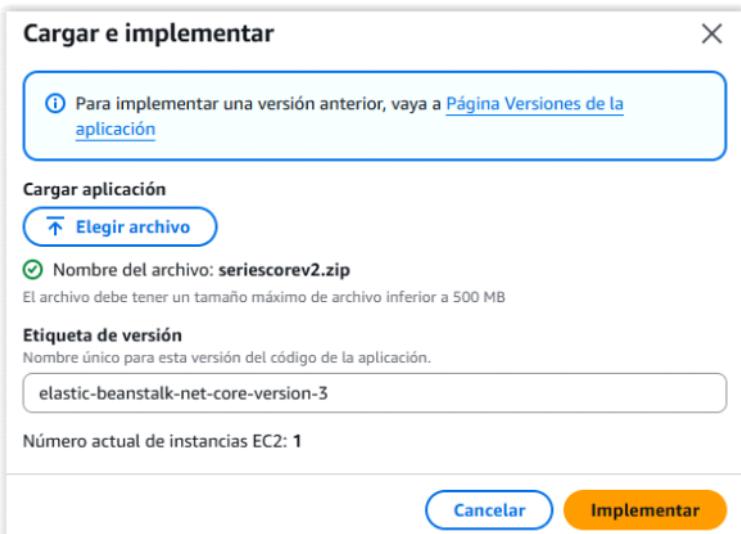
Una vez que hemos visto que es funcional la aplicación en Local, vamos a Desplegar en ELB.

Volvemos a publicar nuestra App.

Con el resultado, hacemos un ZIP llamado v2



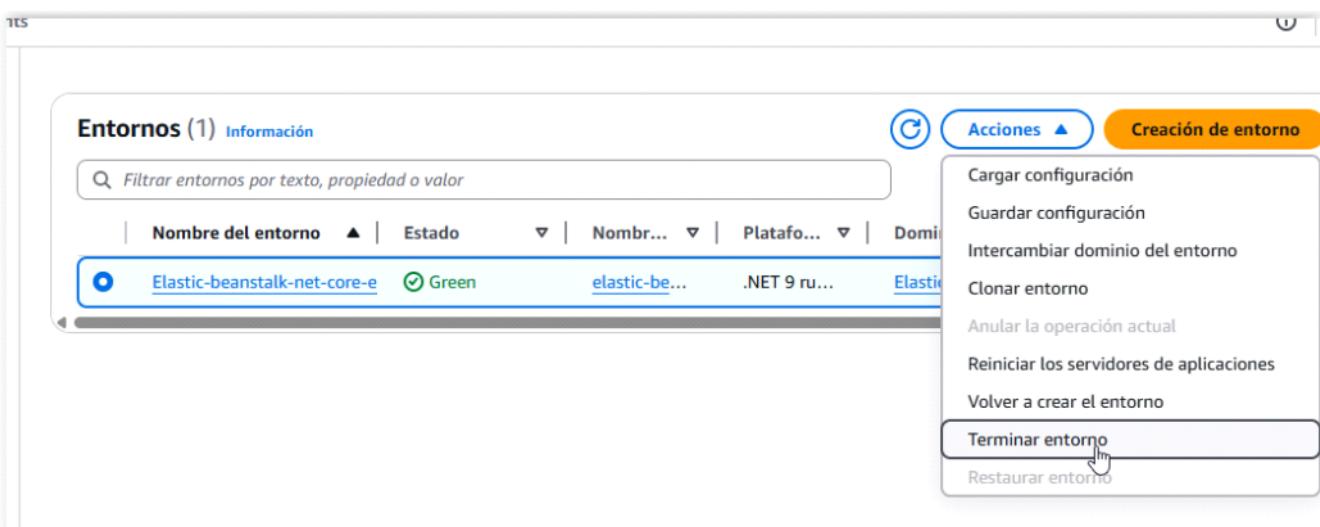
Entramos en nuestro entorno de App e implementamos la aplicación para ver si Todo es funcional.



Mientras está implementando nuestra App, instalamos el siguiente paquete con Visual Studio cerrado

[https://marketplace.visualstudio.com/items?
itemName=AmazonWebServices.AWSToolkitforVisualStudio2022](https://marketplace.visualstudio.com/items?itemName=AmazonWebServices.AWSToolkitforVisualStudio2022)

Mientras lo va instalando, eliminamos nuestro entorno completo de Elastic Beanstalk



La extensión AWS para Visual Studio nos permite administrar ciertas características Y proyectos de AWS en nuestra cuenta, al estilo de Azure.

Existen ciertos elementos que podemos hacer de forma automática como, por ejemplo, Desplegar una aplicación en un Elastic Beanstalk directamente sin entrar en la Consola de AWS

También podemos crear Lambda y publicar directamente.

Para publicar, necesitamos un usuario de AWS con permisos para acceder a nuestra cuenta Y sus recursos.

Como estamos compartiendo equipo con los de la tarde y AWS todavía no tiene muy pulido Los usuarios de Windows con los usuarios de AWS y, a veces, se pueden mezclar.

Vamos a crear un usuario de IAM administrador para visualizar la extensión que Acabamos de crear y, al finalizar la clase, **TODOS LOS DIAS**, quitamos al usuario del grupo de Admins de AWS.

Vamos a crear un nuevo usuario para VS llamado **user-visual-studio**

Dicho usuario estará en un grupo de administración que todos los días quitaremos.

Buscamos IAM y creamos un nuevo grupo llamado **grupo-visual-studio**

Filtrar por Tipo	
Buscar	Todos los tipos
<input type="checkbox"/> Nombre de la política	▲ Tipo
<input checked="" type="checkbox"/>  AdministratorAccess	Administrada por AWS: f... Ninguno

Este usuario **NO** tendrá acceso a la consola de AWS

Detalles de la persona

Nombre de usuario

user-visual-studio

El nombre de usuario puede tener un máximo de 64 caracteres. Caracteres válidos: A-Z, a-z, 0-9, and +, . @ _ - (guion)

Proporcione acceso de usuario a la consola de administración de AWS: *opcional*

Si proporciona acceso a la consola a una persona, se trata de un [práctica recomendada](#) para administrar su acceso en IAM Identity Center.

- ⓘ Si está creando acceso mediante programación a través de claves de acceso o credenciales específicas de servicios para Amazon Keyspaces, puede generarlos después de crear este usuario de IAM. [Más información](#)

Opciones de permisos

Agregar persona al grupo

Agregue la persona a un grupo existente o cree uno nuevo. Le recomendamos que utilice grupos para administrar los permisos de usuario según las funciones laborales.

Copiar permisos

Copie todas las suscripciones a grupos, las políticas administradas adjuntas y las políticas insertadas de una persona existente.

Adjuntar política

Adjunte una política de manera directa. Le sugerimos, en caso de que no lo haga, que la adjunte a un grupo. A continuación, seleccione el grupo adecuado.

Grupos de personas (1/1)

Buscar

Nombre del grupo

▲ | Personas

▼ | Políticas adjuntas

▼ | Creado

grupo-visual-studio

0

AdministratorAccess

2025-

Necesitamos las claves de acceso de este usuario para introducirlas dentro de Visual Studio

Entramos en [Credenciales](#)

Claves de acceso (0)

[Crear clave de acceso](#)

Utilice las claves de acceso para enviar llamadas mediante programación a AWS desde AWS CLI, Herramientas de AWS para PowerShell, AWS SDK o llamadas directas a la API de AWS. Puede tener un máximo de dos claves de acceso (activas o inactivas) a la vez. [Más información](#)

No hay claves de acceso. Como práctica recomendada, evite el uso de credenciales a largo plazo, como las claves de acceso. En su lugar, utilice herramientas que proporcionen credenciales a corto plazo. [Más información](#)

[Crear clave de acceso](#)

Prácticas recomendadas y alternativas para la clave de acceso Informació

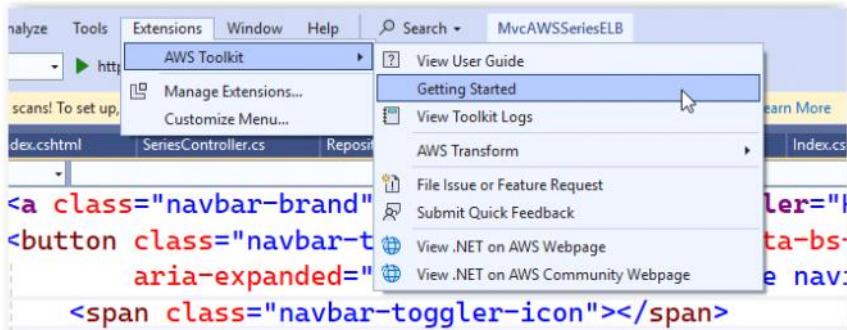
Evite utilizar credenciales a largo plazo como claves de acceso para mejorar su seguridad. Tenga en cuenta los siguientes uso y alternativas.

Caso de uso

● Interfaz de línea de comandos (CLI)

Tiene previsto utilizar esta clave de acceso para permitir que la AWS CLI obtenga acceso a su cuenta de AWS.

Estas credenciales irán dentro de VS.



aws Getting Started: AWS Toolkit with Amazon Q

[Documentation](#) | [GitHub](#)

Features

You can return to this page at any time (Extensions > AWS Toolkit > Getting Started)

Amazon Q Developer and AWS Transform

Build applications faster with your AI coding companion.

[Learn more](#)

[Enable](#)

AWS Toolkit

View, modify, and deploy AWS Resources. Work with S3, Lambda, CloudWatch, and more.

[Learn more](#)

[Enable](#)

aws Getting Started: AWS Toolkit with Amazon Q

[Documentation](#) | [GitHub](#)

← Features

Setup authentication for AWS Toolkit

AWS Toolkit does not support authentication with AWS Builder ID. [Learn more about supported authentication providers](#).

[Edit credentials file directly...](#)

Profile Type [?](#)

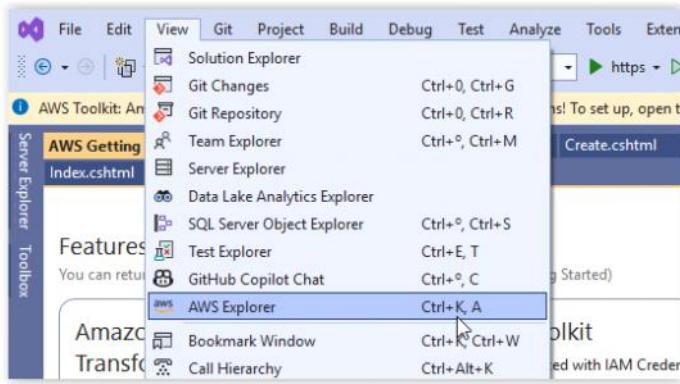
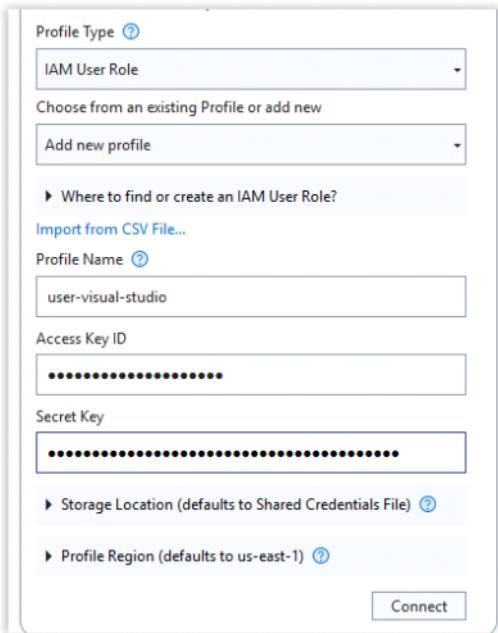
IAM User Role

Choose from an existing Profile or add new

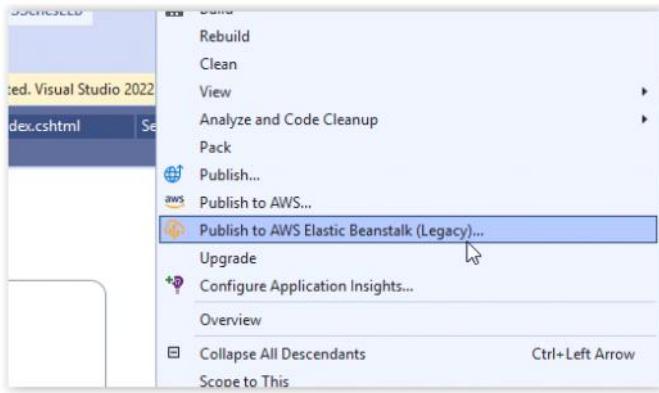
Profile:default

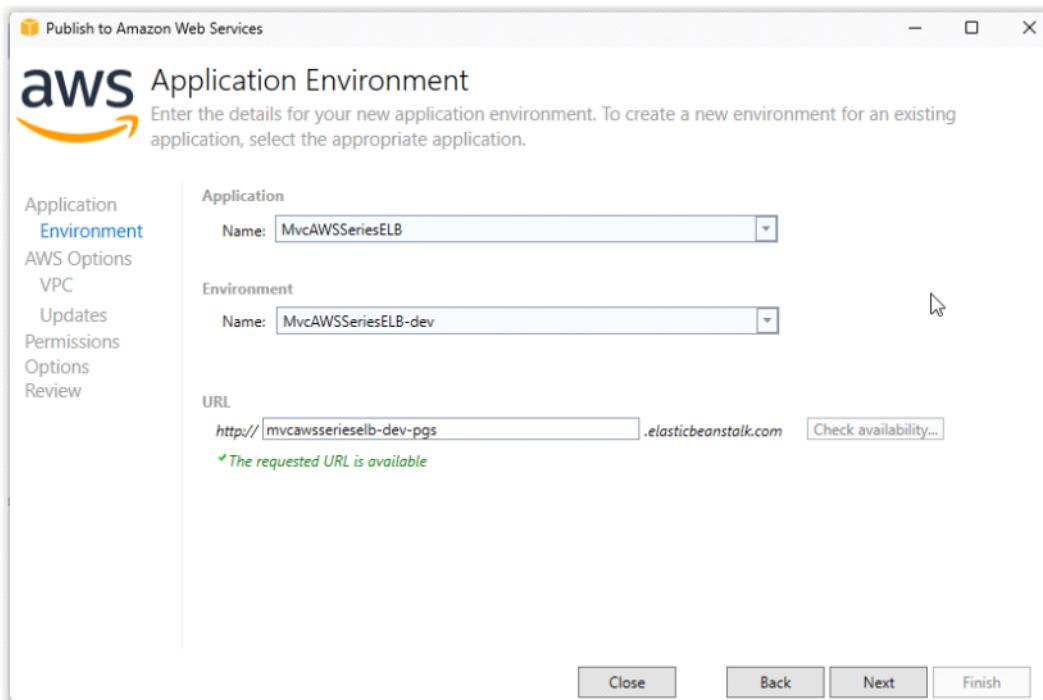
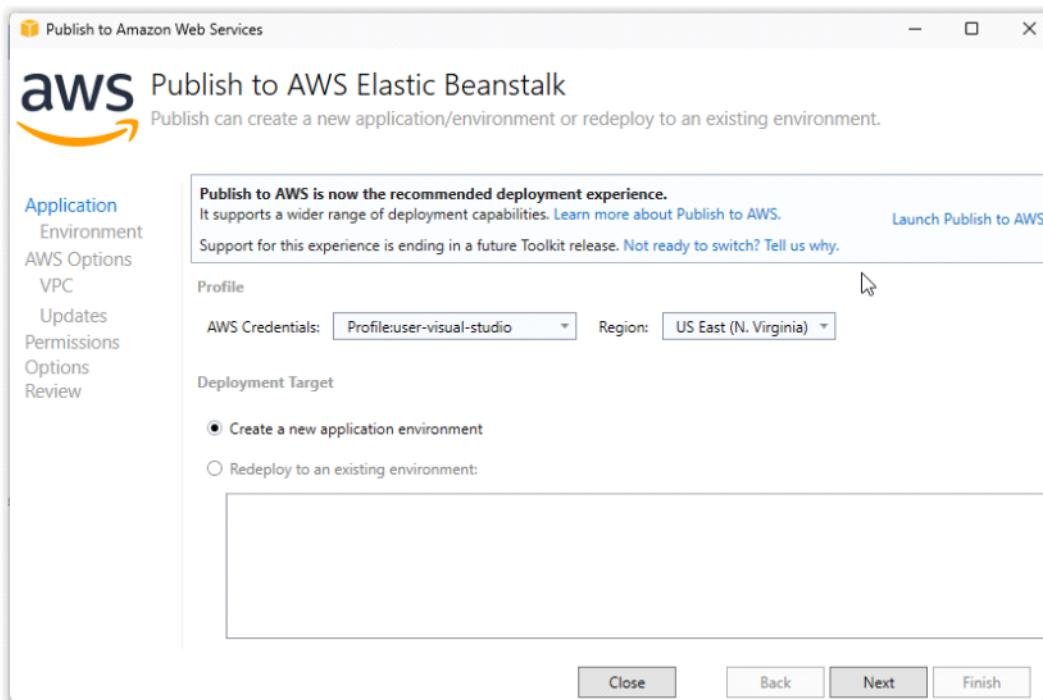
Add new profile

Profile:default

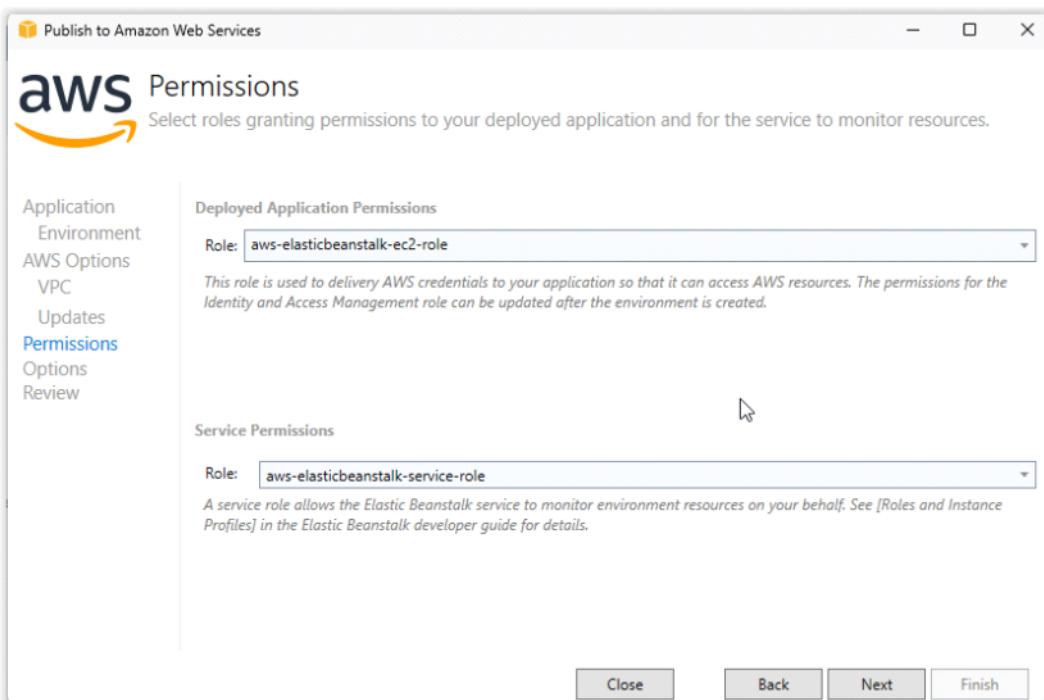
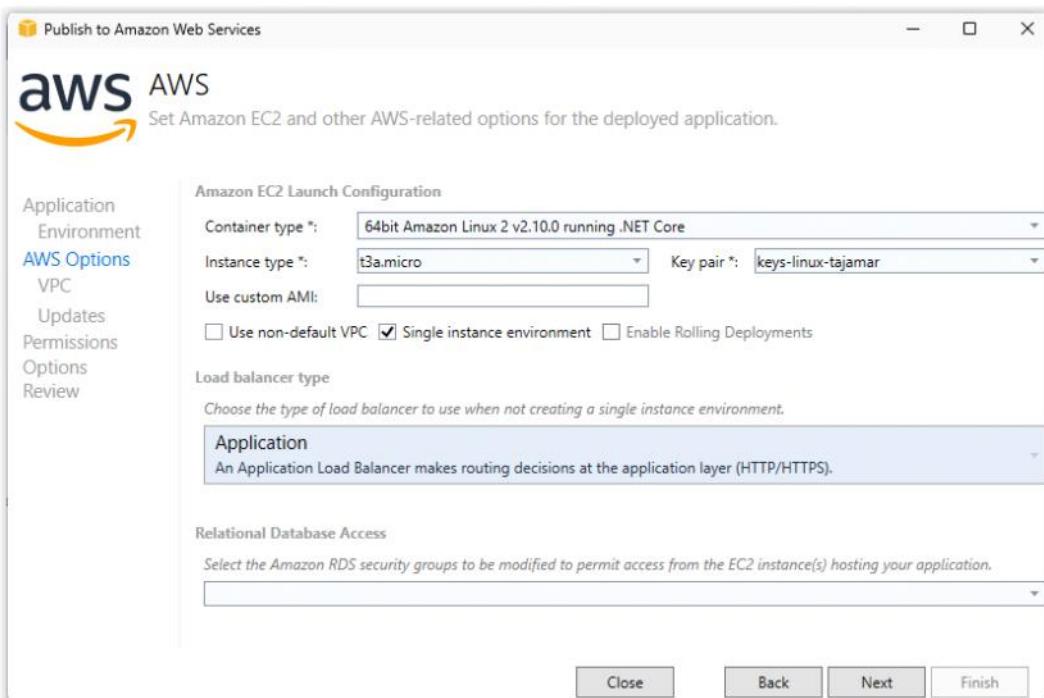


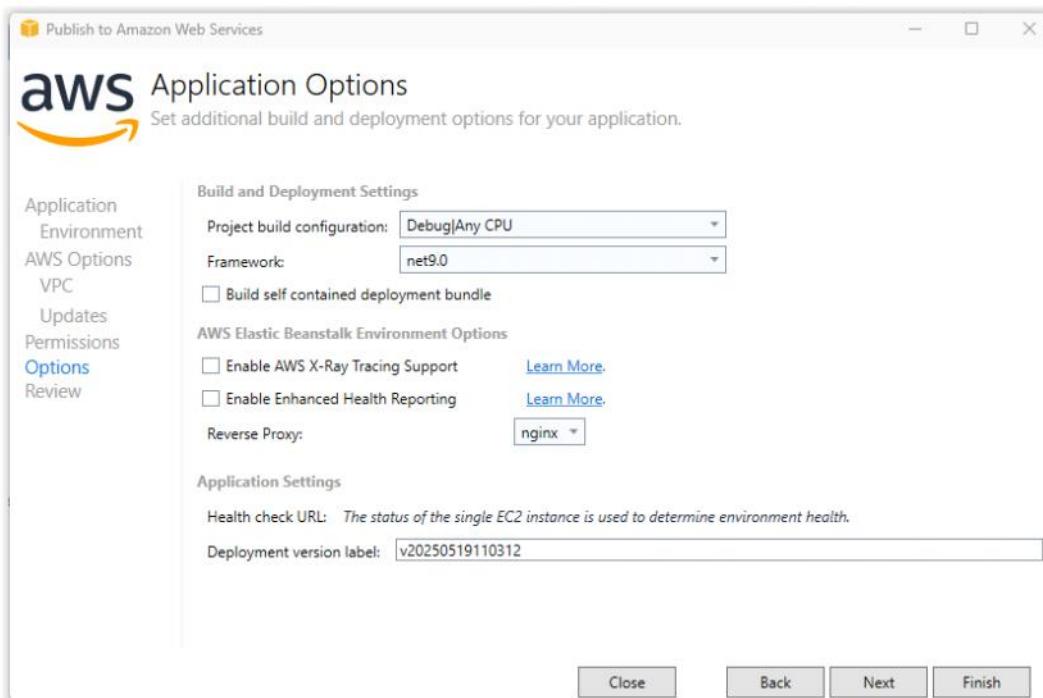
Deberíamos tener habilitado también un nuevo menú para desplegar aplicaciones Directamente desde el proyecto.



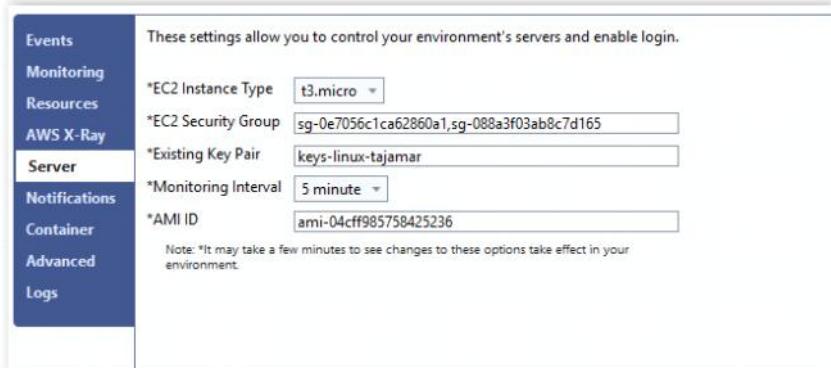


IMPORTANTE: El tipo de instancia siempre será nuestro dinero...Necesitamos MICRO





Si nos da error, ponemos **T2 MICRO o T3 MICRO**



AWS DYNAMO DB

martes, 20 de mayo de 2025 9:11

AWS Dynamo DB es la base de datos NoSQL de Amazon.

Es exactamente igual a Mongo DB o Azure Cosmos DB, es decir, trabajan con tipos de datos JSON en su interior del mismo tipo y que no estén relacionados.

Le encanta a Amazon porque es la base de datos con mayor disponibilidad y contiene una implementación llamada DAX que optimiza las respuestas.

Tiene objetos tipados en su interior, tales como string, number o double.

Tiene una clave de Partición (PK) y otra clave de ordenación FK

Vamos a trabajar en Local, por lo que necesitamos un usuario con permisos sobre Dynamo Db.

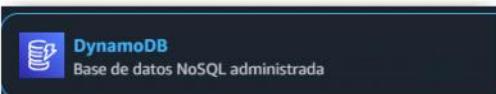
Comenzamos con IAM y creamos un usuario llamado **user-dynamo** en un grupo llamado **grupo-dynamo**



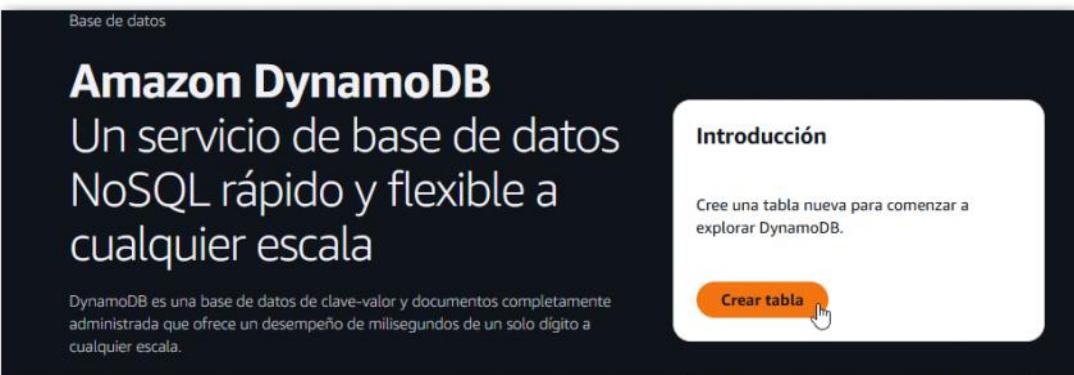
Introducimos las claves dentro de nuestro SO

```
C:\Program Files\Cmdr\n
\ aws configure
AWS Access Key ID [*****XZVV]: AKIA3HQRDDS6ZMCCTZPT
AWS Secret Access Key [*****gnqf]: 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
Default region name [us-east-1]:
Default output format [json]:
```

Abrimos el servicio de Dynamo Db



Creamos una nueva tabla llamada coches



Crear tabla

Detalles de la tabla Información

DynamoDB es una base de datos sin esquemas que solo requiere un nombre de tabla y una clave principal al crear la tabla.

Nombre de la tabla

Se utilizará para identificar su tabla.

Entre 3 y 255 caracteres. Solo se pueden usar letras, números, guiones bajos (_), guiones (-) y puntos (.)

Clave de partición

La clave de partición forma parte de la clave principal de la tabla. Se trata de un valor hash que se utiliza para recuperar elementos de la tabla, así como para asignar datos entre hosts por cuestiones de escalabilidad y disponibilidad.

De 1 a 255 caracteres, distingue entre mayúsculas y minúsculas.

Clave de ordenación - *opcional*

Puede utilizar una clave de ordenación como segunda parte de la clave principal de una tabla. La clave de ordenación le permite ordenar o buscar entre todos los elementos que comparten la misma clave de partición.

De 1 a 255 caracteres, distingue entre mayúsculas y minúsculas.

Creamos una nueva aplicación llamada **MvcDynamoDbCoches**

 **AWSSDK.Extensions.NETCore.Setup** by awsdotnet, 159M downloads
Extensions for the AWS SDK for .NET to integrate with .NET Core configuration and dependency injection frameworks.

4.0.1

 **AWSSDK.DynamoDBv2** by awsdotnet, 135M downloads
Amazon DynamoDB is a fast and flexible NoSQL database service for all applications that need consistent, single-digit millisecond latency at any scale.

Sobre **Models** creamos una nueva clase llamada **Coche**

```
[DynamoDBTable("coches")]
0 references
public class Coche
{
    [DynamoDBHashKey]
    [DynamoDBProperty("idcoche")]
    0 references
    public int IdCoche { get; set; }
    [DynamoDBProperty("marca")]
    0 references
    public string Marca { get; set; }
    [DynamoDBProperty("modelo")]
    0 references
    public string Modelo { get; set; }
    [DynamoDBProperty("imagen")]
    0 references
    public string Imagen { get; set; }
}
```

Creamos una carpeta llamada **Services** y una nueva clase llamada **ServiceDynamoDb**

```
SERVICEDYNAMODB
public class ServiceDynamoDb
{
    private DynamoDBContext context;
    public ServiceDynamoDb()
    {
        //TENEMOS UN CLIENT PARA CREAR EL CONTEXT DE DYNAMO
        AmazonDynamoDBClient client = new AmazonDynamoDBClient();
        //DynamoDBContextBuilder builder = new DynamoDBContextBuilder();
        //builder.();
        this.context = new DynamoDBContext(client);
    }
    public async Task CreateCocheAsync(Coche car)
    {
        await this.context.SaveAsync<Coche>(car);
    }
    public async Task DeleteCocheAsync(int idCoche)
    {
        await this.context.DeleteAsync<Coche>(idCoche);
    }
    public async Task<Coche> FindCocheAsync(int idCoche)
    {
        //SI ESTAMOS BUSCANDO POR PARTITION KEY TENEMOS
        //UN METODO QUE LO REALIZA POR NOSOTROS
        return await this.context.LoadAsync<Coche>(idCoche);
    }
}
```

```

public async Task<List<Coche>> GetCochesAsync()
{
    //DEBEMOS BUSCAR LA TABLA QUE CORRESPONDE A NUESTRO MODEL
    ITable tabla = this.context.GetTargetTable<Coche>();
    //PARA BUSCAR, NO SABEMOS SI BUSCAMOS TODOS O FILTRAMOS
    //DEBEMOS CREAR UN OBJETO LLAMADO ScanOptions QUE LLEVARIA LOS
    //FILTROS
    var scanOptions = new ScanOperationConfig();
    var results = tabla.Scan(scanOptions);
    //DYNAMO DB DENOMINA A LO QUE ALMACENA COMO Document
    List<Document> data = await results.GetNextSetAsync();
    //CONVERTIMOS DOCUMENT A NUESTRO MODEL
    var cars =
        this.context.FromDocuments<Coche>(data);
    return cars.ToList();
}

```

Sobre Controllers agregamos un nuevo controlador llamado **CochesController**

COCHESCONTROLLER

```

public class CochesController : Controller
{
    private ServiceDynamoDb service;
    public CochesController(ServiceDynamoDb service)
    {
        this.service = service;
    }
    public async Task<IActionResult> Index()
    {
        List<Coche> cars = await this.service.GetCochesAsync();
        return View(cars);
    }
    public async Task<IActionResult> Details(int idcoche)
    {
        Coche car = await this.service.FindCocheAsync(idcoche);
        return View(car);
    }
    public async Task<IActionResult> Delete(int idcoche)
    {
        await this.service.DeleteCocheAsync(idcoche);
        return RedirectToAction("Index");
    }
    public IActionResult Create()
    {
        return View();
    }
    [HttpPost]
    public async Task<IActionResult> Create(Coche car)
    {
        await this.service.CreateCocheAsync(car);
        return RedirectToAction("Index");
    }
}

```

Inyectamos el servicio dentro de Program

PROGRAM

```

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddTransient<ServiceDynamoDb>();
builder.Services.AddControllersWithViews();

```

Una vez que hemos visto la funcionalidad, vamos a incluir algo que cree objetos Dinámicos en DynamoDb.

Incluiremos objetos con Motor y objetos sin Motor.

Sobre Models creamos una nueva clase llamada **Motor**

MOTOR

```

public class Motor
{
    [DynamoDBProperty("tipo")]
    0 references
    public string Tipo { get; set; }
    [DynamoDBProperty("caballos")]
    0 references
    public int Caballos { get; set; }
    [DynamoDBProperty("potencia")]
    0 references
    public int Potencia { get; set; }
}

```

Sobre la clase **Coche** incluimos una nueva propiedad del objeto Motor.

COCHE

```

public string Imagen { get; set; }
[DynamoDBProperty("motor")]
0 references
public Motor Motor { get; set; }
}

```

Sobre la vista **Create.cshtml** agregamos la posibilidad de incluir Motor.

CREATE.CSHTML

```

<div class="form-group">
    <input type="checkbox" name="motor" value="true"/>Motor
    <label>Tipo</label>
    <input type="text" name="tipo" class="form-control"/>
    <label>Caballos</label>
    <input type="text" name="caballos" class="form-control"/>
    <label>Potencia</label>
    <input type="text" name="potencia" class="form-control"/>
</div>

```

Modificamos el método POST de Create en el Controller

COCHESCONTROLLER

```

[HttpPost]
public async Task<IActionResult> Create
    (Coche car, string motor, string tipo
     , int potencia, int caballos)
{
    if (motor != null)
    {
        car.Motor = new Motor();
        car.Motor.Tipo = tipo;
        car.Motor.Caballos = caballos;
        car.Motor.Potencia = potencia;
    }
    await this.service.CreateCocheAsync(car);
    return RedirectToAction("Index");
}

```

Dibujamos el motor dentro de **Details.cshtml**

DETAILS.CSHTML

```

<div class="card" style="width: 18rem;">
    
    <div class="card-body">
        <h5 class="card-title">@Model.Marca @Model.Modelo</h5>
        @if (Model.Motor != null) {
            <p class="card-text">Tipo: @Model.Motor.Tipo</p>
            <p class="card-text">Caballos: @Model.Motor.Caballos</p>
            <p class="card-text">Potencia: @Model.Motor.Potencia</p>
        }
        <a asp-controller="Coches"
           asp-action="Index"
           class="btn btn-primary">Volver a coches</a>
    </div>
</div>

```



Index

[Create New](#)

IdCoche	Marca	Modelo	Imagen	Motor		
2	DMG	DELOREAN	Tiene motor		Details	Delete
1	POCO	YO	Sin motor		Details	Delete

Nos quedan por resolver dos características:

- 1) Buscador por alguna Propiedad. Vamos a realizar un método que buscará coches Por Marca

SERVICEDYNAMODB

```
public async Task<List<Coche>> SearchCochesMarcaAsync(string marca)
{
    //PARA BUSCAR SE UTILIZAN UN CONJUNTO DE CONDICIONES LLAMADAS
    //ScanConditions
    List<ScanCondition> conditions = new List<ScanCondition>();
    conditions.Add(new ScanCondition("Marca", ScanOperator.Equal, marca));
    var cars = await
        this.context.ScanAsync<Coche>(conditions)
            .GetRemainingAsync();
    return cars.ToList();
}
```

Implementaremos la funcionalidad dentro de **CochesController**

COCHESCONTROLLER

```
public IActionResult Buscador()
{
    return View();
}

[HttpPost]
public async Task<IActionResult> Buscador(string marca)
{
    List<Coche> cars = await this.service
        .SearchCochesMarcaAsync(marca);
    return View(cars);
}
```

BUSCADOR.CSHTML

```
@model IEnumerable<MvcDynamoDbCoches.Models.Coche>
@{
    ViewData["Title"] = "Index";
}



# Buscador Coches



<form method="post">
    <label>Introduzca marca:</label>
    <input type="text" name="marca"
           class="form-control"/>
    <button class="btn btn-warning">
        Buscar coches
    </button>
</form>

@if (Model != null)
{
    <table class="table table-info">
        <thead>
            <tr>
                <th>
                    @Html.DisplayNameFor(model => model.IdCoche)
                </th>
                <th>
                    @Html.DisplayNameFor(model => model.Marca)
                </th>
                <th>
                    @Html.DisplayNameFor(model => model.Modelo)
                </th>
                <th>
                    @Html.DisplayNameFor(model => model.Imagen)
                </th>
                <th>Motor</th>
                <th></th>
            </tr>
        <tbody>
            @foreach (var item in Model)
            {
                <tr>
                    <td>
                        @item.IdCoche
                    </td>
                    <td>
                        @item.Marca
                    </td>
                    <td>
                        @item.Modelo
                    </td>
                    <td>
                        @item.Imagen
                    </td>
                    <td>
                        @item.Motor
                    </td>
                    <td>
                        <a href="#">Edit</a> <a href="#">Delete</a>
                    </td>
                </tr>
            }
        </tbody>
    </table>
}

```

```

        </tr>
    </thead>
    <tbody>
        @foreach (var item in Model)
        {
            <tr>
                <td> @Html.DisplayFor(modelItem => item.IdCoche) </td>
                <td> @Html.DisplayFor(modelItem => item.Marca) </td>
                <td> @Html.DisplayFor(modelItem => item.Modelo) </td>
                <td>
                    @if (item.Motor != null)
                    {
                        <span style="color:lawngreen">Tiene motor</span>
                    }
                    else
                    {
                        <span style="color:red">Sin motor</span>
                    }
                </td>
                <td>  </td>
                <td>
                    <a asp-controller="Coches"
                        asp-action="Details"
                        asp-route-idcoche="@item.IdCoche"
                        class="btn btn-info">Details</a>
                    <a asp-controller="Coches"
                        asp-action="Delete"
                        asp-route-idcoche="@item.IdCoche"
                        class="btn btn-danger">Delete</a>
                </td>
            </tr>
        }
    </tbody>
</table>
}

```

2) Inyección de dependencias dentro de Program y hacerlo bien sin deprecated

```

private DynamoDBContext context;

0 references
public ServiceDynamoDb()
{
    //TENEMOS UN CLIENT PARA CREAR EL CONTEXT DE DYNAMO
    AmazonDynamoDBClient client = new AmazonDynamoDBClient();
    //DynamoDBContextBuilder builder = new DynamoDBContextBuilder();
    //builder.;

    this.context = new DynamoDBContext(client);
}

```

SOLUCION MANUEL

```

public class ServiceDynamoDb
{
    private IDynamoDBContext context;

    0 references
    public ServiceDynamoDb(IDynamoDBContext context)
    {
        this.context = context;
    }
}

```

```

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddAWSService<IAmazonDynamoDB>();
builder.Services.AddSingleton<IDynamoDBContext, DynamoDBContext>();
builder.Services.AddTransient<ServiceDynamoDb>();
builder.Services.AddControllersWithViews();

```

Otra solución más, gracias Victor

Víctor Castrillo Redondo 11:11 Traducir



```
public ServiceDynamoDb(IAmazonDynamoDB client)
{
    _context = new DynamoDBContextBuilder()
        .WithDynamoDBClient(() => client)
        .Build();
}
```

AWS SECRETS MANAGER

martes, 20 de mayo de 2025 11:12

Este servicio es el equivalente a Azure Key Vault.

Antiguamente, se utilizaba un servicio llamado KMS como equivalente, pero dicho Servicio se ha dejado para el cifrado de elementos como, por ejemplo, los buckets

Con este servicio tendremos todas nuestras claves en la nube de AWS y no estarán expuestas en appsettings.json

Nota: Para utilizar este servicio necesitamos un usuario IAM con permisos para El acceso a los Secrets.

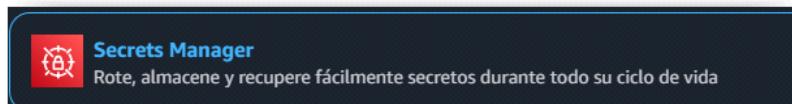
Este servicio tiene características como claves dedicadas, es decir, tiene un elemento Para poder almacenar directamente cadenas de conexión de RDS o podríamos incluir Nuestras claves dedicadas.

Vamos a realizar un ejemplo en el que almacenaremos nuestras claves ficticias En formato JSON.

Tendremos un JSON inventado como este:

JSON

```
{  
    "MySQL": "server=...;port=3306",  
    "BucketImagenes": "https://bucket-imagenes",  
    "ApiSeries": "https://apiseries.html"  
}
```



Tipo de secreto Información

Credenciales para la base de datos de Amazon RDS
 Credenciales para otra base de datos
 Otro tipo de secreto
Clave API, token OAuth, otros.

Pares clave-valor Información

Clave-valor **Texto no cifrado**

MySQL	server=miservidor;port=3306;user id=adminssql	Eliminar
Bucket	https://mibucket.com	Eliminar
Api	https://apiejemplos.azurewebsites.net	Eliminar

+ Agregar fila

Clave de cifrado Información

Puede cifrar con la clave de KMS creada por Secrets Manager o con una clave de KMS administrada por el cliente creada por usted.

aws/secretsmanager **▼** **C**

Agregar clave nueva **✚**

Configurar secreto

Nombre y descripción del secreto [Información](#)

Nombre del secreto

Un nombre descriptivo que le ayuda a encontrar el secreto más adelante.

databsecrets

El nombre del secreto solo debe contener caracteres alfanuméricos y los caracteres / _ += . @ ~

Descripción - opcional

Secretos de martes

Máximo de 250 caracteres.

Configurar rotación - opcional

Configurar rotación automática [Información](#)

Configure AWS Secrets Manager para rotar este dato confidencial de forma automática.

Rotación automática

Programación de rotación [Información](#)

Programador de expresiones de programación

expresión de programación

Unidad de tiempo: Horas

Nos genera un código de muestra para poder recuperar nuestras claves con distintos

Lenguajes

Código de muestra

Utilice estos ejemplos de código para recuperar el secreto en la aplicación.

Java

JavaScript

C#

Python3

Ruby

Go

Rust

```
1  /*
2   * Use this code snippet in your app.
3   * If you need more information about configurations or implementing the sample code, visit the AWS docs:
4   * https://aws.amazon.com/developer/language/net/getting-started
5   */
6
7  using Amazon;
8  using Amazon.SecretsManager;
9  using Amazon.SecretsManager.Model;
10
11 static async Task GetSecret()
12 {
13     string secretName = "databsecrets";
14     string region = "us-east-1";
15 }
```

C# Línea 1, Columna 1 |  Errores: 0 |  Advertencias: 0

El siguiente paso es probar nuestro código y el acceso a las claves.

Creamos una nueva aplicación llamada **AWSConsoleSecretsManager**



Microsoft.Extensions.DependencyInjection  by [aspnet](#), [dotnetframework](#) 9.0.5
Default implementation of dependency injection for Microsoft.Extensions.DependencyInjection.

 Microsoft.Extensions.Configuration by aspnet, dotnetframework, Microsoft 9.0.5
Implementation of key-value pair based configuration for Microsoft.Extensions.Configuration. Includes the memory configuration provider.

 Newtonsoft.Json by dotnetfoundation, jamesnk, newtonsoft, 6,24B download 13.0.3
Json.NET is a popular high-performance JSON framework for .NET

 AWSSDK.SecretsManager by awsdotnet, 119M downloads 4.0.0.4
AWS Secrets Manager enables you to easily create and manage the secrets that you use in your customer-facing apps. Instead of embedding credentials into your sour...

Vamos a crear un Model con las claves que necesitemos.

Dicho Model será el que utilizaremos en cada Service/Repo que necesite de nuestras Claves

Sobre **Models** creamos una nueva clase llamada **KeysModel**

KEYSMODEL

```
0 references
public class KeysModel
{
    0 references
    public string MySql { get; set; }
    0 references
    public string Bucket { get; set; }
    0 references
    public string Api { get; set; }
}
```

Tendremos un Helper con el código algo modificado que hemos copiado de la generación Del Secret

Creamos una carpeta llamada **Helpers** y una clase llamada **HelperSecretManager**

HELPERSECRETMANAGER

```
public class HelperSecretManager
{
    public static async Task<string> GetSecretAsync()
    {
        string secretName = "datasecrets";
        string region = "us-east-1";

        IAmazonSecretsManager client = new AmazonSecretsManagerClient(RegionEndpoint.GetBySystemName(region));

        GetSecretValueRequest request = new GetSecretValueRequest
        {
            SecretId = secretName,
            VersionStage = "AWSCURRENT", // VersionStage defaults to AWSCURRENT if unspecified.
        };

        GetSecretValueResponse response;
        response = await client.GetSecretValueAsync(request);
        string secret = response.SecretString;
        return secret;
    }
}
```

El siguiente paso es recuperar los secretos desde **Program**

PROGRAM

```
using AWSConsoleSecretsManager.Helpers;
using AWSConsoleSecretsManager.Models;
using Newtonsoft.Json;
using System.Runtime.CompilerServices;

Console.WriteLine("Ejemplo Secrets Manager");
string miSecreto = await HelperSecretManager.GetSecretAsync();
Console.WriteLine(miSecreto);
//PODEMOS SERIALIZAR NUESTRO STRING Y DESERIALIZARLO CON
//LA CLASE KEYSMODEL YA QUE LAS PROPIEDADES SE LLAMAN IGUAL
KeysModel model = JsonConvert.DeserializeObject<KeysModel>(miSecreto);
Console.WriteLine("MySQL: " + model.MySql);
```

```
Console.WriteLine("Pulse enter para finalizar");
Console.ReadLine();
```

Creamos un nuevo grupo llamado **grupo-secrets** y un usuario llamado **user-secrets**



Las credenciales de este usuario van dentro de **aws configure**

```
C:\Users\Profesor MCSD Mañana
λ aws configure
AWS Access Key ID [*****TZPT]: AKIA3HQRDDS647U3BHMN
AWS Secret Access Key [*****yaNw]: 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
Default region name [us-east-1]:
Default output format [json]:
```

Y ya podremos ejecutar nuestra aplicación

```
λ C:\Users\Profesor MCSD Mañana
Ejemplo Secrets Manager
{"MySql":"server=miservidor;port=3306;user id=admins", "Bucket":"https://mibucket.com", "Api":"https://apiejemplos.azurewebsites.net"}
MySQL: server=miservidor;port=3306;user id=admins
Pulse enter para finalizar
```

Vamos a incluir algo más de código, emulando como si estuviéramos en un entorno
Más complejo, estilo MVC o API con D.Y.

```
public class Controller
{
    public Controller(KeysModel model){
        //Utilizamos las Keys
    }
}
```

Creamos una clase llamada **ClaseTest** que recibirá nuestro objeto de Keys

```
public class ClaseTest
{
    private KeysModel keys;

    public ClaseTest(KeysModel keys)
    {
        this.keys = keys;
    }

    public string GetApiValue()
    {
        return this.keys.Api;
    }
}
```

Modificamos el código del **Program**

PROGRAM

```
using AWSConsoleSecretsManager;
using AWSConsoleSecretsManager.Helpers;
using AWSConsoleSecretsManager.Models;
using Microsoft.Extensions.DependencyInjection;
using Newtonsoft.Json;
using System.Runtime.CompilerServices;

Console.WriteLine("Ejemplo Secrets Manager");
string miSecreto = await HelperSecretManager.GetSecretAsync();
Console.WriteLine(miSecreto);
//PODEMOS SERIALIZAR NUESTRO STRING Y DESERIALIZARLO CON
//LA CLASE KEYSMODEL YA QUE LAS PROPIEDADES SE LLAMAN IGUAL
KeysModel model = JsonConvert.DeserializeObject<KeysModel>(miSecreto);
Console.WriteLine("MySQL: " + model.MySql);
```

```
//ALMACENAMOS EL OBJETO KEYSMODEL DENTRO DEL BUILDER
var provider = new ServiceCollection()
    .AddTransient<ClaseTest>()
    .AddSingleton<KeysModel>(x => model)
    .BuildServiceProvider();

//EN CUALQUIER CLASE YA PODEMOS RECUPERAR LAS KEYS
//VAMOS A COMPROBAR SI FUNCIONA DENTRO DE CLASETEST
var test = provider.GetService<ClaseTest>();
Console.WriteLine("Api Value: " + test.GetApiValue());

Console.WriteLine("Pulse enter para finalizar");
Console.ReadLine();
```

AWS LAMBDA

martes, 20 de mayo de 2025 13:38

Son funciones **Serverless**, es decir, un código autónomo dentro de la nube de AWS.
Son el equivalente a Azure Functions.

Se pueden escribir en múltiples lenguajes: Python, Node JS, C#, Java...

Dichos códigos son desplegados en la nube y pueden ser utilizados por múltiples elementos.

Por ejemplo, podríamos tener una función que valide credenciales contra Amazon Cognito y Que nos devuelva un Token de acceso si son correctas

Por ejemplo, podríamos utilizar funciones Lambda para IoT (Internet de las cosas). Si tenemos por ahí un Alexa, podemos tener "palabras clave" para activar nuestro Alexa con Un Skill y que utilice nuestra aplicación.

Las funciones Lambda son muy importantes en este entorno Cloud, junto a EC2 es la clave de todo.

Para hacernos una idea, aquí es dónde desplegaremos nuestras Apis.

Las funciones Lambda para ser desplegadas necesitan de un usuario con un Role determinado.

Tenemos un millón de peticiones a Lambda gratuitas.

Las funciones contienen un **handler** que es el encargado de administrar que deseamos realizar en su interior.

Al igual que Azure Functions, podremos visualizar el código o no, todo depende del lenguaje a utilizar.

The screenshot shows the 'Create a function' wizard interface. At the top, there's a header with the Lambda logo and the text 'Lambda Ejecute código sin tener que pensar en los servidores'. Below it, the main title is 'Crear una función' with a 'Información' link. A note says 'Seleccione una de las siguientes opciones para crear la función.' Three options are shown in boxes:

- Crear desde cero**
Empiece con un sencillo ejemplo "Hello World".
- Utilizar un proyecto**
Cree una aplicación Lambda utilizando un código de muestra y los ajustes de configuración predefinidos de casos de uso comunes.
- Imagen del código**
Seleccione una imagen de código que implementará para su función.

Below this is a section titled 'Información básica' with a 'Nombre de la función' field containing 'my-lambda-martes'. A note says 'El nombre de la función debe tener entre 1 y 64 caracteres, debe ser exclusivo de la región y no puede incluir espacios. Los caracteres válidos son guiones bajos (_).' Under 'Tiempo de ejecución' (Execution time), it says 'Elija el idioma para usar para escribir su función. Tenga en cuenta que el editor de código de la consola es compatible con solo Node.js, Python y' with a dropdown set to 'Node.js 22.x'.

At the bottom, there's a section titled '▼ Cambiar el rol de ejecución predeterminado' with a 'Rol de ejecución' note. It shows three options:

- Creación de un nuevo rol con permisos básicos de Lambda**
- Uso de un rol existente
- Creación de un nuevo rol desde la política de AWS templates

A note in a box says 'La creación de roles puede tardar unos minutos. No elimine el rol ni edite las políticas de confianza o de permisos de este rol.'

At the very bottom, a note says 'Lambda creará un rol de ejecución denominado my-lambda-martes-role-ck9n25nu, con permiso para cargar registros a Amazon CloudWatch Logs.'

Nos habrá creado una función de forma gráfica



Y tendremos también un código

```

JS index.mjs x
JS index.mjs > ...
1 export const handler = async (event) => {
2 // TODO implement
3 const response = {
4 statusCode: 200,
5 body: JSON.stringify('Hello from Lambda!'),
6 };
7 return response;
8 };
9

```

Código fuente | Información Cargar desde ▾

EXPLORER MY-LAMBDA-MARTES index.mjs Deploy (Ctrl+Shift+U) Test (Ctrl+Shift+I)

Esto que estamos visualizando es un código "tirado", no sirve de nada.

El código Lambda es solamente una parte de un conjunto.

Por un lado, tenemos el código y, por otro lado, cómo queremos llamar a dicho código.

A esa otra funcionalidad se le llama desencadenador.

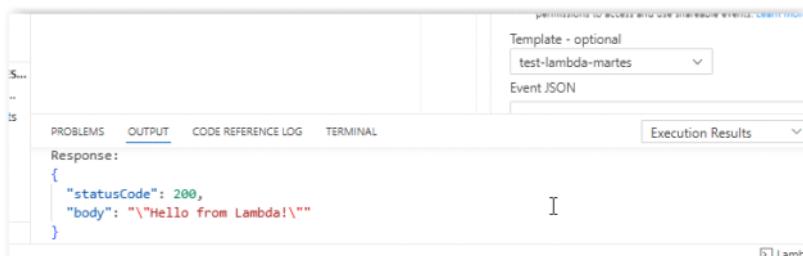
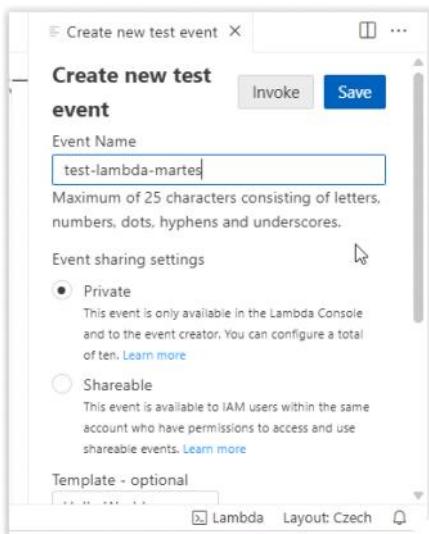
```

JS index.mjs x
JS index.mjs > ...
1 export const handler = ...
2 // TODO implement
3 const response = {
4 statusCode: 200,
5 body: JSON.stringify(
6 );
7 return response;
8 };
9

```

EXPLORER MY-LAMBDA-MARTES index.mjs Deploy (Ctrl+Shift+U) Test (Ctrl+Shift+I)

TEST EVENTS [NONE SELECTED]

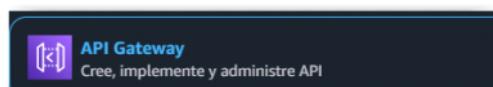


Como desencadenadores, podemos elegir un montón, incluso Alexa (si aparece)

TRIGGER

Vamos a crear un Trigger para realizar la petición a nuestro nuevo Lambda mediante HTTP..

Uno de los Triggers para peticiones HTTP se llama **Api Gateway**



Elija un tipo de API Información

API HTTP

Cree API REST rentables y de baja latencia con características integradas como OIDC y OAuth2, y compatibilidad nativa con CORS.

Funciona con lo siguiente:
Lambda, backends HTTP

[Importar](#) [Crear](#)

Detalles de la API

Nombre de API
Una API HTTP debe tener un nombre. El nombre es un valor que no es único y que se utiliza para identificar y organizar la programación, utilice el ID de API que API Gateway genera en su nombre.

Tipo de dirección IP Información
Seleccione el tipo de direcciones IP que pueden invocar el punto de conexión predeterminado de la API. No es necesario efectuar una selección.
 IPv4
Incluye solo direcciones IPv4.
 Doble pila
Incluye direcciones IPv4 e IPv6.

Integraciones (0) Información
Especifique los servicios de backend con los que se comunicará la API. Se denominan integraciones. Para una integración, responda con la respuesta de la función. Para una integración HTTP, API Gateway envía la solicitud a la URL especificada.

[Agregar integración](#)

responde con la respuesta de la función. Para una integración HTTP,

[Agregar integración](#)

Integraciones (1) Información

Especifique los servicios de backend con los que se comunicará la API. Se denominan integraciones. Para una integración de Lambda, API Gateway invoca la función de Lambda y responde con la respuesta de la función. Para una integración HTTP, API Gateway envía la solicitud a la URL especificada y devuelve la respuesta de la URL.

Lambda

Región de AWS

us-east-1

Función de Lambda

Elegir una función de Lambda o escriba su ARN

Versión

[Learn more](#)

2.0

[Eliminar](#)

[Agregar integración](#)

arn:aws:lambda:us-east-1:772056227005:function:my-lambda-martes



Configurar rutas - *opcional*

Configurar rutas Información

API Gateway usa rutas para exponer las integraciones a los clientes de su API. Para las API HTTP, las rutas constan de dos partes: un método HTTP (p. ej., GET /pets) y una ruta de recurso. Puede definir métodos HTTP específicos para su integración (GET, POST, PUT, PATCH, HEAD, OPTIONS y DELETE) o utilizar el método ANY para que coincida con todos los métodos que no haya definido en un recurso determinado.

Método

ANY

Ruta de recurso

/my-lambda-martes

Destino de integración

my-lambda-martes

[Eliminar](#)

[Agregar ruta](#)

[Cancelar](#)

[Revisar y crear](#)

[Anterior](#)

[Siguiente](#)

Una etapa son los entornos de configuración de producción para los despliegues.

De hecho, tendremos varios entornos (desarrollo y producción) para desplegar nuestras APIs

Por último, debemos enlazar nuestra función Lambda con API Gateway

[Lambda](#) > [Añadir desencadenadores](#)

El evento de prueba "test-lambda-martes" se ha guardado correctamente.



API Gateway

aws api application-services backend HTTP REST serverless

Agregue una API a su función de Lambda para crear un punto de enlace HTTP que invoque la función. API Gateway admite REST, HTTP WebSocket. [Más información](#)

Intención

Utilice una API existente o haga que cree una automáticamente.

Crear una API nueva

API HTTP

api-gateway-martes

4am9vluah



se crean recursos basados en el nombre de la función de Lambda y agrega el r

Introduzca el nombre o el ID de la API

Lambda asignará los permisos necesarios para Amazon API Gateway invocar la función Lambda desde este desencadenador. [Otro](#)

Lambda > Añadir desencadenadores

El evento de prueba "test-lambda-martes" se ha guardado correctamente.

Etapa de implementación
Seleccione la etapa de la API en la que Lambda agrega el desencadenador. Puede usar el historial de implementación de la consola de API Gateway para reimplementación activa de esta etapa.

\$default  

Seguridad
Configure el mecanismo de seguridad del punto de conexión de la API.

Abrir 

Lambda añadirá los permisos necesarios para Amazon API Gateway para invocar la función Lambda desde este desencadenador. [Obtener información](#) sobre el modelo de permisos de Lambda.

Y nos ha generado una URL asociada a nuestro Api Gateway

Código | Probar | Monitorear | **Configuración** | Alias | Versiones

Configuración general

Desencadenadores (1) [Información](#)

 Corregir errores  Editar  Eliminar  Agregar desencadenador

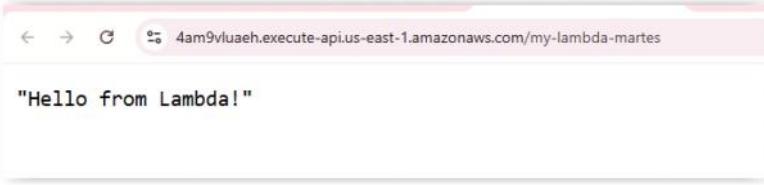
Buscar desencadenadores  1 < 1 >

Desencadenador

 API Gateway: api-gateway-martes
arn:aws:execute-api:us-east-1:772056227005:4am9vluah/*/*my-lambda-martes
Punto de enlace de API: <https://4am9vluah.execute-api.us-east-1.amazonaws.com/my-lambda-martes>

 Detalles

Y ya tendremos todo configurado para la petición



"Hello from Lambda!"

Vamos a realizar una función Lambda utilizando Visual Studio.

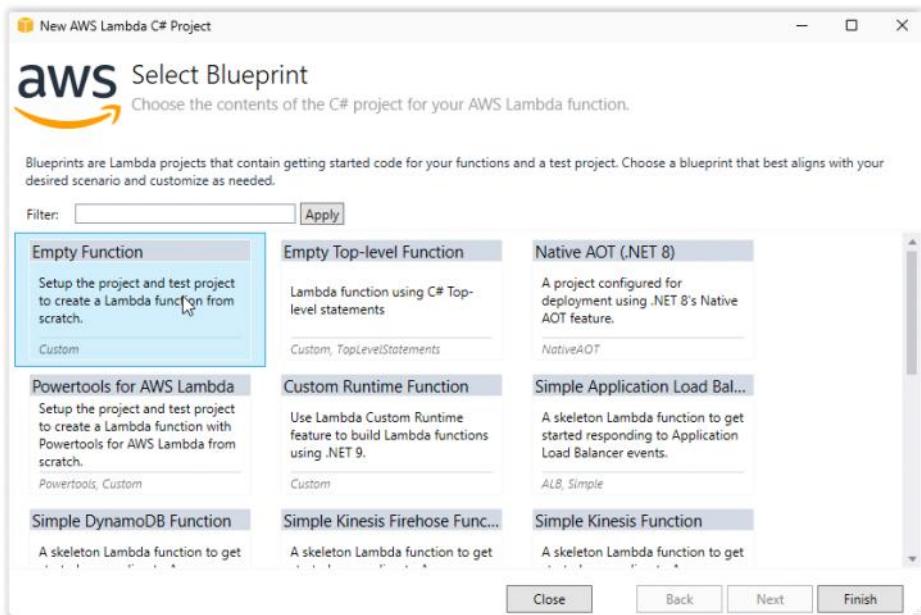
Nota: Necesitamos las herramientas de AWS para Visual Studio.

El tipo de proyecto es **AWS Lambda Project**

El nombre del proyecto será **MyFunctionLambdaNetCore**

 AWS Lambda Project (.NET Core - C#)
A C# project for creating AWS Lambda Functions using .NET Core.

C# AWS Cloud Serverless



Tendremos el código de la función para ejecutar con un Handler

```
public string FunctionHandler(string input, ILambdaContext context)
{
    return input.ToUpper();
}
```

Nuestra función tiene dos argumentos:

- 1) El contenido que recibirá la función (**string**).
- 2) El contexto que contiene información sobre el entorno. Podría ser importante para acceder a Elementos de la función (ficheros) una vez publicada en la nube

Para ejecutarlo, podemos hacerlo directamente con **cmdler** o con **F5**

Con **F5** solamente funcionará con la página de **Mock** si no recibe argumentos **string**

AWS .NET 8.0 Mock Lambda Test Tool

Run .NET Lambda function code inside this tool. IDEs can attach their debuggers to this tool and step through the Lambda code.

If you are developing .NET Lambda function using **custom runtimes** or **C# top level statements** that use the **Amazon.Lambda.RuntimeSupport** NuGet package the [Executable Assembly](#) page should be used to test the function.

Config File	Function
aws-lambda-tools-defaults.json	MyFunctionLambdaNetCore::MyFunctionLam
AWS Credential Profile	AWS Region
user-visual-studio	us-east-1
Example Requests: -- select a request --	
Function Input:	
Example Requests: API Gateway V2 HTTP API	

Veremos que tenemos un error porque estamos enviando JSON y recibimos string y no se entera

```
execution.  
System.Exception: Error deserializing the input JSON  
to type String  
at  
Amazon.Lambda.TestTool.Runtime.LambdaExecutor.BuildPar  
request, ILambdaContext context) in  
C:\build\Tools\LambdaTestTool\src\Amazon.Lambda.TestTo  
258  
at
```

Function Input:

```
"Datos para la función"
```

Execute Function Save Request

Response:

The area below shows the result returned by your function execution.

```
"Hoy es miércoles!!!!!!, Sr/Sra Alumno"
```

El siguiente paso es publicar nuestra función.

Para poder publicar una función NO se utiliza el usuario que tenemos en Visual Studio, son elementos distintos.

Necesitamos un usuario con Permisos sobre **Lambda** para agregarlo a nuestra consola de cmdere mediante **AWS Configure**

Abriremos **IAM** y creamos un nuevo grupo llamado **grupo-lambda** y un usuario llamado **user-lambda**

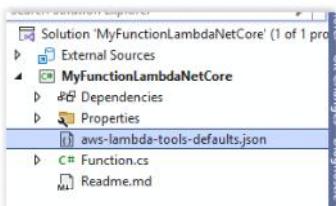


```
C:\Users\Profesor MCSD Mañana  
\ aws configure  
AWS Access Key ID [*****8HMN]: AKIA3HQRRDS67EDSKAFQ  
AWS Secret Access Key [*****5u7G]: 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  
Default region name [us-east-1]:  
Default output format [json]:
```

El siguiente paso que necesitamos es un **Role** para poder desplegar la aplicación.

Por un lado está el usuario (quién) y por otro, el Role (como)

Dentro del proyecto, tenemos un fichero JSON que indicará cómo se desplegará nuestra función



En dicho fichero debemos indicar el Role de ejecución en la nube, una vez publicada.

Vamos a crear un Role para la ejecución de la función.

Seleccionar entidad de confianza Información

Tipo de entidad de confianza

Servicio de AWS
Permita que servicios de AWS como EC2, Lambda u otros realicen acciones en esta cuenta.

Cuenta de AWS
Permitir a las entidades de otras cuentas de AWS que le pertenezcan a usted o a un tercero realizar acciones en esta cuenta.

Identidad web
Permite a las personas federadas por el proveedor de identidad web externo especificado asumir este rol para realizar acciones en esta cuenta.

Federación SAML 2.0
Permitir que las personas federadas con SAML 2.0 a partir de un directorio corporativo realicen acciones en esta cuenta.

Política de confianza personalizada
Cree una política de confianza personalizada para permitir que otras personas realicen acciones en esta cuenta.

Caso de uso

Permita que un servicio de AWS, como EC2, Lambda u otros, realicen acciones en esta cuenta.

Servicio o caso de uso

Lambda

Elija un caso de uso para el servicio especificado.

Caso de uso

Lambda
Allows Lambda functions to call AWS services on your behalf.

- AWSDeepLensLambdaFunctionAccessPolicy
- AWSLambda_FullAccess
- AWSLambda_ReadOnlyAccess

Asignar nombre, revisar y crear

Detalles del rol

Nombre del rol

Ingrrese un nombre significativo para identificar a este rol.

role-execution-lambda-function

64 Caracteres máximos. Utilice caracteres alfanuméricos y '+, @-_-^.

Descripción

Agregue una breve explicación para este rol.

Allows Lambda functions to call AWS services on your behalf.

Máximo de 1000 caracteres. Utilice letras (A-Z y a-z), números (0-9), tabulaciones, nuevas líneas o cualquiera de los siguientes

Dentro del fichero JSON de la aplicación, debemos indicar una nueva key llamada **function-role** apuntando a Nuestro Role recién creado

[role-execution-lambda-function](#)

```
    "function-architecture": "x86_64",
    "function-runtime": "dotnet8",
    "function-memory-size": 512,
    "function-timeout": 30,
    "function-handler": "MyFunctionLambdaNetCore::MyFunctionLambda",
    "function-role": "role-execution-lambda-function"
}
```

El siguiente paso es publicar nuestra aplicación.

Para ello, abrimos **cmdler** y entramos en la ruta de nuestra función.

Debemos instalar la siguiente herramienta para publicar funciones:

```
C:\Users\Profesor MCSD Mañana\Documents\Proyectos\MyFunctionLambdaNetCore
λ dotnet tool install -g Amazon.Lambda.Tools
λ dotnet lambda deploy-function my-function-net-core-miercoles
```

Subimos la función con el siguiente comando:

```
dotnet lambda deploy-function NOMBRE_FUNCION_EN_AWS
```

```
C:\Users\Profesor MCSD Mañana\Documents\Proyectos\MyFunctionLambdaNetCore
λ dotnet lambda deploy-function my-function-net-core-miercoles
```

Y tendremos la función publicada

Funciones (1)					
	Nombre de la función	Descripción	Tipo de paquete	Tiempo de ejecución	Última modificación
<input type="checkbox"/>	my-function-net-core-miercoles	-	Zip	.NET 8 (C#/F#/PowerShell)	hace 1 minuto

Agregamos un nuevo disparador de tipo Api Gateway HTTP

Agregar desencadenador

Configuración del desencadenador Información

API Gateway aws api application-services backend HTTP REST serverless

Agregue una API a su función de Lambda para crear un punto de enlace HTTP que invoque la función. API Gateway admite REST, HTTP y API de WebSocket. [Más información](#)

Intención
Utilice una API existente o haga que cree una automáticamente.

Crear una API nueva
 Usar una API existente

Tipo de API

- API HTTP**
Cree API REST rentables y de baja latencia con características integradas, como OIDC y OAuth2, así como compatibilidad nativa con CORS.
- API de WebSocket**
Cree una API de WebSocket con conexiones persistentes para casos de uso en tiempo real, como aplicaciones de chat o paneles.
- API de REST**
Desarrolle una API REST en la que obtenga control total sobre la solicitud y la respuesta, junto con las capacidades de administración de la API.

Seguridad
Configure el mecanismo de seguridad del punto de conexión de la API.

[Abrir](#)

Ajustes adicionales

Nombre de la API
Elija un nombre para la API. Los nombres de las API no tienen por qué ser únicos.

Etapa de implementación
Seleccione la etapa de la API en la que Lambda agrega el desencadenador. Puede usar el historial de implementación de la consola de implementación activa de esta etapa.

Uso compartido de recursos entre orígenes (CORS)
CORS es necesario para llamar a la API desde una página web que no está alojada en el mismo dominio. Esta opción permite el uso compartido de recursos entre orígenes (CORS) desde cualquier dominio al agregar el encabezado Access-Control-Allow-Origin a todas las respuestas.

Lo probamos con un Evento y listo

Ejecutando la función: sin errores ([registros](#))

▼ Detalles

"Hoy es miércoles!!!!!!, Sr/Sra Hola caracola"

Resumen

Código SHA-256 5Th9AjeEw/Oa3lbNm3F5kbbM/xWDe90xKBBgfhIUMQQ=	Tiempo de ejecución hace 1 minuto
Versión de la función \$LATEST	ID de solicitud 30169d7f-d81f-410f-9568-2035f
Duración 37.07 ms	Duración facturada 38 ms
Recursos configurados 512 MB	Memoria máx. utilizada 74 MB

Resultado de registro

Vamos a modificar nuestra función para recibir información en formato JSON

Recibiremos la información mediante un **Model**.

Creamos una carpeta llamada **Models** y una clase llamada **TestModel**

TESTMODEL

```
0 references
public class TestModel
{
    0 references
    public string Key1 { get; set; }
    0 references
    public string Key2 { get; set; }
    0 references
    public string Key3 { get; set; }
}
```

Utilizamos Newtonsoft

 **Newtonsoft.Json** by dotnetfoundation, jamesnk, newtonsoft, 6.25B download 13.0.3
Json.NET is a popular high-performance JSON framework for .NET

Modificamos el código de nuestra función

```
0 references
public string FunctionHandler(TestModel input, ILambdaContext context)
{
    //Devolvemos un JSON con la información
    var data = new
    {
        dato1 = input.Key1,
        dato2 = input.Key2,
        dato3 = input.Key3,
        mensaje = "Recibiendo información en Lambda!!!"
    };
    string json = JsonConvert.SerializeObject(data);
    return json;
}
```

Publicamos nuestro Lambda de nuevo

Una vez publicado, podemos visualizar la petición desde Api Gateway

Si utilizamos el Api Gateway ya podremos visualizar que funciona correctamente

```

{
  "dato1": null,
  "dato2": null,
  "dato3": null,
  "mensaje": "Recibiendo información en Lambda!!!"
}

```

Recibimos los datos NULL. Si lo probamos con un Evento, veremos que es funcional.

Probar evento [Información](#)

Para invocar la función sin guardar un evento, modifique el evento y luego elija Probar. Lambda utiliza el evento modificado para invocar la función, pero no sobrescribe el evento original hasta que elija Guardar.

Acción de evento de prueba

Crear un nuevo evento Editar evento guardado

Nombre del evento

evento-prueba

Evento JSON

```

1 * {
2   "key1": "Hola",
3   "key2": "Cara",
4   "key3": "Cola"
5 }

```

[Formato JSON](#)

Código | **Probar** | Monitorear | Configuración | Alias | Versiones

Ejecutando la función: sin errores ([registros](#))

Detalles

```
{"\"dato1\":\"Hola\",\"dato2\":\"Cara\",\"dato3\":\"Cola\",\"mensaje\":\"Recibiendo información en Lambda!!!\"}"
```

Resumen

Código SHA-256 5Th9AjeEw/Oa3lbNm3F5kbbM/xWDe90xKBBgfhIUMQQ=	Tiempo de ejecución hace 20 segundos
Versión de la función \$LATEST	ID de solicitud 6dc3c622-17c5-4fa5-9ef9-ebb9fe59723b
Duración 10.04 ms	Duración facturada 11 ms
Recursos configurados	Memoria máx. utilizada

LAMBDA GAMES

Creamos un Role con permisos sobre S3, Lambda y CloudWatch

Seleccionar entidad de confianza [Información](#)

Tipo de entidad de confianza

Servicio de AWS

Permita que servicios de AWS como EC2, Lambda u otros realicen acciones en esta cuenta.

Identidad web

Permite a las personas federadas por el proveedor de identidad web externo especificado asumir este rol para realizar acciones en esta cuenta.

Política de confianza personalizada

Cree una política de confianza personalizada para permitir que otras personas realicen acciones en esta cuenta.

Cuenta de AWS

Permitir a las entidades de otras cuentas de AWS que le pertenezcan a usted o a un tercero realizar acciones en esta cuenta.

Federación SAML 2.0

Permitir que las personas federadas con SAML 2.0 a partir de un directorio corporativo realicen acciones en esta cuenta.

Caso de uso

Permita que un servicio de AWS, como EC2, Lambda u otros, realicen acciones en esta cuenta.

Servicio o caso de uso

Lambda

Elija un caso de uso para el servicio especificado.

Caso de uso

Lambda

Allows Lambda functions to call AWS services on your behalf.

 AmazonS3FullAccess

Admir

 AWSLambda_FullAccess

Admir

 CloudWatchFullAccess

Adminis

Asignar nombre, revisar y crear

Detalles del rol

Nombre del rol

Ingrese un nombre significativo para identificar a este rol.

lambda-role-s3-games

64 Caracteres máximos. Utilice caracteres alfanuméricos y '+ =,_@-_-.'

Descripción

Agregue una breve explicación para este rol.

Allows Lambda functions to call AWS services on your behalf.

Máximo de 1000 caracteres. Utilice letras (A-Z y a-z), números (0-9), tabulaciones, nuevas líneas o cualquiera de los siguientes caracte

Información básica

Nombre de la función

Escriba un nombre para describir el propósito de la función.

lambda-game

El nombre de la función debe tener entre 1 y 64 caracteres, debe ser exclusivo de la región y no puede incluir espacios. Los caracteres válidos son a-z, A-Z, 0-9, guiones (-) y guiones bajos (_).

Tiempo de ejecución | [Información](#)

Elija el idioma para usar para escribir su función. Tenga en cuenta que el editor de código de la consola es compatible con solo Node.js, Python y Ruby.

Python 3.13



Arquitectura | [Información](#)

Elija la arquitectura del conjunto de instrucciones que desea para el código de la función.

arm64

x86_64

Permisos [Información](#)

De forma predeterminada, Lambda creará un rol de ejecución con permisos para cargar registros en Amazon CloudWatch Logs. Puede personalizar este rol predeterminado más adelante al agregar los disparadores.

Creamos un bucket llamado **bucket-games**

▼ Cambiar el rol de ejecución predeterminado

Rol de ejecución
Seleccione un rol que defina los permisos de la función. Para crear un rol personalizado, vaya a la [consola de IAM](#).

Creación de un nuevo rol con permisos básicos de Lambda
 Uso de un rol existente
 Creación de un nuevo rol desde la política de AWS templates

Rol existente
Seleccione un rol existente que haya creado para usarlo con esta función de Lambda. El rol debe tener permiso para cargar registros en Amazon CloudWatch Metrics.

lambda-role-s3-games

Consulte el rol [lambda-role-s3-games](#) en la consola de IAM.

El siguiente paso es jugar con nuestro Lambda. Nuestro desencadenador será subir un fichero CSV a S3

Incluimos un nuevo desencadenador S3

Agregar desencadenador

Configuración del desencadenador [Información](#)

Bucket
Seleccione o escriba el ARN de un bucket de S3 que actúa como origen de eventos. El bucket debe estar en la misma región que la función.

s3/bucket-comics-paco

Región del bucket: us-east-1

Tipos de eventos
Seleccione los eventos que desea que activen la función de Lambda. Si lo desea, también puede configurar un prefijo o un sufijo para un evento. Sin embargo, en cada bucket, no puede haber eventos individuales con configuraciones múltiples que tengan prefijos o sufijos superpuestos que puedan coincidir con la misma clave de objeto.

Todos los eventos de creación de objetos

Invocación recurrente
Si la función escribe objetos en un bucket de S3, asegúrese de utilizar diferentes buckets de S3 para las entradas y las salidas. Escribir en el mismo bucket aumenta el riesgo de crear una invocación recurrente, lo que puede resultar en un aumento del uso de Lambda y de los costos. [Más información](#)

Acepto que no se recomienda utilizar el mismo bucket de S3 para las entradas y las salidas, y que esta configuración puede provocar invocaciones recurrentes, así como también un aumento del uso de Lambda y de los costos.

Lambda añadirá los permisos necesarios para AWS S3 para invocar la función Lambda desde este desencadenador. [Obtenga más información](#) sobre el modelo de permisos de Lambda.

Escribimos el siguiente código

```
import json
import urllib.parse
import boto3
print('Loading function')
s3 = boto3.client('s3')
out_bucket = 'bucket-comics-paco '
def lambda_handler(event, context):
    # TODO implement
    print("Subiendo fichero")
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'])
    encoding='utf-8'
    print("Nombre del fichero: " + key)
    response = s3.get_object(Bucket=bucket, Key=key)
    print(response)
    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }
```

Posteriormente, realizamos un **Deploy** y vamos a visualizar las métricas para ver los registros de la función

Código | Probar | **Monitorear** | Configuración | Alias | Versiones

Monitor Información

Ver registros de CloudWatch | Ver Application Signals | Ver rastros de X-Ray | Ver Lambda Insights

Ver perfiles de CodeGuru

Filter metrics by Recurso: nombre de la función:\$LATEST

Investigate with AI - new 3h 1d 1sem. Zona horaria UTC Explorar contenido relacionado

CloudWatch metrics

Y subimos un nuevo fichero a nuestro bucket

Eventos de registro

Puede utilizar la barra de filtros a continuación para buscar y hacer coincidir términos, frases o valores en sus eventos de registro. [Más información sobre los patrones de filtro](#)

Filtrar eventos: pulse Intro para buscar 1m 1h Zona horaria UTC

Mostrar ▾

Marca temporal	Mensaje
2025-05-22T09:30:24.743Z	Loading function
2025-05-22T09:30:24.943Z	START RequestId: cedf921a-97e0-4c2f-ae6e-8d0309197363 Version: \$LATEST
2025-05-22T09:30:24.943Z	Subiendo fichero
2025-05-22T09:30:24.944Z	Nombre del fichero: 2.jpg
2025-05-22T09:30:25.217Z	{'ResponseMetadata': {'RequestId': '8CYNAS1KVQE03PZG', 'HostId': 'xHHIAAnUkFNX7o0P3XphpoPktio+85wk0DHpSh...', 'HTTPStatusCode': 200, 'HTTPHeaders': {'Content-Type': 'application/json', 'Content-Length': '113', 'Date': 'Tue, 22 May 2025 09:30:25 UTC', 'X-Amzn-Request-Id': 'cedf921a-97e0-4c2f-ae6e-8d0309197363', 'X-Amzn-Trace-Id': 'Root=1-5e0a1a1f-100000000000000000000000; Sampled=true; Segment=1; TraceId=1-5e0a1a1f-100000000000000000000000'}, 'RequestId': 'cedf921a-97e0-4c2f-ae6e-8d0309197363', 'HostId': 'xHHIAAnUkFNX7o0P3XphpoPktio+85wk0DHpSh...', 'HTTPStatusCode': 200, 'HTTPHeaders': {'Content-Type': 'application/json', 'Content-Length': '113', 'Date': 'Tue, 22 May 2025 09:30:25 UTC', 'X-Amzn-Request-Id': 'cedf921a-97e0-4c2f-ae6e-8d0309197363', 'X-Amzn-Trace-Id': 'Root=1-5e0a1a1f-100000000000000000000000; Sampled=true; Segment=1; TraceId=1-5e0a1a1f-100000000000000000000000'}, 'HTTPHeaders': {'Content-Type': 'application/json', 'Content-Length': '113', 'Date': 'Tue, 22 May 2025 09:30:25 UTC', 'X-Amzn-Request-Id': 'cedf921a-97e0-4c2f-ae6e-8d0309197363', 'X-Amzn-Trace-Id': 'Root=1-5e0a1a1f-100000000000000000000000; Sampled=true; Segment=1; TraceId=1-5e0a1a1f-100000000000000000000000'}, 'HTTPHeaders': {'Content-Type': 'application/json', 'Content-Length': '113', 'Date': 'Tue, 22 May 2025 09:30:25 UTC', 'X-Amzn-Request-Id': 'cedf921a-97e0-4c2f-ae6e-8d0309197363', 'X-Amzn-Trace-Id': 'Root=1-5e0a1a1f-100000000000000000000000; Sampled=true; Segment=1; TraceId=1-5e0a1a1f-100000000000000000000000'}}, END RequestId: cedf921a-97e0-4c2f-ae6e-8d0309197363
2025-05-22T09:30:25.227Z	REPORT RequestId: cedf921a-97e0-4c2f-ae6e-8d0309197363 Duration: 283.12 ms Billed Duration: 284 ms Memory...
2025-05-22T09:30:25.227Z	No hay eventos recientes en este momento. Reintento automático en pausa. Reanudar

Volver arriba ^

Una vez que hemos visto que es funcional, vamos a jugar...

Lo que haremos será subir un fichero CSV, analizar los datos con Pandas y generar un gráfico con dichos datos

Para poder utilizar librerías "externas" dentro de funciones, debemos utilizar el siguiente contenido

<https://pypi.org/>

pandas 2.2.3

pip install pandas

Powerful data structures for data analysis, time series, and statistics

Navegación Descripción de proyecto

- Descripción de proyecto
- Histórico de versiones
- Archivos de descarga



pytz 2025.2

pip install pytz



Pub

World timezone definitions, modern and historical

Navegación

≡ Descripción de proyecto

⌚ Histórico de versiones

⬇ Archivos de descarga

Descripción de proyecto

Author: Stuart Bishop <stuart@stuartbishop.net>

Introduction

numpy 2.2.6

pip install numpy



Pub

Fundamental package for array computing in Python

Navegación

≡ Descripción de proyecto

⌚ Histórico de versiones

⬇ Archivos de descarga



Descripción de proyecto

matplotlib 3.10.3

pip install matplotlib



Pub

Python plotting package

Navegación

≡ Descripción de proyecto

⌚ Histórico de versiones

⬇ Archivos de descarga

Descripción de proyecto

pypi v3.10.3 conda-forge v3.10.3 downloads 87M/month powered by NumFOCUS

help forum discourse chat on gitter issue tracking github PR Welcome

Tests passing Azure Pipelines succeeded build unknown codecov 90% version scheme EPEI

Extraemos los ficheros y hacemos un ZIP con ellos

El siguiente paso es agregar una Capa para las librerías

Lambda > Funciones > lambda-game

lambda-game

[Limitación](#) [Copiar ARN](#)

Información general de la función [Información](#)

[Exportar a Infrastructure Composer](#) [Descargar](#)

[Diagrama](#) | [Plantilla](#)

lambda-game

S3

+ Agregar destino

+ Agregar desencadenador

Descripción -

Última actualización hace 1 hora

ARN de la función arn:aws:lambda:eu-west-1:7720005:function:lambda-game

URL de la función

Nombre: pandas

Descripción - Opcional: Librerías de pandas

Cargar un archivo .zip Cargar un archivo de Amazon S3

[Elegir archivo](#)

python.zip 33.94 MB

arm64 x86_64

x86_64 X

Una vez creada la capa, es el momento de asociar la capa con nuestra función

lambda-game

[Limitación](#)

Información general de la función [Información](#) [Exportar a Infrastructure Composer](#)

[Diagrama](#) | [Plantilla](#)

lambda-game

Layers (1)

S3

+ Agregar destino

+ Agregar desencadenador

Código | Probar | Monitorear | **Configuración** | Alias | Versiones

▶ Cifrado con clave KMS administrada por el cliente de AWS KMS [Información](#)

Configuración del tiempo de ejecución [Información](#)

[Editar](#) [Editar la configuración de administración del tiempo de ejecución](#)

Tiempo de ejecución Python 3.13	Controlador Información lambda_function.lambda_handler	Arquitectura Información x86_64
------------------------------------	---	--

▶ Configuración de administración del tiempo de ejecución

Capas [Información](#)

[Editar](#) [Añadir una capa](#)

Orden de combinación	Nombre	Versión de la capa	Tiempos de ejecución compatibles	Arquitecturas compatibles	ARN de la capa
No hay datos que mostrar.					

Elija una capa

Fuente de capa | [Información](#)
Elija entre capas con una versión ejecutable compatible y una arquitectura de conjunto de instrucciones o especifique el nombre de recurso de Amazon (ARN) de una versión de capa. También puede [crear una nueva capa](#).

Capas de AWS
Elige una capa de una lista de capas proporcionadas por AWS.

Capas personalizadas
Choose a layer from a list of layers created by your AWS account.

Especificar un ARN
Proporcione el ARN y especifique una capa.

Capas personalizadas
Layers created by your AWS account that are compatible with your function's runtime.

Versión

Una vez que hemos visto que el proyecto compila correctamente, es el momento de implementar el código

```
import json
import urllib.parse
import boto3
from io import StringIO, BytesIO
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
print('Loading function')
s3 = boto3.client('s3')
out_bucket = 'bucket-comics-paco'
def lambda_handler(event, context):
    # get bucket and key of file that caused trigger
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'], encoding='utf-8')
    try:
        # open data in CSV
        response = s3.get_object(Bucket=bucket, Key=key)

        csv_data = response['Body'].read().decode('utf-8')
        # convert data from string to list then list of lists
        dl = csv_data.split("\n")
        dll = [x.replace(',', '').replace('\r', '').split(',') for x in dl if len(x) > 0]
        # convert list of lists into dataframe
        data_df = pd.DataFrame(dll[1:], columns = dll[0])

        # get name and data of first column
        col_name = data_df.columns[0]
        data = data_df[col_name]
        data = np.array(data.to_list(), dtype=np.float64)
        # generate S3 key for resulting plot
        out_key = key.split('/')[-1].split('.')[0] + '.png'

        # verify clean plot
        plt.cla()
        plt.clf()
        # plot data
        img_data = BytesIO()
        plt.plot(data, label=col_name)
        plt.legend(loc="upper left")

        plt.savefig(img_data, format='png')
        img_data.seek(0)

        # put plot in S3 bucket
        bucket = boto3.resource('s3').Bucket(out_bucket)
        bucket.put_object(Body=img_data, ContentType='image/png', Key=out_key)

        # generate presigned url
        url = s3.generate_presigned_url('get_object',
            Params={'Bucket': out_bucket, 'Key': out_key},
```

```

    ExpiresIn=86400)

# log url in CloudWatch
print(url)

return "DONE"

except Exception as e:
    print(e)
    print('Error getting object {} from bucket {}'.format(key, bucket))
    raise e

```

Debemos modificar la configuración para ampliar el tamaño de memoria y el tiempo de duración para generar la imagen

Entramos en Configuración

The screenshot shows the AWS Lambda function configuration interface. It includes sections for 'Descripción - Opcional' (Description - Optional), 'Memoria' (Memory) set to 1024 MB, 'Almacenamiento efímero' (Temporary storage) set to 'None', and 'Tiempo de espera' (Execution timeout) set to 1 minute. The 'Snapstart' section notes that it reduces the start-up time by caching the function's code. The 'Tiempos de ejecución admitidos' (Supported execution times) section lists .NET 8, C#/F#/PowerShell, Java 11, Java 17, Java 21, Python 3.12, and Python 3.13.

CODIGO GOT

```

import json
import urllib.parse
import boto3
from io import StringIO, BytesIO
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
print('Loading function')
s3 = boto3.client('s3')
out_bucket = 'bucket-comics-paco'
def lambda_handler(event, context):
    # get bucket and key of file that caused trigger
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'], encoding='utf-8')
    try:
        # open data in CSV
        response = s3.get_object(Bucket=bucket, Key=key)
        url = f'https://{out_bucket}.s3.us-east-1.amazonaws.com/{key}'
        #https://bucket-comics-paco.s3.us-east-1.amazonaws.com/got.csv
        print(url)
        #url_bucket = "https://bucket-comics-paco.s3.us-
east-1.amazonaws.com/raw_data.png"
        #csv_data = response['Body'].read().decode('utf-8')
        df = pd.read_csv(url)
        print("leyendo...")
        df_killer = df.groupby('killer')['death_no'].agg(['count'])
        print("killers")
        df_primeros = df_killer.head(10)
        print("primeros")
        # generate S3 key for resulting plot
        out_key = key.split('/')[-1].split('.')[0] + '.png'

        #data = df_primeros.groupby('killer').sum().plot(kind='pie',
y='count')
        data = df_primeros.groupby('killer').sum()
        print("data")
        # verify clean plot
        plt.cla()
        plt.clf()

        # plot data
        img_data = BytesIO()
        #df_primeros.groupby('killer').sum().plot(kind='pie', y='count')
        plt.pie(x=data["count"], labels=data.index)
        # plt.pie(x=data["killer"], labels=data.count)
        # plt.pie(data, labels = mylabels)
        # plt.show()
        # plt.plot(data, label=col_name)
        # plt.legend(loc="upper left")
        print("graficos")
        plt.savefig(img_data, format='png')
        img_data.seek(0)

        # put plot in S3 bucket
        bucket = boto3.resource('s3').Bucket(out_bucket)
        bucket.put_object(Body=img_data, ContentType='image/png', Key=out_key)
    
```

```
# generate presigned url
url = s3.generate_presigned_url('get_object',
    Params={'Bucket': out_bucket, 'Key': out_key},
    ExpiresIn=86400)

# log url in CloudWatch
print(url)

return "DONE"

except Exception as e:
    print(e)
    print('Error getting object {} from bucket {}.'.format(key, bucket))
    raise e
```

API GATEWAY

miércoles, 21 de mayo de 2025 10:58

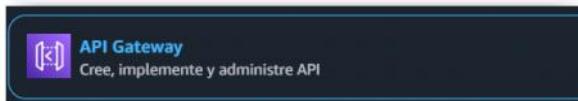
Api Gateway es un punto de entrada para acceder a nuestros códigos Lambda.

Genera EndPoints para poder realizar peticiones, es decir, los endpoint de cualquier Api que tengamos.

Hemos utilizado Api Gateway mediante HTTP, que nos genera solamente una URL para una función.

El siguiente paso es utilizar un Api Gateway REST que nos genera **endpoints** y nos permitirá llamar a nuestros Apis.

Para probar la primera teoría es utilizar el ejemplo de la función Lambda con tres Keys que acabamos de Publicar.



API REST

Desarrolle una API REST en la que obtenga control total de la solicitud y la respuesta, junto con las capacidades de administración de la API.

Funciona con lo siguiente:

Lambda, HTTP, servicios de AWS



Crear API de REST Información

Detalles de la API

Nueva API
Cree una API de REST nueva.

Clonar API existente
Cree una copia de una API en esta cuenta de AWS

Importar API
Importe una API desde una definición de OpenAPI.

API de ejemplo
Obtenga más información sobre API Gateway con

Nombre de API

my-first-api

Descripción: *opcional*

La API de REST 'my-first-api(x20jc2ono7)' se creó correctamente.

Recursos

Detalles del recurso

Ruta /

Métodos (0)

Tipo de método ▲ | Tipo de integración

No se han encontrado

No se han definido

Gateway de API

API

Nombres de dominio personalizados

Asociaciones de acceso a nombres de dominio

Enlaces de VPC

API: my-first-api

Recursos

Etapas

Autorizadores

Respuestas de puerta de enlace

Modelos

Política de recursos

Crear recurso

Detalles del recurso

Recurso de proxy [Información](#)

Los recursos de proxy gestionan las solicitudes a todos los subrecursos. Para crear un recurso de proxy, utilice un parámetro de ruta que termine con un signo más, por ejemplo {proxy+}.

Ruta de recurso

/

Nombre del recurso

testing

CORS (uso compartido de recursos entre orígenes) [Información](#)

Cree un método OPTIONS que permita todos los orígenes, todos los métodos y varios encabezados comunes.

[Cancelar](#)

[Crear recurso](#)

Creamos un nuevo método GET, POST, PUT o DELETE

Recursos

[Acciones de API](#)

[Implementar API](#)

[Crear recurso](#)

/

/testing

OPTIONS

Detalles del recurso

[Eliminar](#)

[Actualizar documentación](#)

[Habilitar CORS](#)

Ruta

/testing

ID de recurso

u4ohm9

Métodos (1)

[Eliminar](#)

[Crear método](#)

[OPTIONS](#)

Simulación

Ninguna

No obligatorio

Crear método

Detalles del método

Tipo de método

POST

Tipo de integración

Función de Lambda

Integre su API con una función de Lambda.



HTTP

Lleve a cabo la integración con un punto de conexión HTTP existente.



Simulación

Genere una respuesta basada en las asignaciones y transformaciones de API Gateway.



Servicio de AWS

Lleve a cabo la integración con un servicio de AWS.



Enlace de VPC

Lleve a cabo la integración con un recurso al que no se pueda acceder a través de la red pública de Internet.



Integración de proxy de Lambda

Envíe la solicitud a la función de Lambda como un evento estructurado.

Función de Lambda

Proporcione el nombre de la función de Lambda o un alias. También puede proporcionar un ARN de otra cuenta.

us-east-1

Elegir una función de Lambda o escriba su ARN

arn:aws:lambda:us-east-1:772056227005:function:my-function-net-core-miercoles

Otorgue permiso a API Gateway para invocar la función de Lambda

Al guardar los cambios, API Gateway actualiza la política basada en recursos de la función de Lambda para permitir que esta API la invoque.

Tiempo de espera de integración | [Información](#)

Por defecto, puede introducir un tiempo de espera de integración de 50 a 29 000 milisegundos. Puede usar las Service Quotas para aumentar el tiempo de espera de integra

29000

► Configuración de solicitud de método

► Parámetros de cadenas de consulta de URL

► Encabezados de solicitud HTTP

▼ Cuerpo de la solicitud

Tipo de contenido

application/json

Modelo

Empty



Eliminar

Agregar modelo

Y tendremos múltiples métodos para las llamadas

Recursos

Acciones de API ▾

Implementar API

Crear recurso

/
 / /testing

GET

OPTIONS

POST

/testing - GET - Ejecución de método

Actualizar documentación

Eliminar

ARN

arn:aws:execute-api:us-east-1:772056227005:x20jc2ono7/*:GET/testing

ID de recurso

u4ohm9



Solicitud de
método



Solicitud de
integración



Respuesta de
método



Respuesta de
integración



Para poder utilizar nuestro nuevo API necesitamos implementar el API en Producción dentro de un Entorno.

El entorno nos genera unas URLs de acceso a nuestros métodos.

Recursos

Acciones de API ▾

Implementar API

Crear recurso

/
 / /testing

GET

OPTIONS

POST

/testing - GET - Ejecución de método

Actualizar documentación

Eliminar

ARN

arn:aws:execute-api:us-east-1:772056227005:x20jc2ono7/*:GET/testing

ID de recurso

u4ohm9



Solicitud de
método



Solicitud de
integración



Respuesta de
método



Respuesta de
integración



Deploy API

Cree o seleccione una etapa en la que se implementará la API. Puede utilizar el historial de implementaciones para revertir o cambiar la implementación activa de una etapa. [Learn more](#)

Etapa
Nueva etapa

Nombre de etapa
dev

ⓘ Se creará una nueva etapa con los ajustes predeterminados. Edite la configuración de la etapa en la página **Etapa**.

Descripción de la implementación
Una nueva implementación de Testing en Development

Cancelar **Implementación**

Una vez creado, ya podremos probar y jugar.

Crear recurso

- /
- /testing
 - GET
 - OPTIONS
 - POST**

1:772056227005:x20jc2ono7/*/POST/testing

Flows:

- Client → Method Request
- Method Request → Integration Request
- Integration Request → Lambda Integration
- Lambda Integration → Response of Integration
- Response of Integration ← Response of Method
- Response of Method ← Integration Response

Buttons at the bottom: integración | Respuesta de integración | Respuesta de método | **Pruebas** | >

Método de pruebas
Realice una llamada de prueba al método. Cuando realiza una llamada de prueba, API Gateway omite la autorización e invoca directamente el método.

Cadenas de consulta
param1=value1¶m2=value2

Crear recurso

- /
- /testing
 - GET
 - OPTIONS
 - POST**

Cuerpo de la solicitud

```

1 [
2   "key1": "Prueba1",
3   "key2": "Prueba2",
4   "key3": "Prueba3333"
5 ]
    
```

Buttons:

0 0 5:2 JSON Espacios: 4

Pruebas

Y podremos invocar a nuestro Api

The screenshot shows the AWS Lambda Test API interface. On the left, under 'Crear recurso', there's a tree view with a root node '/' having a child '/testing'. Under '/testing', there are four methods: GET, OPTIONS, and POST (highlighted in green). On the right, under 'Pruebas', there's a test result for a POST request to '/testing'. The result shows the following details:

- Solicitud:** /testing
- Latencia en ms:** 55
- Estado:** 200
- Cuerpo de respuesta:**

```
{
    "dato1": "Prueba1",
    "dato2": "Prueba2",
    "dato3": "Prueba3333",
    "mensaje": "Recibiendo información en Lambda!!!"
}
```
- Encabezados de respuesta:**

```
{
    "Content-Type": "application/json",
    "X-Amzn-Trace-Id": "Root=1-682d9a16-dd9e28948c423954facea0d9;Parent=6f710035277d2903;Sampled=0;Lineage=1:f9e29e29:0"
}
```

Eliminamos Etapa, Recursos y Api Gateway y Función Lambda

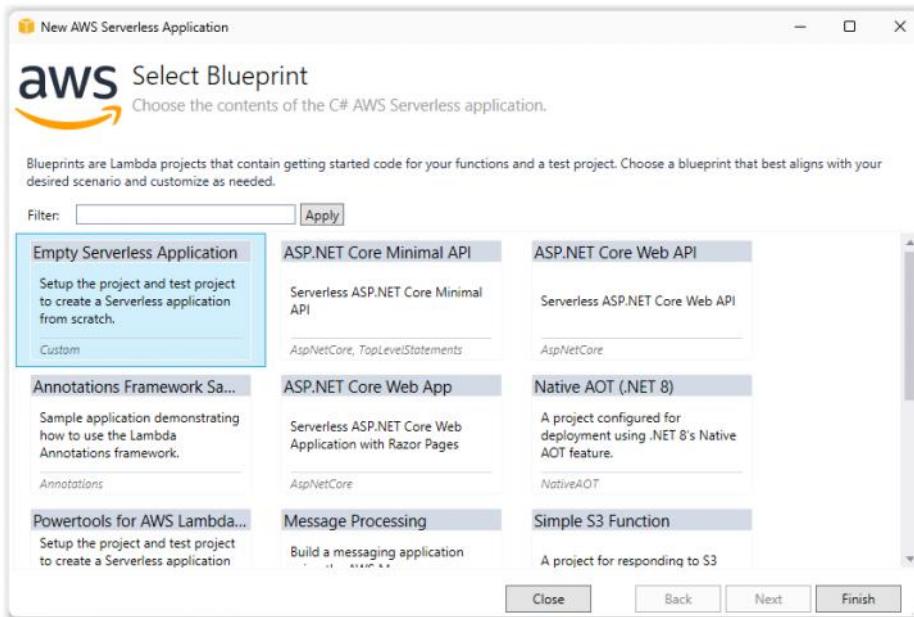
API GATEWAY REST

Al utilizar un Api Gateway estamos llamando a Lambda en el código.

Pero lo que nos interesa a nosotros es visualizar cómo podemos enviar nuestros Apis a un Lambda y Con Api Gateway.

Dichas Apis, para ser desplegadas no se hace con proyectos de AWS Lambda puros, tal y como hemos visto hoy.
Se hacen con proyectos de tipo **Serverless**

Vamos a crear un nuevo proyecto de tipo **AWS Serverless** llamado **ApiAWSPersonas**



Creamos una nueva carpeta llamada **Models** y una clase llamada **Persona**

PERSONA

```

0 references
public class Persona
{
    0 references
    public int IdPersona { get; set; }
    0 references
    public string Nombre { get; set; }
    0 references
    public string Email { get; set; }
    0 references
    public int Edad { get; set; }
}

```

Agregamos el Nuget de NewtonSoft



Tenemos una clase llamada **Function1** que es la encargada de trabajar con los métodos de nuestro Api.

En el constructor vamos a crear una serie de personas de Testing.

FUNCTION1

```

public class Functions
{
    public List<Persona> personasList;
    /// <summary>
    /// Default constructor that Lambda will invoke.
    /// </summary>
    public Functions()
    {
        this.personasList = new List<Persona>();
        Persona p = new Persona
        {
            IdPersona = 1, Nombre = "Adrian", Email="adri@gmail.com", Edad = 2
5
        };
        this.personasList.Add(p);
        p = new Persona
        {
            IdPersona = 2,
            Nombre = "Lucia",
            Email = "lucia@gmail.com",
            Edad = 22
        };
        this.personasList.Add(p);
        p = new Persona
        {
            IdPersona = 3,
            Nombre = "Antonia",
            Email = "antonia@gmail.com",
            Edad = 45
        };
        this.personasList.Add(p);
        p = new Persona
        {
            IdPersona = 4,
            Nombre = "Carlos",
            Email = "carlos@gmail.com",
            Edad = 27
        };
        this.personasList.Add(p);
    }

    /// <summary>
    /// A Lambda function to respond to HTTP Get methods from API Gateway
    /// </summary>
    /// <remarks>
    /// This uses the <see href="https://github.com/aws/aws-lambda-dotnet/blob/master/Libraries/src/Amazon.Lambda.Annotations/README.md">
    /// Lambda Annotations</see>
    /// programming model to bridge the gap between the Lambda programming model and a more idiomatic .NET model.
    ///
    /// This automatically handles reading parameters from an APIGatewayProxyRequest
    /// as well as syncing the function definitions to serverless.template each time you build.
    ///
    /// If you do not wish to use this model and need to manipulate the API Gateway
    /// objects directly, see the accompanying Readme.md for instructions.
    /// </remarks>
    /// <param name="context">
    Information about the invocation, function, and execution environment</param>
    /// <returns>
    The response as an implicit <see cref="APIGatewayProxyResponse"/></returns>
    [LambdaFunction]
    [RestApi(LambdaHttpMethod.Get, "/")]
    public IActionResult Get(ILambdaContext context)
    {
        context.Logger.LogInformation("Handling the 'Get' Request");
        string json = JsonConvert.SerializeObject(this.personasList);
        return Ok(json);
    }
}

```

Para poder publicar nuestro Api en un Gateway de Rest es necesario utilizar **CloudFormation**

Esta funcionalidad genera un Artifact (zip) dentro de un Bucket para poder desplegar nuestro Api Gateway en un Lambda

Necesitamos crear un Bucket para que CloudFormation sea capaz de desplegar el Artifact.

The screenshot shows the AWS S3 console with a bucket named "buckets-artifact-paco". The "Configuración general" tab is selected. Under "Tipo de bucket", "Uso general" is chosen (radio button selected). The bucket name is "buckets-artifact-paco". A note says: "Recomendado para la mayoría de los casos de uso y patrones de acceso. Los buckets de uso general son del tipo de bucket de S3 original. Permiten una combinación de clases de almacenamiento que almacenan objetos de forma redundante en múltiples zonas de disponibilidad." Under "Nombre del bucket", the name is "buckets-artifact-paco". A note says: "Los nombres de los buckets deben tener entre 3 y 63 caracteres y ser únicos dentro del espacio de nombres global. Los nombres de los buckets también deben empezar y terminar con a-z, 0-9, puntos (.) y guiones (-). [Más información](#)".

Publicamos nuestra aplicación indicando el nombre del **Stack** que será el nombre de nuestro Api.

Seleccionamos el nombre de nuestro nuevo **Bucket**

The dialog is titled "Publish AWS Serverless Application". It shows the "Profile" section with "AWS Credentials: Profile:user-visual-studio" and "Region: US East (N. Virginia)". Under "CloudFormation Settings", "Stack Name" is set to "ApiPersonasStack" and "S3 Bucket" is set to "buckets-artifact-paco". A checkbox "Save settings to aws-lambda-tools-defaults.json for future deployments." is checked. At the bottom are buttons for "Close", "Back", "Next", and "Publish".

Y nos generará una URL de producción

The screenshot shows the AWS CloudFormation console with a stack named "Stack: ApiPersonasStack". The status is "CREATE_COMPLETE". The "Description" field states: "An AWS Serverless Application. This template is partially managed by Amazon.Lambda.Annotations (v1.6.1.0)." Below it, the "AWS Serverless URL" is listed as "<https://lm1ex1y50m.execute-api.us-east-1.amazonaws.com/Prod>". The bottom navigation bar has tabs for "Events" and "Resources".

El siguiente paso es visualizar cómo incluir otro método dentro de la función y del Api Gateway

Tenemos un fichero llamado **serverless.template**

```

serverless.template Stack: ApiPersonasStack Functions.cs NuGet: ApiAWSPersonas Persona.cs
31     ],
32     "PackageType": "Zip",
33     "Events": {
34       "RootGet": {
35         "Type": "Api",
36         "Properties": {
37           "Path": "/",
38           "Method": "GET"
39         }
40       }
41     }
42   }
43 }
...

```

Por suerte, este fichero ya genera los endpoints de forma automática.

Agregamos otro método a nuestro Api

```

[LambdaFunction]
[RestApi(LambdaHttpMethod.Get, "/find")]
1 reference
public IHttpActionResult Find(ILambdaContext context)
{
    Persona persona = this.personasList[0];
    string json = JsonConvert.SerializeObject(persona);
    return HttpResults.Ok(json);
}

```

Publicamos para visualizar el nuevo contenido



Nos ha generado un nuevo método dentro de `serverless.template`

```

serverless.template Stack: ApiPersonasStack Functions.cs NuGet: ApiAWSPersonas Persona.cs
62     "Timeout": 30,
63     "Policies": [
64       "AWSLambdaBasicExecutionRole"
65     ],
66     "PackageType": "Zip",
67     "Handler": "ApiAWSPersonas::ApiAWSPersonas.Functions_Find_Generated"
68     "Events": {
69       "RootGet": {
70         "Type": "Api",
71         "Properties": {
72           "Path": "/find",
73           "Method": "GET"
74         }
75       }
76     }
77   }
78 }
79

```

Los parámetros se declaran igual que hicimos con cualquier Api tradicional: `{param}`

Deben ser el primer argumento, es decir, el último argumento siempre será el Context

Modificamos el método Find

```

[LambdaFunction]
[RestApi(LambdaHttpMethod.Get, "/find/{id}")]
1 reference
public IHttpResponse Find(int id, ILambdaContext context)
{
    Persona persona =
        this.personasList.FirstOrDefault(x => x.IdPersona == id);
    string json = JsonConvert.SerializeObject(persona);
    return HttpResults.Ok(json);
}

```

Y ya tenemos un maravilloso método con parámetros!!!!



El último paso es visualizar el comportamiento recibiendo objetos.

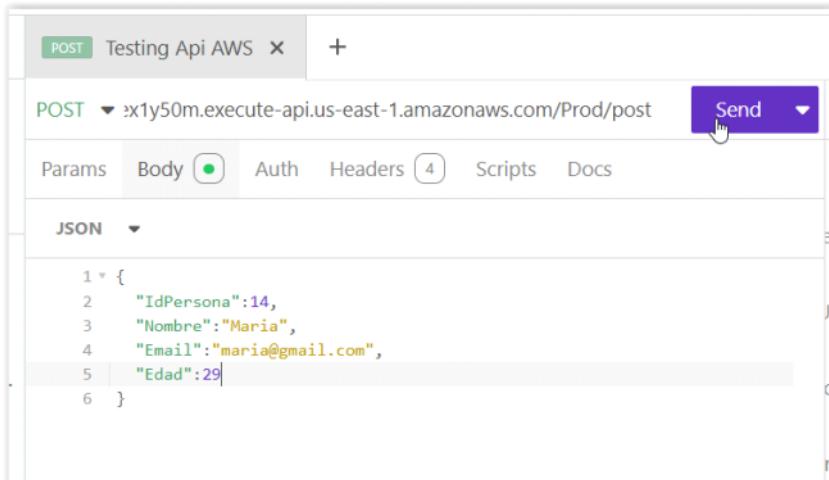
Vamos a crear un método POST para recibir un Model Persona y ver si funciona.

```

[LambdaFunction]
[RestApi(LambdaHttpMethod.Post, "/post")]
//PARA RECIBIR OBJETOS, DEBEMOS INCLUIR [FromBody]
//EN LA DECLARACION DEL METODO EN EL OBJETO
0 references
public IHttpResponse Post
    ([FromBody]Persona persona, ILambdaContext context)
{
    string json = JsonConvert.SerializeObject(persona);
    return HttpResults.Ok(json);
}

```

Y podremos comprobar que es funcional

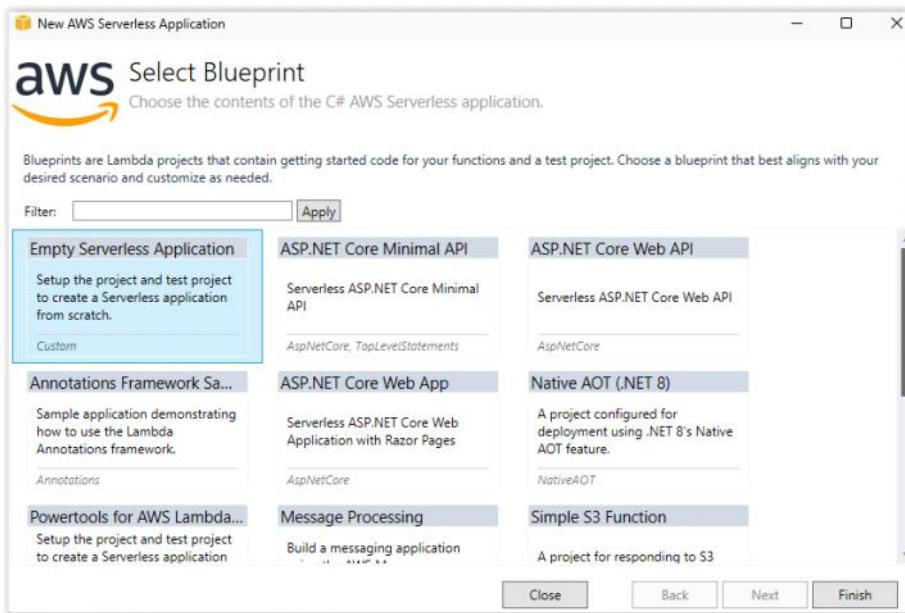


El siguiente paso es realizar un CRUD con un proyecto de base de datos real.

Iniciamos nuestra base de datos RDS

Creamos un nuevo proyecto de tipo AWS Serverless Application llamado ApiCrudAWSSeries





Agregamos los siguientes Nuget

The screenshot shows the NuGet Package Manager interface with the following packages listed:

- Microsoft.EntityFrameworkCore** by aspnet, dotnetframework, EntityFramework, Microsoft Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations. EF Core works with SQL Server, Azure SQL Database, SQLite...
- Microsoft.EntityFrameworkCore.Tools** by aspnet, dotnetframework, EntityFramework 9.0.5 Entity Framework Core Tools for the NuGet Package Manager Console in Visual Studio.
- MySql.EntityFrameworkCore** by MySQL, 9.1M downloads 9.0.3 MySql.EntityFrameworkCore adds support for Microsoft Entity Framework Core.
- Newtonsoft.Json** by dotnetfoundation, jamesnk, newtonsoft, 6.25B downloads 13.0.3 Json.NET is a popular high-performance JSON framework for .NET

Sobre **Models** creamos una nueva clase llamada **Serie**

SERIE

```
[Table("SERIES")]
0 references
public class Serie
{
    [Key]
    [Column("IDSERIE")]
    0 references
    public int IdSerie { get; set; }
    [Column("SERIE")]
    0 references
    public string Nombre { get; set; }
    [Column("IMAGEN")]
    0 references
    public string Imagen { get; set; }
    [Column("ANYO")]
    0 references
    public int Anyo { get; set; }
}
```

Creamos una carpeta llamada **Data** y una clase llamada **SeriesContext**

SERIESCONTEXT

```

public class SeriesContext : DbContext
{
    0 references
    public SeriesContext(DbContextOptions<SeriesContext> options)
        : base(options) { }

    0 references
    public DbSet<Serie> Series { get; set; }
}

```

Creamos una carpeta llamada **Repositories** y una clase llamada **RepositorySeries**

REPOSITORYSERIES

```

public class RepositorySeries
{
    private SeriesContext context;

    0 references
    public RepositorySeries(SeriesContext context)
    {
        this.context = context;
    }

    0 references
    public async Task<List<Serie>> GetSeriesAsync()
    {
        return await this.context.Series.ToListAsync();
    }
}

```

Sobre **Functions** implementamos el método GET

FUNCTIONS

```

public class Functions
{
    private RepositorySeries repo;

    public Functions(RepositorySeries repo)
    {
        this.repo = repo;
    }

    [LambdaFunction]
    [RestApi(LambdaHttpMethod.Get, "/")]
    public async Task<IHttpResponse> Get(ILambdaContext context)
    {
        context.Logger.LogInformation("Handling the 'Get' Request");
        List<Serie> series = await this.repo.GetSeriesAsync();
        return HttpResults.Ok(series);
    }
}

```

Necesitamos resolver las dependencias y nuestra cadena de conexión por algún "lugar"

awsmysqlpaco.c23akq6oglxj.us-east-1.rds.amazonaws.com

Las dependencias se resuelven en una clase llamada **Startup**

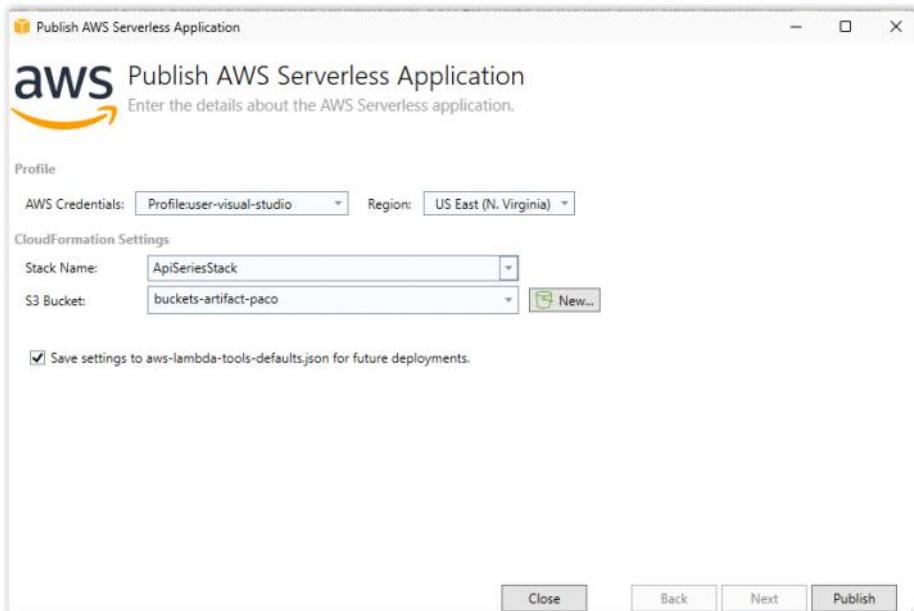
STARTUP

```

public class Startup
{
    1 reference
    public void ConfigureServices(IServiceCollection services)
    {
        services.AddTransient<RepositorySeries>();
        string connectionString = @"server=awsmysqlpaco.c23akq6oglxj.us-east-1.rds.
        services.AddDbContext<SeriesContext>
            (options => options.UseMySQL(connectionString));
        // Configure the TConfiguration here
    }
}

```

Publicamos cruzando mucho los dedos...



Iniciamos nuestra base de datos

Ponemos a nuestro **user-visual-studio** en nuestro grupo-visual

Nos quedamos con la implementación de un Api con Lambda y datos.

Intentamos poner nuestro proyecto en versión 9, el problema está en que la Máquina dónde está instalado el Lambda se ve que es la versión 8 de Net Core

Debemos utilizar dicha versión para nuestros Apis

Vamos a bajar nuestro Api a la versión 8 e intentar desplegarlo.

- 1) Desinstalamos los Nuget de la versión 9:
 - a. Dependency Injection, Entity y MySql

- 2) Volvemos a instalarlos en la versión 8 de Net Core.

En Local nos funciona, vamos a probar en AWS.

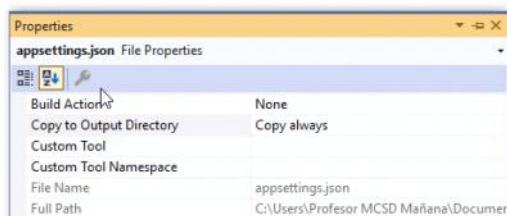
Publicamos nuestro nuevo Api.

Una vez que hemos visto que es funcional el Get, vamos a escribir el resto de Métodos dentro del Repository

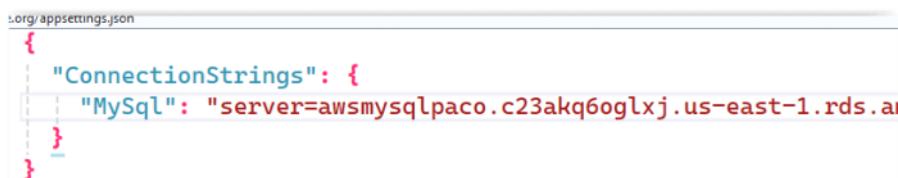
REPOSITORYSERIES

Vamos a poner nuestra cadena de conexión dentro de un Settings. Json

Creamos un nuevo fichero llamado **appsettings.json** y le decimos **Copy Always**



APPSETTINGS.JSON



Debemos instalar el siguiente Nuget para poder recuperar el fichero de configuración



Implementamos Startup

```
[LambdaStartup]
public class Startup
{
    public void ConfigureServices(IServiceCollection services)
```

```

{
    var builder = new ConfigurationBuilder()
        .SetBasePath(Directory.GetCurrentDirectory())
        .AddJsonFile("appsettings.json", true);

    var configuration = builder.Build();
    services.AddSingleton< IConfiguration>(configuration);

    services.AddTransient<RepositorySeries>();
    string connectionString =
        configuration.GetConnectionString("MySQL");
    services.AddDbContext< SeriesContext >
        (options => options.UseMySQL(connectionString));
}
}

```

El siguiente paso es implementar nuestros métodos de acción en el Controller

FUNCTIONS

```

public class Functions
{
    private RepositorySeries repo;

    public Functions(RepositorySeries repo)
    {
        this.repo = repo;
    }

    [LambdaFunction]
    [RestApi(LambdaHttpMethod.Get, "/")]
    public async Task<IHttpResult> Get(ILambdaContext context)
    {
        context.Logger.LogInformation("Handling the 'Get' Request");
        List< Serie > series = await this.repo.GetSeriesAsync();
        return HttpResults.Ok(series);
    }

    [LambdaFunction]
    [RestApi(LambdaHttpMethod.Get, "/find/{id}")]
    public async Task<IHttpResult> FindSerie(int id, ILambdaContext context)
    {
        Serie serie = await this.repo.FindSerieAsync(id);
        return HttpResults.Ok(serie);
    }

    [LambdaFunction]
    [RestApi(LambdaHttpMethod.Post, "/post")]
    public async Task<IHttpResult>
        Post([FromBody] Serie serie, ILambdaContext context)
    {
        await this.repo.CreateSerieAsync(serie.Nombre
            , serie.Imagen, serie.Año);
        return HttpResults.Ok();
    }

    [LambdaFunction]
    [RestApi(LambdaHttpMethod.Put, "/put/{id}")]
    public async Task<IHttpResult>
        Put(int id, [FromBody] Serie serie, ILambdaContext context)
    {
        await this.repo.UpdateSerieAsync(id, serie.Nombre
            , serie.Imagen, serie.Año);
        return HttpResults.Ok();
    }

    [LambdaFunction]
    [RestApi(LambdaHttpMethod.Delete, "/delete/{id}")]
    public async Task<IHttpResult> Delete(int id)
    {
        await this.repo.DeleteSerieAsync(id);
        return HttpResults.Ok();
    }
}

```

Publicamos el Api y comprobamos la funcionalidad con **Insomnia**

The screenshot shows the Insomnia REST Client interface. At the top, there's a navigation bar with 'Application', 'File', 'Edit', 'View', 'Window', 'Tools', and 'Help'. On the right, there's an 'Invite' button and an email address 'pacoserrano@gmail.com'. Below the navigation is a search bar with 'Search..'. The main area has tabs for 'GET Get Series', 'POST New Serie', 'PUT Update Serie', 'DEL Del Serie', and 'GET Find Serie'. The 'GET Get Series' tab is active, showing a successful response with status code 200 OK, 188 ms duration, and 1958 B size, timestamped 'Just Now'. The 'Params' tab is selected. The 'Preview' tab shows a JSON response:

```

1 [
2   {
3     "IdSerie": 1,
4     "Nombre": "Juego de tronos",
5     "Imagen": "https://cadenaer00.epimg.net/series/2019/05/23/television/1558591913_020782_1558595107_noticia_normal.jpg",
6     "Año": 2011
7   },
8   {
9     "IdSerie": 2,
10    "Nombre": "The Mandalorian",
11    "Imagen": "https://cadenaer00.epimg.net/series/2019/05/23/television/1558591913_020782_1558595107_noticia_normal.jpg",
12    "Año": 2011
13  }
14 ]

```

Below the preview, there's a placeholder 'Enter a URL and send to get a response' and a note 'Select a body type from above to send data in the body of a request'. On the left, there's a sidebar with icons for home, add environment, add cookies, add certificates, and a list of actions: 'Find Serie', 'Del Serie', 'Update Serie', 'New Serie', and 'Get Series' (which is currently selected).

API GATEWAY NET CORE

viernes, 23 de mayo de 2025 10:39

Como acabamos de visualizar, el coste de crear Apis es muy alto, no por dinero, Sino por desarrollo.
¿Dónde está Swagger/Scalar?

¿Cómo puedo diferenciar Controllers?

Esta forma está bien si vamos a "pensar" como en un Minimal Api.

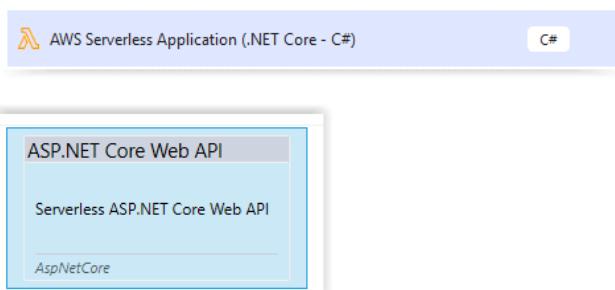
Cuando tenemos multitud de Controllers y Actions este proyecto no nos sirve.

Tenemos una forma de crear nuestros Apis de forma tradicional, es decir, Con Controllers y todo lo demás.

El proyecto será del mismo tipo **AWS Serverless Application**

Internamente, se utiliza un **Proxy** para las redirecciones en Api Gateway

Creamos un nuevo proyecto llamado **ApiAWSeriesCorrecto**



Como vemos, seguimos trabajando con Net Core 8

General

Output type
Specifies the type of application to build.
Console Application

Target framework ?
Specifies the version of .NET that the application targets. This option can have different values depending on which versions of .NET are installed on your computer.
.NET 8.0

[Install other frameworks](#)

Agregamos los siguientes Nuget:

Microsoft.EntityFrameworkCore by aspnet, dotnetframework 9.0.5
Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations...

Pomelo.EntityFrameworkCore.MySql by AmamiyaYuuko, bgrai 8.0.3
Pomelo's MySQL database provider for Entity Framework Core.

Sobre **Models** creamos una clase llamada **Serie**

SERIE

```

[Table("SERIES")]
0 references
public class Serie
{
    [Key]
    [Column("IDSERIE")]
    0 references
    public int IdSerie { get; set; }
    [Column("SERIE")]
    0 references
    public string Nombre { get; set; }
    [Column("IMAGEN")]
    0 references
    public string Imagen { get; set; }
    [Column("ANYO")]
    0 references
    public int Anyo { get; set; }
}

```

Sobre Data creamos una clase llamada SeriesContext

SERIESCONTEXT

```

public class SeriesContext : DbContext
{
    0 references
    public SeriesContext(DbContextOptions<SeriesContext> options)
        : base(options) { }

    0 references
    public DbSet<Serie> Series { get; set; }
}

```

Sobre Repositories creamos una nueva clase llamada RepositorySeries

REPOSITORYSERIES

```

public class RepositorySeries
{
    private SeriesContext context;

    public RepositorySeries(SeriesContext context)
    {
        this.context = context;
    }

    public async Task<List<Serie>> GetSeriesAsync()
    {
        return await this.context.Series.ToListAsync();
    }

    public async Task<Serie> FindSerieAsync(int idSerie)
    {
        return await this.context.Series
            .FirstOrDefaultAsync(z => z.IdSerie == idSerie);
    }

    private async Task<int> GetMaxIdSerieAsync()
    {
        return await this.context.Series
            .MaxAsync(x => x.IdSerie) + 1;
    }

    public async Task CreateSerieAsync
        (string nombre, string imagen, int anyo)
    {
        Serie serie = new Serie();
        serie.IdSerie = await this.GetMaxIdSerieAsync();
        serie.Nombre = nombre;
        serie.Imagen = imagen;
        serie.Anjo = anyo;
        await this.context.Series.AddAsync(serie);
        await this.context.SaveChangesAsync();
    }

    public async Task UpdateSerieAsync
        (int idSerie, string nombre, string imagen, int anyo)
    {
        Serie serie = await this.FindSerieAsync(idSerie);
        serie.Nombre = nombre;
        serie.Imagen = imagen;
    }
}

```

```

        serie.Año = año;
        await this.context.SaveChangesAsync();
    }

    public async Task DeleteSerieAsync(int idSerie)
    {
        Serie serie = await this.FindSerieAsync(idSerie);
        this.context.Series.Remove(serie);
        await this.context.SaveChangesAsync();
    }
}

```

Sobre **appsettings.json** incluimos nuestra cadena de conexión

APPSETTINGS.JSON



```

{
  "Logging": {
    "LogLevel": {
      "Default": "Information"
    }
  },
  "ConnectionStrings": {
    "MySql": "server=awsmysqlpaco.c23akq6oglxj.us-east-1.r"
  }
}

```

Lo bueno es que trabajamos de forma "tradicional", mediante **Controllers**

Sobre **Controllers**, agregamos una nueva clase llamada **SeriesController**

SERIESCONTROLLER

```

[Route("api/[controller]")]
[ApiController]
public class SeriesController : ControllerBase
{
    private RepositorySeries repo;

    public SeriesController(RepositorySeries repo)
    {
        this.repo = repo;
    }

    [HttpGet]
    public async Task<ActionResult<List<Serie>>> GetSeries()
    {
        return await this.repo.GetSeriesAsync();
    }

    [HttpGet("{id}")]
    public async Task<ActionResult<Serie>> FindSerie(int id)
    {
        return await this.repo.FindSerieAsync(id);
    }

    [HttpPost]
    public async Task<ActionResult> Create(Serie serie)
    {
        await this.repo.CreateSerieAsync(serie.Nombre,
            serie.Imagen, serie.Año);
        return Ok();
    }

    [HttpPut]
    public async Task<ActionResult> Edit(Serie serie)
    {
        await this.repo.UpdateSerieAsync(serie.IdSerie
            , serie.Nombre, serie.Imagen, serie.Año);
        return Ok();
    }

    [HttpDelete("{id}")]
    public async Task<ActionResult> Delete(int id)
    {
        await this.repo.DeleteSerieAsync(id);
        return Ok();
    }
}

```

Realizamos la inyección dentro de **Startup.cs**

STARTUP

```

public class Startup
{
    public Startup(IConfiguration configuration)
    {

```

```

        Configuration = configuration;
    }

    public IConfiguration Configuration { get; }

    public void ConfigureServices(IServiceCollection services)
    {
        string connectionString =
            Configuration.GetConnectionString(" MySql ");
        services.AddTransient<RepositorySeries>();
        services.AddDbContext<SeriesContext>
            (options => options.UseMySql(connectionString
                , ServerVersion.AutoDetect(connectionString)));
        //ESTE TIPO DE APIS UTILIZAN UN PROXY PARA LAS LLAMADAS
        //ES IMPRESCINDIBLE HABILITAR CORS
        services.AddCors(options =>
        {
            options.AddPolicy("AllowOrigin", x => x.AllowAnyOrigin());
        });
        services.AddControllers();
    }

    public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
    {
        if (env.IsDevelopment())
        {
            app.UseDeveloperExceptionPage();
        }
        app.UseCors(options => options.AllowAnyOrigin());
        app.UseHttpsRedirection();
        app.UseRouting();
        app.UseAuthorization();

        app.UseEndpoints(endpoints =>
        {
            endpoints.MapControllers();
            endpoints.MapGet("/", async context =>
            {
                await context.Response.WriteAsync("Welcome to running ASP.NET
Core on AWS Lambda");
            });
        });
    }
}

```

El siguiente paso es ponerlo "bonito" y agregar **Swagger**



Modificamos el código de **Startup**

STARTUP

```

public class Startup
{
    public Startup(IConfiguration configuration)
    {
        Configuration = configuration;
    }

    public IConfiguration Configuration { get; }

    public void ConfigureServices(IServiceCollection services)
    {
        string connectionString =
            Configuration.GetConnectionString(" MySql ");
        services.AddTransient<RepositorySeries>();
        services.AddDbContext<SeriesContext>
            (options => options.UseMySql(connectionString
                , ServerVersion.AutoDetect(connectionString)));
        //ESTE TIPO DE APIS UTILIZAN UN PROXY PARA LAS LLAMADAS
        //ES IMPRESCINDIBLE HABILITAR CORS
        services.AddCors(options =>
        {
            options.AddPolicy("AllowOrigin", x => x.AllowAnyOrigin());
        });

        services.AddSwaggerGen(options =>
        {
            options.SwaggerDoc("v1", new OpenApiInfo
            {
                Title = "Api AWS Viernes!!!",
                Version = "v1"
            });
        });
        services.AddControllers();
    }

    public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
    {
        if (env.IsDevelopment())
        {
            app.UseDeveloperExceptionPage();
        }
        app.UseCors(options => options.AllowAnyOrigin());
    }
}

```

```

app.UseSwagger();
app.UseSwaggerUI(options =>
{
    options.SwaggerEndpoint(url: "swagger/v1/swagger.json"
        , "ApiSeriesAWS");
    options.RoutePrefix = "";
});

app.UseHttpsRedirection();
app.UseRouting();
app.UseAuthorization();

app.UseEndpoints(endpoints =>
{
    endpoints.MapControllers();
    endpoints.MapGet("/", async context =>
    {
        await context.Response.WriteAsync("Welcome to running ASP.NET
Core on AWS Lambda");
    });
});
}
}

```

Y funciona y todo!!!!

Swagger
Supported by SMARTBEAR

Select a definition | ApiSeriesAWS

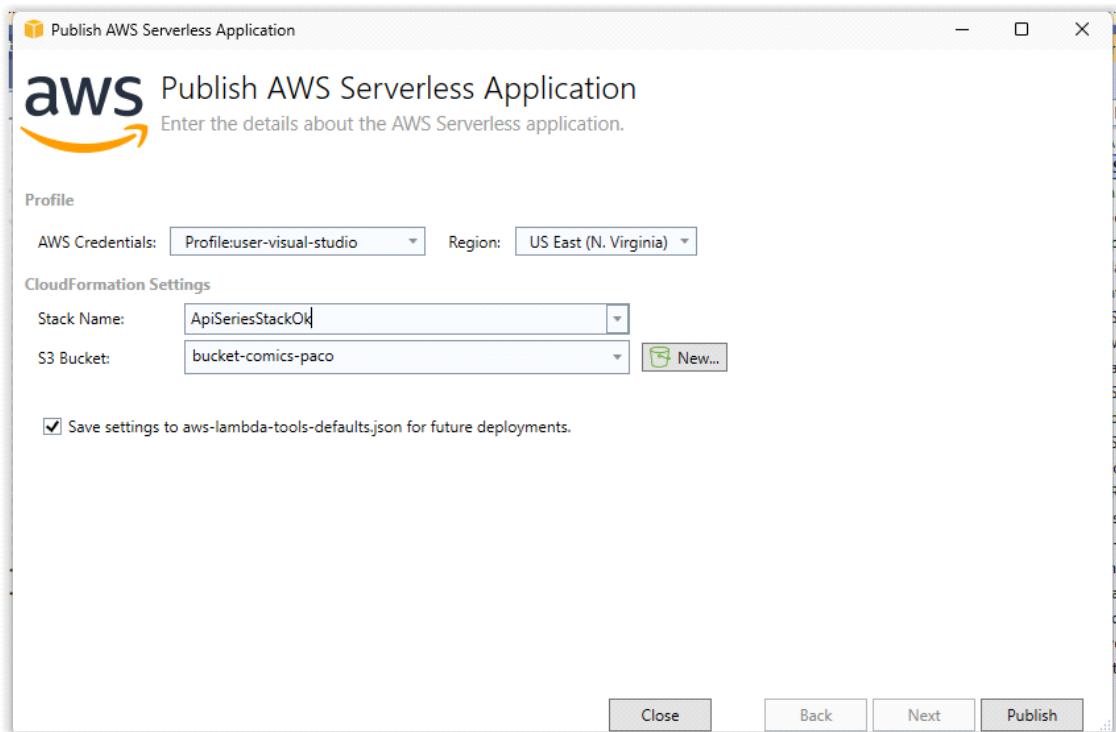
Api AWS Viernes!!! v1 OAS 3.0

<https://localhost:57565/swagger/v1/swagger.json>

Series

- GET /api/Series
- POST /api/Series
- PUT /api/Series
- GET /api/Series/{id}
- DELETE /api/Series/{id}

El siguiente paso es publicar nuestro servicio mediante Api Gateway



Y ya tendremos la aplicación publicada

Swagger. Supported by SMARTBEAR

Select a definition ApiSeriesAWS

Api AWS Viernes!!! v1 OAS 3.0

<https://jpcwzzvg9f.execute-api.us-east-1.amazonaws.com/Prod/swagger/v1/swagger.json>

Servers

/Prod

Series

GET /api/Series

POST /api/Series

PUT /api/Series

Vamos a visualizar los recursos que nos ha generado a diferencia del proyecto anterior.

Etapas

[Acciones de etapa ▾](#)[Create stage](#)

- Prod
- /
 - DELETE
 - GET
 - HEAD
 - OPTIONS
 - PATCH
 - POST
 - PUT
 - /{proxy+}
 - DELETE
 - GET
 - HEAD
 - OPTIONS

Anulaciones de métodos

[Crear anulación](#)

De forma predeterminada, los métodos heredan la configuración de nivel de etapa. Para personalizar la configuración de un método, configure las anulaciones de métodos.

ⓘ Este método hereda su configuración de la etapa "Prod".

URL de invocación

□ <https://jpcwzzvg9f.execute-api.us-east-1.amazonaws.com/Prod/{proxy+}>



AWS LAMBDA S3 CAPAS

viernes, 23 de mayo de 2025 12:52

Para la siguiente práctica vamos a visualizar que son las Capas/Layers de funciones Lambda

Vamos a realizar una función en código Python.

Dicha función, lo que hará será leer cuando subamos un fichero a cualquier Bucket de un S3.

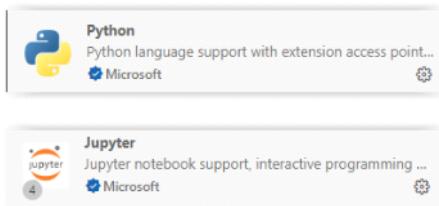
Leeremos dicho fichero y, mediante Python vamos a generar un gráfico con los datos.

Instalamos **Python 3.13**

<https://www.python.org/>

Lo único que tenemos que hacer es marcar Add PATH

Abrimos VS Code e instalamos las siguientes extensiones:

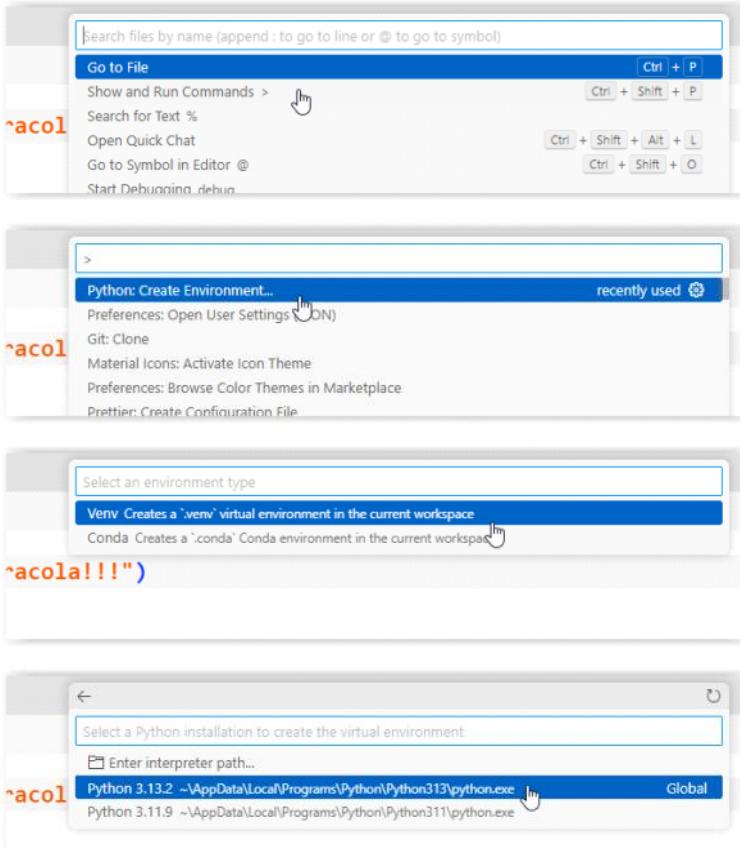


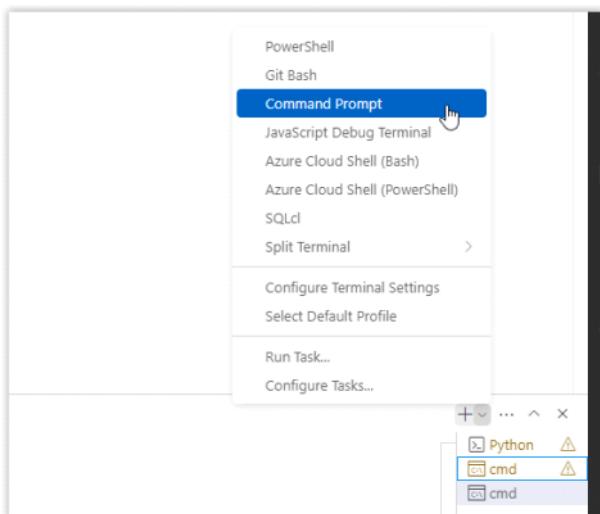
Creamos una carpeta de pruebas llamada **PythonAWS** y la abrimos con VS Code

Creamos una carpeta llamada **data** y dentro nuestro fichero **got.csv**



Para probar si funciona bien esto, creamos un nuevo fichero llamado **test.py**





PROBLEMS OUTPUT PORTS SQL HISTORY TASK MONITOR TERMINAL AZURE

```
Microsoft Windows [Versión 10.0.26100.4061]
(c) Microsoft Corporation. Todos los derechos reservados.

(.venv) C:\PythonAWS>
```

Acabamos de crear un entorno aislado para nuestras librerías de Python

Lo que vamos a realizar es mostrar un gráfico con datos de Ciencia de Game Of Thrones.

Necesitamos primero algo que se llama análisis de datos que utiliza una librería llamada **pandas**

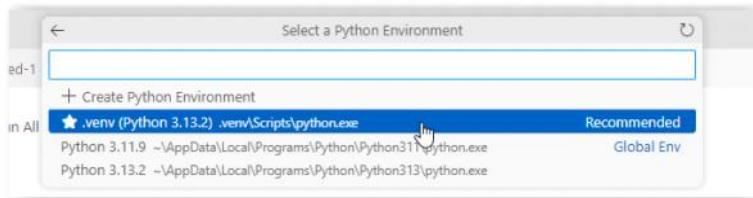
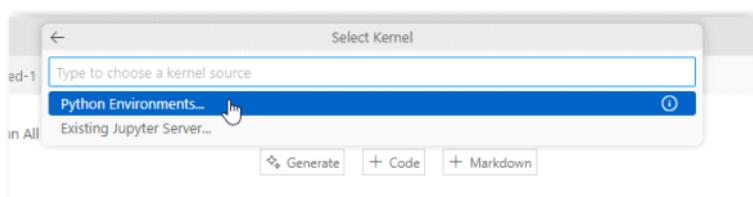
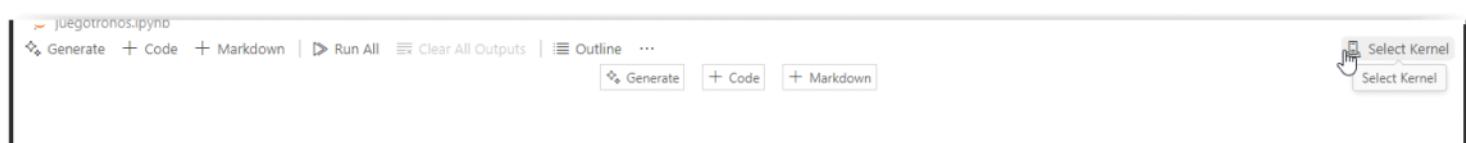
Para instalar: **pip install pandas**

```
import pandas as pd

# creamos un dataframe con el fichero CSV
df = pd.read_csv("data/got.csv")
#print(df.head(10))
#imprimimos el dataframe
```

La siguiente prueba para ver esto mejor es utilizar un cuaderno de **Jupyter** que es lo que se utilizará para analizar datos.

Creamos un nuevo fichero **juegotrones.ipynb**



Agrupamos el Dataframe por asesino

```
df_killer = df.groupby('killer')['death_no'].agg(['count'])
df_killer
```

count

Necesitamos mostrar gráficos

```
(.venv) C:\PythonAWS>pip install matplotlib
```

Vamos a separar estos datos por temporada
Para generar gráficos con los primeros asesinos por temporada

```
df_season = df[ df['season'] == 4]
df_season.to_csv('data/got4.csv')
df_season
```

	name	allegiance	season	episode	location	killer	killers_house	me
Baratheon of King's	House Baratheon of					Sandor "the Hound"	House	
274	.	.	.	4	1	Riverlands	.	.

Una vez que tenemos los ficheros por temporada, vamos a utilizar un Lambda con S3

Necesitamos un bucket público llamado **bucket-got-paco**

Step 1: Select Policy Type

A Policy is a container for permissions. The different types of policies you can create are an [IAM Policy](#), an [S3 Bucket Policy](#), an [SNS Topic Policy](#), a [VPC Endpoint Policy](#), and an [SQS Queue Policy](#).

Select Type of Policy

Step 2: Add Statement(s)

A statement is the formal description of a single permission. See [a description of elements](#) that you can use in statements.

Effect Allow Deny

Principal

Use a comma to separate multiple values.

AWS Service

All Services (*)

Use multiple statements to add permissions for more than one service.

Actions All Actions (*)

Amazon Resource Name (ARN)

ARN should follow the following format: arn:aws:s3:::\${BucketName}/\${KeyName}.

Use a comma to separate multiple values.

Add Conditions (Optional)

Necesitamos un Role con permisos sobre **Lambda, S3 y CloudWatch**

Abrimos IAM

Seleccionar entidad de confianza Información

Tipo de entidad de confianza

Servicio de AWS

Permita que servicios de AWS como EC2, Lambda u otros realicen acciones en esta cuenta.

Cuenta de AWS

Permitir a las entidades de otras cuentas de AWS que le pertenezcan a usted o a un tercero realizar acciones en esta cuenta.

Caso de uso

Permita que un servicio de AWS, como EC2, Lambda u otros, realicen acciones en esta cuenta.

Servicio o caso de uso

Lambda

Elija un caso de uso para el servicio especificado.

Caso de uso

Lambda

Allows Lambda functions to call AWS services on your behalf.



Llamamos al Role lambda-role-got

Detalles del rol

Nombre del rol

Ingrese un nombre significativo para identificar a este rol.

lambda-role-got

64 Caracteres máximos. Utilice caracteres alfanuméricos y '+ =_.@-'.

Descripción

Agregue una breve explicación para este rol.

Allows Lambda functions to call AWS services on your behalf.

Máximo de 1000 caracteres. Utilice letras (A-Z y a-z), números (0-9), tabulaciones, nuevas líneas o cualquiera de los siguientes caracte

Una vez que tenemos el bucket, vamos a Lambda y creamos una nueva función llamada lambda-got

Información básica

Nombre de la función

Escriba un nombre para describir el propósito de la función.

lambda-got

El nombre de la función debe tener entre 1 y 64 caracteres, debe ser exclusivo de la región y no puede incluir espacios. Los caracteres válidos son a-z, A-Z, 0-9, guiones (-) y guiones bajos (_).

Tiempo de ejecución | Información

Elija el idioma para usar para escribir su función. Tenga en cuenta que el editor de código de la consola es compatible con solo Node.js, Python y Ruby.

Python 3.13



Arquitectura | Información

Elija la arquitectura del conjunto de instrucciones que desea para el código de la función.

arm64

x86_64

Permisos | Información

De forma predeterminada, Lambda creará un rol de ejecución con permisos para cargar registros en Amazon CloudWatch Logs. Puede personalizar este rol predeterminado más adelante al agregar los disparadores.

▼ Cambiar el rol de ejecución predeterminado

Rol de ejecución

Seleccione un rol que defina los permisos de la función. Para crear un rol personalizado, vaya a la consola de IAM [\[\]](#).

- Creación de un nuevo rol con permisos básicos de Lambda
- Uso de un rol existente
- Creación de un nuevo rol desde la política de AWS templates

Rol existente

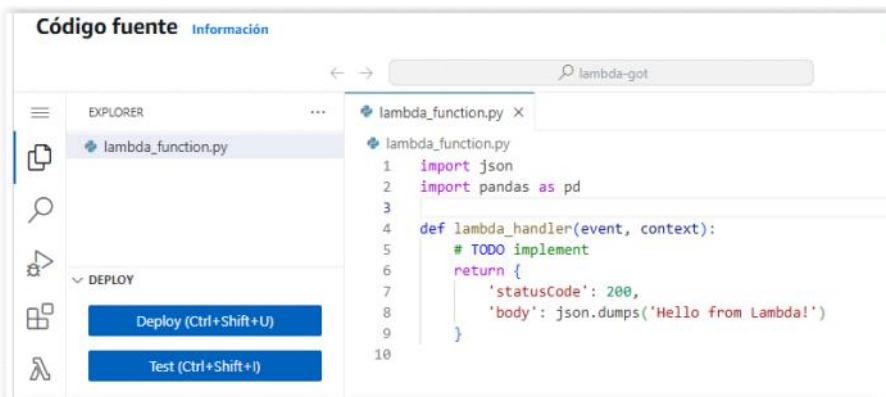
Seleccione un rol existente que haya creado para usarlo con esta función de Lambda. El rol debe tener permiso para cargar registros en Amazon

lambda-role-got



Consulte el rol lambda-role-got [\[\]](#) en la consola de IAM.

Incluimos el siguiente código en la función



```
lambda_function.py
1 import json
2 import pandas as pd
3
4 def lambda_handler(event, context):
5     # TODO implement
6     return {
7         'statusCode': 200,
8         'body': json.dumps('Hello from Lambda!')
9     }
10
```

Y veremos que nos estará dando un error



Ejecutando la función: error ([registros](#))

Detalles

```
{ "errorMessage": "Unable to import module 'lambda_function': No module named 'pandas'", "errorType": "Runtime.ImportModuleError", "requestId": "", "stackTrace": [] }
```

Resumen

Código SHA-256
VYM8TVI7qh7LVJVpUBXUIGuoUJxOWZ5TSGMr7A
01CEUU-

Tiempo de ejecución
hace 1 minuto

Los códigos que tenemos son aislados, necesitamos las librerías para poder ejecutar los códigos en AWS

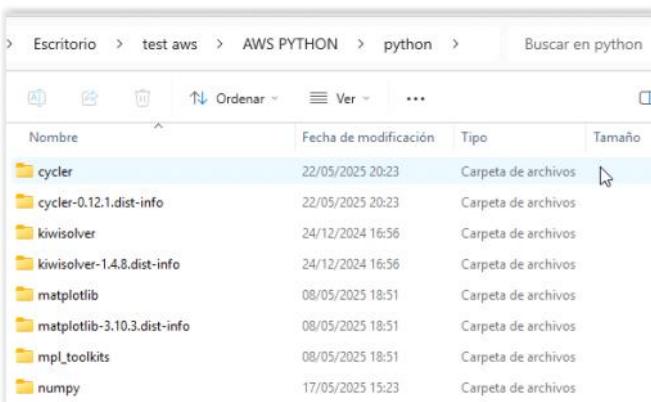
Dichos códigos con librerías se llaman **Layers**

Dependiendo del lenguaje, tenemos que descargar la librería adecuada, por ejemplo, Python necesita

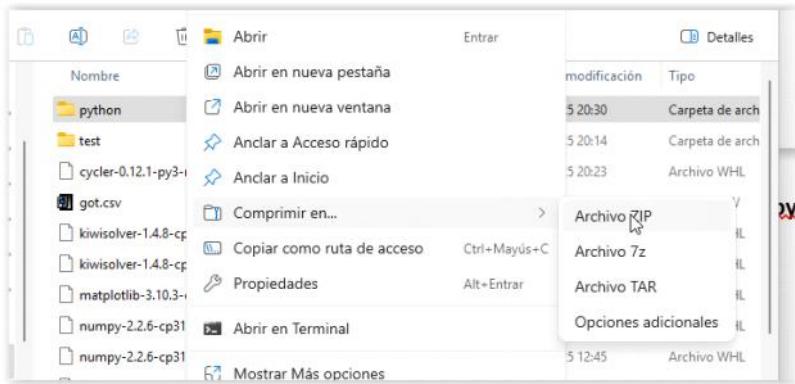
Una librería con la versión (3.13), el SO (Linux) y el motor (x86)



Con todas las librerías, tendríamos que extraerlas y crear una carpeta llamada **python** y poner todos los elementos en su interior



Una vez que tenemos todo, simplemente realizar un ZIP con todo el contenido de la carpeta **python**



Dicho zip se utiliza para las capas de AWS.

Abrimos Lambda y vamos a crear una nueva capa con **python.zip**

Crear capa

Configuración de capa

Nombre
pandas

Descripción - Opcional
Pandas y lo que toque...

Cargar un archivo .zip
 Cargar un archivo de Amazon S3

[Elegir archivo](#)

python.zip
44.22 MB

Para los archivos mayores de 10 MB, considere la posibilidad de cargarlos usando Amazon S3.

Arquitecturas compatibles - Opcional | [Información](#)
Elija las arquitecturas de conjunto de instrucciones compatibles para la capa.

x86_64 [X](#)

Versiones ejecutables compatibles - Opcional | [Información](#)
Elija hasta 15 versiones ejecutables.

Python 3.13 [X](#)

Asociamos nuestra nueva Layer a la función Lambda

lambda-got

▼ Información general de la función [Información](#) [Exportar a Infras](#)

[Diagrama](#) [Plantilla](#)

 **lambda-got**
 Layers (0)

[+ Agregar desencadenador](#) [+ Agregar destino](#)

Capas [Información](#)

Orden de combinación | Nombre | Versión de la capa | Tiempos de ejecución compatibles | Arquitecturas compatibles | ARN de la capa

No hay datos que mostrar.

Elija una capa

Fuente de capa [Información](#)
 Elija entre capas con una versión ejecutable compatible y una arquitectura de conjunto de instrucciones o especifique el nombre de recurso de Amazon (ARN) de una versión de capa. También puede [crear una nueva capa](#).

Capas de AWS
 Elija una capa de una lista de capas proporcionadas por AWS.

Capas personalizadas
 Choose a layer from a list of layers created by your AWS account.

Especificar un ARN
 Proporcione el ARN y especifique una capa.

Capas personalizadas
 Layers created by your AWS account that are compatible with your function's runtime.

pandas

Versión

15

Y ya tenemos PANDAS!!!

Código [Probar](#) Monitorear Configuración Alias Versiones

Ejecutando la función: sin errores ([registros](#))

▼ Detalles

```
{
  "statusCode": 200,
  "body": "\"Hello from Lambda!\""
}
```

El siguiente paso que vamos a realizar es leer un fichero CSV cuando lo suban a un S3

Escribimos el siguiente código dentro de nuestra función lambda

```
import json
import pandas as pd
#Librería para el S3
import boto3
import urllib.parse
#Creamos el cliente S3
s3 = boto3.client('s3')
#nuestro nombre de bucker
bucket_name = 'bucket-got-paco'
def lambda_handler(event, context):
    #Un bucket está compuesto por el nombre del
    propio bucket y la key que es
    #el fichero que estamos subiendo
    #Tenemos la opción de leer el propio bucket
    s3 o podemos leer la URL
    bucket = event['Records'][0]['s3']['bucket']
    ['name']
```

```

key =
urllib.parse.unquote_plus(event['Records'][0]
['s3']['object']['key'], encoding='utf-8')
response = s3.get_object(Bucket=bucket,
Key=key)
print('Tenemos fichero!!!!')
print(response)
return {
'statusCode': 200,
'body': json.dumps('Hello from Lambda!')}
}

```

Este código ya NO podemos probarlo directamente con un evento, debe ser probado mediante la subida de algún Fichero a nuestro Bucket.

Agregar desencadenador

Configuración del desencadenador

S3 aws asynchronous storage

Bucket
Seleccione o escriba el ARN de un bucket de S3 que actúa como origen de eventos. El bucket debe estar en la misma región que la función.
s3/bucket-got-paco

Región del bucket: us-east-1

Tipos de eventos
Seleccione los eventos que desea que activen la función de Lambda. Si lo desea, también puede configurar un prefijo o un sufijo para un evento. Sin embargo, en cada evento no puede haber eventos individuales con configuraciones múltiples que tengan prefijos o sufijos superpuestos que puedan coincidir con la misma clave de objeto.

Todos los eventos de creación de objetos

Prefijo - Opcional
Introduzca un único prefijo opcional para limitar las notificaciones a los objetos cuyas claves comienzan por los caracteres coincidentes. Todos los [caracteres especiales](#) deben estar codificados en URL.
p. ej., imágenes/

Sufijo - Opcional
Introduzca un único sufijo opcional para limitar las notificaciones a los objetos cuyas claves terminen por los caracteres coincidentes. Todos los [caracteres especiales](#) deben estar codificados en URL.
p. ej., .jpg

Invocación recurrente
Si la función escribe objetos en un bucket de S3, asegúrese de utilizar diferentes buckets de S3 para las entradas y las salidas. Escribir en el mismo bucket aumenta el riesgo de crear una invocación recurrente, lo que puede resultar en un aumento del uso de Lambda y de los costos. [Más información](#)

Acepto que no se recomienda utilizar el mismo bucket de S3 para las entradas y las salidas, y que esta configuración puede provocar invocaciones recurrentes, así como también un aumento del uso de Lambda y de los costos.

Lambda añadirá los permisos necesarios para AWS S3 para invocar la función Lambda desde este desencadenador. [Obtenga más información](#) sobre el modelo de permisos de Lambda.

[Cancelar](#) [Agregar](#)

El siguiente paso será visualizar que ha pasado mediante **CloudWatch**

Código fuente Información

Monitor Información

Ver registros de CloudWatch [\[\]](#) Ver Application Signals [\[\]](#) Ver rastros de X-Ray [\[\]](#) Ver Lambda Insights [\[\]](#)

Ver perfiles de CodeGuru [\[\]](#)

Filter metrics by [Función](#)

3h 1d 1sem. [\[\]](#) Zona horaria UTC [\[\]](#) Explorar contenido

Recomendaciones de alarma [\[\]](#)

CloudWatch metrics

Crear una aplicación web sencilla

En este tutorial aprenderá a hacer lo siguiente:

- Crear una aplicación sencilla, que consista una función de Lambda con una URL de función que genere una página web.

Monitor Información

Ver registros de CloudWatch [\[\]](#) Ver Application Signals [\[\]](#)

Ver perfiles de CodeGuru [\[\]](#)

CloudWatch < Favoritos y recientes ▶ Paneles ▶ Operaciones de inteligencia artificial ▶ Alarms ▶ ▶ Registros Grupos de registros Anomalías de registros Live Tail Logs Insights Contributor Insights

Bytes almacenados 1.97 KB Detección de anomalías Configurar

Flujos de registros Etiquetas Detección de anomalías Filtros de métricas Filtros de suscripción

Flujos de registros (5) Eliminar Crear flujo de registros Buscar en todas las regiones

Filtrar flujos de registro o probar la búsqueda de prefijos

Secuencia de registro Hora del último evento

2025/05/26/[\$LATEST]3bdd7063055477c8fa1475d5eb1e27	2025-05-26 08:16:14 (UTC)	
2025/05/26/[\$LATEST]872de0819d954dd8b1799e13c64324f8	2025-05-26 07:52:57 (UTC)	
2025/05/23/[\$LATEST]715635b2b01b4a4f9c80d3184b628eb1	2025-05-23 12:20:23 (UTC)	
2025/05/23/[\$LATEST]29ef4d32e9e24499a2bf4e6f5e83eefc	2025-05-23 12:04:13 (UTC)	

Una vez que hemos visto cómo controlar lo que subimos a un Bucket, nos falta un último paso de configuración.

Por defecto, la ejecución de una función está delimitada a 3 segundos y solamente admite unos pocos bytes

Si queremos subir ficheros más grandes (Análisis de datos) debemos configurar el tiempo de respuesta de nuestra función y la memoria que utilizará para leer su código.

Entramos en la función y en **Configuración**

Editar configuración básica

Configuración básica [Información](#)

Descripción - Opcional

Memoria [Información](#)
La CPU asignada a la función es proporcional a la memoria configurada.

MB
Establezca la memoria en un valor entre 128 MB y 10240 MB

Almacenamiento efímero [Información](#)
Puede configurar hasta 10 GB de almacenamiento efímero (/tmp) para la función. [Ver precios](#)

MB
Establezca el almacenamiento efímero (/tmp) entre 512 MB y 10240 MB

Una vez que está todo configurado y hemos visto cómo leer la ejecución de la función, vamos a cargar un nuevo Código leyendo un CSV y devolviendo los 5 primeros asesinos.

Tenemos dos formas de realizar esta acción:

- 1) Leer el contenido del fichero (privado)
- 2) Leer directamente la URL de nuestro fichero. (público)

URI DE S3
 s3://buckets-artifact-paco/1.png

Nombre de recurso de Amazon (ARN)
 arn:aws:s3:::buckets-artifact-paco/1.png

Etiqueta de entidad (Etag)
 27ec8d56f148ba7b6d06bf42e3f9518a

URL del objeto
 <https://buckets-artifact-paco.s3.us-east-1.amazonaws.com/1.png>

Primero probamos que pueda leer los primeros 5 asesinos

CODIGO LAMBDA

```
import json
import pandas as pd
#Librería para el S3
import boto3
import urllib.parse
#Creamos el cliente S3
s3 = boto3.client('s3')
#nuestro nombre de bucker
bucket_name = 'buckets-artifact-paco'
def lambda_handler(event, context):
    #Un bucket está compuesto por el nombre del
    propio bucket y la key que es
    #el fichero que estamos subiendo
    #Tenemos la opción de leer el propio bucket
    s3 o podemos leer la URL
    bucket = event['Records'][0]['s3']['bucket']
    ['name']
    key =
    urllib.parse.unquote_plus(event['Records'][0]
    ['s3']['object']['key'], encoding='utf-8')
    response = s3.get_object(Bucket=bucket,
    Key=key)
    print('Tenemos fichero!!!!')
    url = f"https://{bucket_name}.s3.us-
east-1.amazonaws.com/{key}";
    df = pd.read_csv(url)
    print("leyendo file " + key)
    df_killer = df.groupby('killer')
    ['death_no'].agg(['count'])
    df_primeros = df_killer.head(5)
    print(df_primeros)
    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }
```

Y veremos la información

► Marca temporal	Mensaje
	No hay eventos antiguos en este momento. Volver a intentar
► 2025-05-26T08:46:45.264Z	INIT_START Runtime Version: python:3.13.v40 Runtime Version ARN: arn:aws:lambda:us-east-1:runtim
► 2025-05-26T08:46:47.779Z	START RequestId: 1f471200-dcc8-43e0-8487-708e0c40e999 Version: \$LATEST
► 2025-05-26T08:46:47.830Z	Tenemos fichero!!!!
► 2025-05-26T08:46:47.893Z	leyendo file got1.csv
► 2025-05-26T08:46:47.929Z	count
► 2025-05-26T08:46:47.929Z	Killer
► 2025-05-26T08:46:47.929Z	Arya Stark 2
► 2025-05-26T08:46:47.929Z	Boar 1

Ahora que por fin podemos leer los ficheros CSV con nuestro bucket, vamos a generar un gráfico y almacenarlo

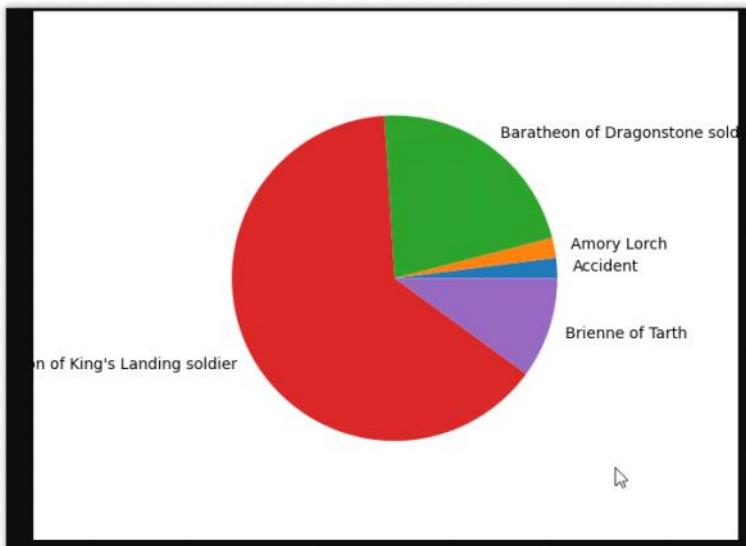
En el mismo bucket con el nombre del fichero subido.

CODIGO LAMBDA

```
import json
import pandas as pd
#Librería para el S3
import boto3
import urllib.parse
import matplotlib.pyplot as plt
from io import StringIO, BytesIO
#Creamos el cliente S3
s3 = boto3.client('s3')
#nuestro nombre de bucker
bucket_name = 'buckets-artifact-paco'
def lambda_handler(event, context):
    #Un bucket está compuesto por el nombre del propio bucket y la key que es
    #el fichero que estamos subiendo
    #Tenemos la opción de leer el propio bucket s3 o podemos leer la URL
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']
    ['key']), encoding='utf-8')
    response = s3.get_object(Bucket=bucket, Key=key)
    print('Tenemos fichero!!!!')
    url = f"https://[{bucket_name}].s3.us-east-1.amazonaws.com/{key}"
    df = pd.read_csv(url)
    print("leyendo file " + key)
    df_killer = df.groupby('killer')['death_no'].agg(['count'])
    df_primeros = df_killer.head(5)
    print(df_primeros)
    #capturamos los datos para el gráfico
    data = df_primeros.groupby('killer').sum()
    #generamos un nuevo gráfico limpio
    plt.cla()
    plt.clf()
    #creamos una imagen en memoria para el fichero
    img_data = BytesIO()
    #generamos el gráfico con los datos
    plt.pie(x=data["count"], labels=data.index)
    print("generando gráficos")
    #guardamos la imagen
    plt.savefig(img_data, format='png')
    img_data.seek(0)

    #el último paso es subir nuestra imagen a nuestro bucket
    #nombre de imagen
    image_name = key + ".png"
    bucket = boto3.resource('s3').Bucket(bucket_name)
    bucket.put_object(Body=img_data, ContentType='image/png', Key=image_name)
    print("Tenemos una imagen subida!!!!")
    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }
```

Y ya tenemos gráficos con Lambda!!!



ALEXA AWS LAMBDA

Lunes, 26 de mayo de 2025 11:26

El cacharro de Amazon que habla.

Podemos crear **Skills** para personalizar la experiencia de dichos dispositivos, incluso que lean nuestros Recursos de AWS si lo necesitamos.

Tenemos dos formas de crear esta funcionalidad:

- 1) **Lambda:** Tenemos códigos de ejemplos en los que, mediante un Trigger podemos invocar a Un Skill de Alexa y que nos conteste
- 2) **Net Core:** Con una función Lambda podemos generar un código nuestro

Elementos para este laboratorio:

- **LAMBDA**
- **Amazon Developers:** Es una página que no tiene que ver con AWS aunque lo podríamos enlazar.
Podemos implementar características en esta página para Alexa

<https://developer.amazon.com/es/>

Entramos en Lambda y vamos a crear una función de prueba con código ya listo para visualizar cómo Podemos interactuar con Alexa.

Y entramos en **Aplicaciones**. Una aplicación es una función que ya viene con un desencadenador.

Aplicaciones sin servidor [Información](#)

Public applications (8) Private applications

[X](#) Coincidencias: 8 Sort by Best Match ▾

Show apps that create custom IAM roles or resource policies < 1 >

[alexa-skills-kit-python36-factskill](#)
This Alexa sample skill is a template for a basic fact skill. When the skill is invoked, Alexa will select a random fact from a list of interesting facts about a topic, Alexa will select a fact at random and tell it to the user when the skill is invoked.

[alexa-skills-kit-nodejs-factskill](#)
This Alexa sample skill is a template for a basic fact skill. Provided a list of interesting facts about a topic, Alexa will select a fact at random and tell it to the user when the skill is invoked.

skills fact python39 alexa skills fact alexa

Nos habrá creado una función con un Trigger de Alexa

▼ Información general de la función [Información](#) Exportar a Infrastructure Composer Descarga

Diagrama | Plantilla

serverlessrepo-alexa-skil-alexaskillskitnodejsfact-Sjv9UPAEYtYF

Layers (0)

Alexa + Agregar destino

+ Agregar desencadenador

Descripción
Demonstrate a basic fact skill built with the ASK NodeJS SDK

Última modificación
hace 51 segundos

ARN de la función
arn:aws:lambda:us-east-1:772057005:function:serverlessrepo-alexa-skil-alexaskillskitnodejsfact-Sjv9UPAEYtYF

Aplicación
[serverlessrepo-alexa-skills-kit-nodejs-factskill](#)

URL de la función [Información](#)

También tenemos un código que nos devuelve mensajes random al ser invocado

Lambda > Funciones > serverlessrepo-alexa-skil-alexaskillskitnodejsfact-Sjv9UPAEYtYF

index.js

```
247 },
248 };
249
250 const esData = {
251   translation: {
252     SKILL_NAME: 'Curiosidades del Espacio',
253     GET_FACT_MESSAGE: 'Aquí está tu curiosidad: ',
254     HELP_MESSAGE: 'Puedes decir dime una curiosidad del espacio o puedes ',
255     HELP_REPROMPT: 'Como te puedo ayudar?',
256     FALLBACK_MESSAGE: 'La skill Curiosidades del Espacio no te puede ayu',
257     FALLBACK_REPROMPT: 'Como te puedo ayudar?',
258     ERROR_MESSAGE: 'Lo sentimos, se ha producido un error.',
259     STOP_MESSAGE: 'Adiós!',
260   }
261 }
```

package.json

DEPLOY Deploy (Ctrl+Shift+U) Test (Ctrl+Shift+I)

TEST EVENTS [NONE SELECTED]

```

    JS index.js
    package.json

    ✓ DEPLOY
        Deploy (Ctrl+Shift+U)
        Test (Ctrl+Shift+T)

    ✓ TEST EVENTS [NONE SELECTED]
        + Create new test event

    0 ▲ 0 ▷ Amazon Q
    Ln 20, Col 1 (32 selected) Spaces: 2 UTF-8 LF JavaScript Lambda Layout US

```

```

    247 },
    248 );
    249
    250 const esData = {
    251     translation: [
    252         SKILL_NAME: 'Curiosidades del Espacio',
    253         GET_FACT_MESSAGE: 'Aquí está tu curiosidad: ',
    254         HELP_MESSAGE: 'Puedes decir dime una curiosidad del espacio o puede',
    255         HELP_REPROMPT: 'Como te puedo ayudar?',
    256         FALLBACK_MESSAGE: 'La skill Curiosidades del Espacio no te puede ayu',
    257         FALLBACK_REPROMPT: 'Como te puedo ayudar?',
    258         ERROR_MESSAGE: 'Lo sentimos, se ha producido un error.',
    259         STOP_MESSAGE: 'Adiós!',
    260         FACTS: [
    261             [
    262                 'Un año en Mercurio es de solo 88 días',
    263                 'A pesar de estar más lejos del Sol, Venus tiene temperaturas',
    264                 'En Marte el sol se ve la mitad de grande que en la Tierra',
    265                 'Júpiter tiene el día más corto de todos los planetas',
    266                 'El sol es una esfera casi perfecta'
    267             ]
    268         ]
    269     ]
    270 }
    271
    272 module.exports.handler = (event, context, callback) => {
    273     const response = {
    274         version: '1.0',
    275         sessionAttributes: {},
    276         responseType: 'SimpleCard',
    277         cardTitle: 'Curiosidades del Espacio',
    278         message: 'Aquí está tu curiosidad: ' + esData.translation.GET_FACT_MESSAGE,
    279         reprompt: 'Puedes decir dime una curiosidad del espacio o puedes preguntar "¿Qué es la curiosidad del espacio?"',
    280         endSession: true
    281     }
    282
    283     callback(null, response)
    284 }

```

Una vez que tenemos esto montado lo que necesitamos hacer es enlazar nuestro Alexa con esta función

amazon developer

All Search

Dashboard Apps & Services Alexa Login with Amazon Amazon Dash Replenishment Reporting

Alexa Skills Kit

Introducing the Small Business Program

Developers earning less than \$100K annually receive benefits equivalent to 10% of App Store revenue.

Alexa Voice Service

Alexa Connect Kit

Amazon appstore

Build for voice with Alexa, Amazon's voice service and the brain behind the Amazon Echo

Build Android apps and games for Amazon Fire tablet, and Amazon's mobile app store.

Creamos un nuevo Skill

Skills Earnings Payments Hosting Settings

Alexa Skills Skill examples Learn more

Search by skill name or skill ID Hide hidden & removed skills Create Skill

SKILL NAME	LANGUAGE	MODIFIED	STATUS	ACTIONS
verdades pacd				

1. Name your Skill

Enter a name for your skill. This is not the same as the skill invocation name, which you'll set up after you complete the skill setup.

Skill name

verdades pacd

13/50 characters

Brand names are only allowed if you provide proof of rights in the testing instructions or if you use the brand name in a referential manner (that is, to a brand name for referential usage: unofficial, unauthorized, fan, fandom, for, about).

2. Choose a primary locale

A locale refers to a language and the country in which it's spoken. Build your skill in your primary locale. You can add more locales later.

Spanish (Spain)

1. Choose a type of experience

Tell us what kind of experience you want to build and we'll recommend a voice interaction model (also known as a skill model) to get you started. A skill model has pre-defined words and phrases that users can say when interacting with your skill. You'll be able to customize what Alexa responds with that is unique to your skill's use-case.

- Food & Drink ⓘ Games & trivia ⓘ Movies & TV ⓘ Music & Audio ⓘ News ⓘ Smart home ⓘ
- Other ⓘ

Name, Locale
verdades paco, Spanish (Spain)

2 Experience, Model, Hosting service

3 Templates

4

Based on what you've told us, we think this model would work best for your skill.

Note: Different models allow for different degrees of customization. Make sure to review these potential limitations while choosing a model.

Recommended models

Custom

Design a unique experience for your users from scratch. A custom model enables you to create all of your skill's interactions.

What this skill type offers

- Built in voice interactions to stop, cancel, navigate to home, get help, and more.
- Customize your own visual and audio responses within your interaction model, with APL, or with Alexa

Provision your own

Provision your own endpoint and backend resources for your skill. This is recommended for skills that have significant data transfer requirements.

Things to know

- You're in full control of your endpoint and backend resources.
- No usage limits.
- You'll need your own AWS account
- You won't have access to the console's code editor.

Alexa-hosted (Node.js)

Alexa will host skills in your account and get you started with a Node.js template.

Things to know

- Get your skill up and running in less than a minute with free hosting across all Alexa regions.
- Unlimited Lambda calls, 25GB S3 storage, 250GB/month S3 throughput, and a single Dynamo table with 10M reads and writes.
- If you exceed usage limits, you can use your own AWS account later.

Alexa-hosted (Python)

Alexa will host skills in your account and get you started with a Python template.

Things to know

- Get your skill up and running in less than a minute with free hosting across all Alexa regions.
- Unlimited Lambda calls, 25GB S3 storage, 250GB/month S3 throughput, and a single Dynamo table with 10M reads and writes.
- If you exceed usage limits, you can use your own AWS account later.

Templates

Select a quick start template to get started with a skill that contains an interaction model and backend code that work together from the start. Templates come with a pre-built interaction model and default endpoint so that you can start testing as soon as the skill is created.

Choose a type of experience(s) to filter by

- Games & trivia View other templates

Start from Scratch

This skill gets you started with the required intents and with code demonstrating "Hello World" functionality if you are building an Alexa-hosted skill.

[Learn more](#)

By Alexa ⓘ

Fact Skill

Build an engaging fact skill about any topic. Alexa will select a fact at random and share it with the user when the skill is invoked. [Learn more](#)

Includes: custom intents, Personalization

By Alexa ⓘ

Scheduling Skill

Build a skill to allow users to schedule appointments on your calendar, receive email confirmations and reminders.

[Learn more](#)

Includes: voice permissions, reminders, API calls, session persistence

By Dabbie Lab ⓘ

Esto nos genera un código que a nosotros nos dará igual...

Lo que nos interesa es un **endpoint** que va a generar y que será el que enlazaremos con nuestra función

CUSTOM

- > Invocations
- > Interaction Model
- < Assets
- Slot Types (0)
- Multimodal Responses
- Tasks
- Interfaces
- Endpoint**
- Build History

Service Endpoint Type

Select how you will host your skill's service endpoint. Best practices in choosing lambda regions. [Learn more here](#)

AWS Lambda ARN (Recommended)

Your Skill ID: amzn1.ask.skill.fe6f2000-a912-4708-aa44-f87f56dd4ad2

Default Region (Required): North America

North America (Optional): arn:aws:lambda:us-east-1:<aws_account_id>:function:<lambda_name>

[Copy to Clipboard](#)

Volvemos a la función Lambda y vamos a quitar nuestro Trigger e incluir el personalizado generado ahora mismo

Lambda > Funciones > serverlessrepo-alexa-skil-alexaskillskitnodejsfact-Sjv9UPAEYtYF

Configuración general

Desencadenadores (1/1) [Información](#)

Corregir errores | Editar | Eliminar | Agregar desencadenador

Buscar desencadenadores

Desencadenador

Alexa: alexa-appkit.amazon.com

La verificación del ID de la habilidad no está configurada para este desencadenador. Recomendamos que elimine este desencadenador y agregue uno nuevo que tenga habilitada la verificación del ID de la habilidad. Obtenga más información.

Detalles

Agregamos un nuevo desencadenador de Alexa

Agregar desencadenador

Configuración del desencadenador [Información](#)

Alexa alexa iot voice

Elija un producto de Alexa

Kit de habilidades de Alexa (seleccionado) | Alexa Smart Home

La verificación del ID de la habilidad es un sencillo mecanismo para comprobar el identificador de habilidad de una solicitud de entrada procedente de una habilidad. Para configurarlo, especifique el ID de la habilidad que encontrará en el panel del conjunto de habilidades de Alexa (también denominado ID de la aplicación). [Obtenga más información](#).

Verificación del ID de la habilidad

Habilitar (recomendado) (seleccionado) | Deshabilitar

ID de la habilidad

amzn1.ask.skill.fe6f2000-a912-4708-aa44-f87f56dd4ad2

Lambda añadirá los permisos necesarios para Amazon Alexa para invocar la función Lambda desde este desencadenador. [Obtenga más información](#) sobre el modelo de permisos de Lambda.

[Cancelar](#) | **Agregar**

El siguiente paso es agregar el ARN de nuestra función con el Skill de Alexa

The screenshot shows the 'Información general de la función' (General Function Information) section. It displays the ARN of the Lambda function: arn:aws:lambda:us-east-1:772056227005:function:serverlessrepo-alexa-skills-kit-nodejs-factskill-Sjv9UPAEYtYF. Other details include the function name, description, last modified time, application, and URL.

Service Endpoint Type

Select how you will host your skill's service endpoint. Best practices in choosing lambda regions. [Learn more here](#)

AWS Lambda ARN (Recommended)

Your Skill ID amzn1.ask.skill.fe6f2000-a912-4708-aa44-f87f56dd4ad2 [Copy to Clipboard](#)

Default Region (Required) arn:aws:lambda:us-east-1:772056227005:function:serverlessrepo-

North America (Optional) arn:aws:lambda:us-east-1:<aws_account_id>:function:<lambda_name>

Guardamos y pulsamos en **Build Skill**

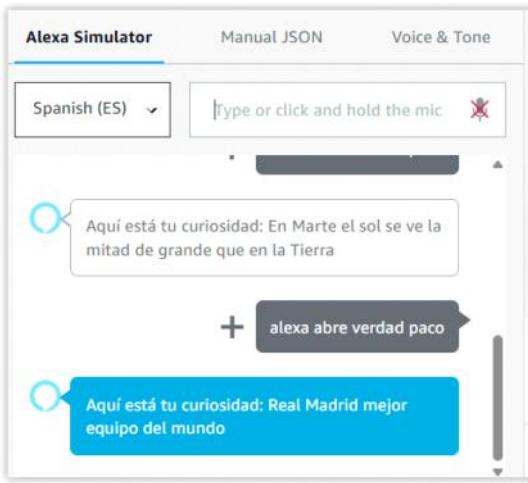
The screenshot shows the 'Invocations' section of the ASK build interface. It includes fields for 'Skill Invocation Name' (set to 'verdad paco'), 'Skill Launch Phrases', 'Intent Launch Phrases', and 'Interaction Model'. A note states: 'For example, if the invocation name is "daily horoscopes", users can say:' followed by examples like 'User: Alexa, open daily horoscopes' and 'User: Alexa, ask daily horoscopes for the horoscope for Gemini'. A note also says: 'Note: Not following the rules and guidelines mentioned below can result in certification failure.' A list of rules follows:

- One-word invocation names are allowed only when it is unique to your brand/intellectual property and proof of rights is provided in the testing instructions.
- Two-word invocation names should not contain articles ("a", "an" or "the") or preposition ("for", "to", "of", "about", "up", "by", "at", "off", "with"). For example, "a bicycle", "an espresso", "the amuse".

Una vez que tenemos todo listo, simplemente podemos invocarlo en **Test**

La forma de llamar a esto es mediante **Alexa abre NOMBRE DEL SKILL**

Y lo tenemos!!!



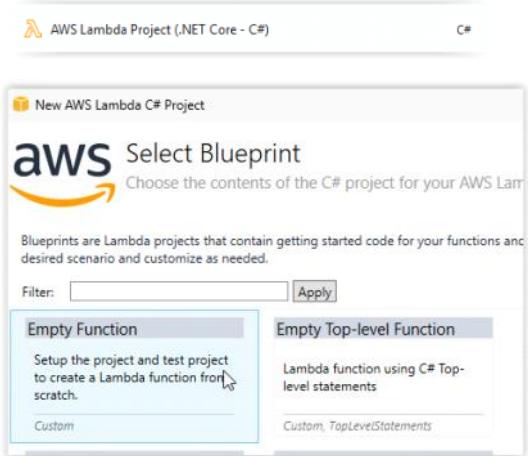
Lo siguiente que vamos a realizar es interactuar con nuestro nuevo juguete con una base de datos.

Lo que haremos será que leerá de un Servicio que vamos a crear con nuestro código...

- 1) Abrimos nuestro RDS y si lo tenemos activado, esperamos, sino, lo iniciamos.

Lo que vamos a realizar es una función Lambda que leerá de un SQL Server con películas y, Posteriormente publicaremos el Lambda y haremos las peticiones a Alexa.

Creamos una nueva aplicación AWS Lambda Project llamada **LambdaAlexaPelis**



Agregamos los siguientes Nuget

- Microsoft.EntityFrameworkCore** by aspnet, dotnetframework, EntityFrameworkCore 8.0.16
Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations. EF Core works wit...
- Microsoft.EntityFrameworkCore.SqlServer** by aspnet, dotnetframework 9.0.5
Microsoft SQL Server database provider for Entity Framework Core.
- Alexa.NET** by timheuer, 594K downloads 1.22.0
A simple .NET Core library for handling Alexa Skill request/responses.
- Amazon.Lambda.Serialization.Json** by awsdotnet, 32.3M downloads 2.2.4
Amazon Lambda .NET Core support - Serialization.Json package.

Sobre **Models** creamos una clase llamada **Pelicula**

PELICULA

```
public class Pelicula
{
    public int IdPelicula { get; set; }
    public string Genero { get; set; }
    public string Titulo { get; set; }
    public string Argumento { get; set; }
    public string Foto { get; set; }
    public string Actores { get; set; }
    public int Precio { get; set; }
    public string YouTube { get; set; }
}
```

```

public class PeliculasContext : DbContext
{
    0 references
    public PeliculasContext(DbContextOptions<PeliculasContext> options)
        : base(options) { }

    0 references
    public DbSet<Pelicula> Peliculas { get; set; }
}

```

<https://stackoverflow.com/questions/47888294/how-to-use-dependency-injection-in-aws-lambda-c-sharp-implementation>

PROBAMOS LA FUNCIONALIDAD DE INYECCION DE DEPENDENCIAS EN FUNCTION

```

public class Function
{
    private static ServiceProvider ServiceProvider { get; set; }

    public Function()
    {
        var services = new ServiceCollection();
        ConfigureServices(services);
        ServiceProvider = services.BuildServiceProvider();
    }

    private static void ConfigureServices(IServiceCollection services)
    {
        services.AddTransient<RepositoryPeliculas>();
        string connectionString = @"Data Source=sqltajamarpgs.database.windows.net;Initial Catalog=AZURETAJAMAR;Persist Security Info=True;User ID=admins;Password=Admin123;Trust Server Certificate=True";
        services.AddDbContext<PeliculasContext>(options => options.UseSqlServer(connectionString));
    }

    public async Task<Pelicula> FunctionHandler(ILambdaContext context)
    {
        var repo = ServiceProvider.GetService<RepositoryPeliculas>();
        Pelicula peli =
            await repo.FindPeliculaAsync(1);
        return peli;
    }
}

```

CON STARTUP (Gracias Jaime!!!)

Con Startup.cs

```

using LambdaAlexaPeliculas.Data;
using LambdaAlexaPeliculas.Repositories;
using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Configuration;
using System;

namespace LambdaAlexaPeliculas
{
    public class Startup
    {
        public static IServiceProvider ConfigureServices()
        {
            var services = new ServiceCollection();

            // CONFIGURACIÓN DEL BUILDER PARA LEER appsettings.json
            var config = new ConfigurationBuilder()
                .SetBasePath(Directory.GetCurrentDirectory())
                .AddJsonFile("appsettings.json", optional: true, reloadOnChange: true)
                .Build();

            // LEE LA CADENA DE CONEXIÓN
            string connectionString = config.GetConnectionString("AzureSql");

            // CONFIGURA EL DB CONTEXT
            services.AddDbContext<PeliculasContext>(options =>
                options.UseSqlServer(connectionString));

            // INYECCIÓN DEL REPOSITORIO
            services.AddTransient<RepositoryPeliculas>();

            return services.BuildServiceProvider();
        }
    }
}

[assembly:
LambdaSerializer(typeof(Amazon.Lambda.Serialization.SystemTextJson.DefaultLambdaJsonSerializer
))]

namespace LambdaAlexaPeliculas
{
    public class Function
    {
        private readonly RepositoryPeliculas repo;

        public Function()
        {
            var provider = Startup.ConfigureServices();
            this.repo = provider.GetService<RepositoryPeliculas>();
        }
    }
}

```

```

    }

    public async Task<List<Models.Pelicula>> FunctionHandler(object input, ILambdaContext
context)
    {
        return await this.repo.GetPeliculasAsync();
    }
}
}

```

Más estos nuggets.



El siguiente paso es configurar nuestra función para que reciba información de Alexa y la utilicemos dentro de

Nuestra función.

```

public class Function
{
    ILambdaLogger log;

    public async Task<SkillResponse>
        FunctionHandler(SkillRequest input, ILambdaContext context)
    {
        ServiceApiPeliculas service = new ServiceApiPeliculas();
        SkillResponse response = new SkillResponse();
        response.Response = new ResponseBody();
        response.Response.ShouldEndSession = false;
        IOutputSpeech innerResponse = null;
        this.log = context.Logger;
        log.LogLine($"Skill Request Object:" +
            + JsonConvert.SerializeObject(input));
        if (input.GetRequestType() == typeof(LaunchRequest))
        {
            innerResponse = new PlainTextOutputSpeech();
            (innerResponse as PlainTextOutputSpeech).Text =
                "Soy tu Alexa privado. Pideme una película...¿Qué número de Película quieres?";
        }
        else if (input.GetRequestType() == typeof(IntentRequest))
        {
            var intentRequest = (IntentRequest)input.Request;
            if (intentRequest.Intent.Name == "NUESTRO INTENT")
            {
                log.LogLine("PIDIENDO DATOS!!!!!!!");
                string slotJson = JsonConvert.SerializeObject(intentRequest.Intent.Slots);
                int idPelicula = GetSlotValue(slotJson);
                log.LogLine($"Id Pelí: " + idPelicula);
                log.LogLine($"Slots pelí: " + slotJson);
                Pelicula peli = await service.FindPelicula(idPelicula);
                if (peli != null)
                {
                    innerResponse = new PlainTextOutputSpeech();
                    (innerResponse as PlainTextOutputSpeech).Text =
                        peli.Argumento;
                }
                else
                {
                    innerResponse = new PlainTextOutputSpeech();
                    (innerResponse as PlainTextOutputSpeech).Text =
                        "No he encontrado tu Pelí " + idPelicula;
                }
            }
            else
            {
                innerResponse = new PlainTextOutputSpeech();
                (innerResponse as PlainTextOutputSpeech).Text =
                    "Ni idea de lo que me hablas";
            }
        }
        else
        {
            innerResponse = new PlainTextOutputSpeech();
            (innerResponse as PlainTextOutputSpeech).Text =
                "Ni idea de lo que me hablas, en else";
        }
    }

    response.Response.OutputSpeech = innerResponse;
    response.Version = "1.0";
    return response;
}

private int GetSlotValue(string dataJson)
{
    // string dataJson = "{ 'idpersonaje': { 'name': 'idpersonaje', " +
    // "'value': '2', 'confirmationStatus': 'NONE', 'source': 'USER', " +
    // "'slotValue': { 'type': 'Simple', 'value': '2' } } }";
    var jsonObject = JObject.Parse(dataJson);
    var data = ( JObject )jsonObject[ "idpelicula" ];
    var nombre = ( string )data[ "name" ];
    var id = ( string )data[ "value" ];
    return int.Parse( id );
}

private string GetSlotValueString(string dataJson)
{
    try
    {
        var jsonObject = JObject.Parse(dataJson);
        var data = ( JObject )jsonObject[ "idpelicula" ];
        log.LogLine($"Data " + data);
        var nombre = ( string )data[ "name" ];
        log.LogLine($"nombre " + nombre);
        var id = ( string )data[ "value" ];
    }
}

```

```

        log.LogLine($"Id " + id);
        return "Nombre " + nombre + ", Id: " + id;
    }
    catch (Exception ex)
    {
        log.LogLine($"Error Gordo... " + ex);
        return ex.ToString();
    }
}

```

Nos interesa cambiar el nombre del INTENT

```

        else if (input.GetRequestType() == typeof(IntentRequest))
    {
        var intentRequest = (IntentRequest)input.Request;
        if (intentRequest.Intent.Name == "preguntapelis")
        {
            log.LogLine("PIDIENDO DATOS!!!!!!!");

```

FUNCTION

```
[assembly: LambdaSerializer(typeof(Amazon.Lambda.Serialization.Json.JsonSerial
izer))]
```

```

public class Function
{
    ILambdaLogger log;
    private static ServiceProvider ServiceProvider { get; set; }

    public Function()
    {
        var services = new ServiceCollection();
        ConfigureServices(services);
        ServiceProvider = services.BuildServiceProvider();
    }

    private static void ConfigureServices(IServiceCollection services)
    {
        services.AddTransient<RepositoryPeliculas>();
        string connectionString = @"Data Source=sqltajamarpgs.database.windows
.net;Initial Catalog=AZURETAJAMAR;Persist Security Info=True;User ID=adminsqliq;
Password=Admin123;Trust Server Certificate=True";
        services.AddDbContext<PeliculasContext>
            (options => options.UseSqlServer(connectionString));
    }

    public async Task<SkillResponse>
        FunctionHandler(SkillRequest input, ILambdaContext context)
    {
        var repo = ServiceProvider.GetService<RepositoryPeliculas>();
        SkillResponse response = new SkillResponse();
        response.Response = new ResponseBody();
        response.Response.ShouldEndSession = false;
        IOoutputSpeech innerResponse = null;
        this.log = context.Logger;
        log.LogLine($"Skill Request Object: "
            + JsonConvert.SerializeObject(input));
        if (input.GetRequestType() == typeof(LaunchRequest))
        {
            innerResponse = new PlainTextOutputSpeech();
            (innerResponse as PlainTextOutputSpeech).Text =
                "Soy tu Alexa privado. Pideme una pelicula...¿Qué número de Película quieres?";
        }
        else if (input.GetRequestType() == typeof(IntentRequest))
        {
            var intentRequest = (IntentRequest)input.Request;
            if (intentRequest.Intent.Name == "preguntapelis")
            {
                log.LogLine("PIDIENDO DATOS!!!!!!!");
                string slotJson = JsonConvert.SerializeObject
                    (intentRequest.Intent.Slots);
                int idpelicula = GetSlotValue(slotJson);
                log.LogLine($"Id Peli: " + idpelicula);
                log.LogLine($"Slots peli: " + slotJson);
                Pelicula peli = await repo.FindPeliculaAsync(idpelicula);
                if (peli != null)
                {
                    innerResponse = new PlainTextOutputSpeech();
                    (innerResponse as PlainTextOutputSpeech).Text =
                        peli.Argumento;
                }
                else
                {
                    innerResponse = new PlainTextOutputSpeech();
                    (innerResponse as PlainTextOutputSpeech).Text =
                        "No he encontrado tu Peli " + idpelicula;
                }
            }
            else
            {
                innerResponse = new PlainTextOutputSpeech();
                (innerResponse as PlainTextOutputSpeech).Text =
                    "Ni idea de lo que me hablas";
            }
        }
        else
        {
            innerResponse = new PlainTextOutputSpeech();
            (innerResponse as PlainTextOutputSpeech).Text =
                "Ni idea de lo que me hablas, en else";
        }
        response.Response.OutputSpeech = innerResponse;
        response.Version = "1.0";
        return response;
    }

    private int GetSlotValue(string dataJson)
    {
        var jsonObject = JObject.Parse(dataJson);
        var data = (JObject)jsonObject["idpelicula"];
        var nombre = (string)data["name"];
    }
}

```

```

        var id = (string)data["value"];
        return int.Parse(id);
    }

    private string GetSlotValueString(string dataJson)
    {
        try
        {
            var jsonObject = JObject.Parse(dataJson);
            var data = (JObject)jsonObject["idpelicula"];
            log.LogLine($"Data " + data);
            var nombre = (string)data["name"];
            log.LogLine($"nombre " + nombre);
            var id = (string)data["value"];
            log.LogLine($"Id " + id);
            return "Nombre " + nombre + ", Id: " + id;
        }
        catch (Exception ex)
        {
            log.LogError($"Error Gordo... " + ex);
            return ex.ToString();
        }
    }
}

```

Incluimos permisos sobre CloudWatch al Role de Lambda

Puedes asociar hasta 10 políticas administradas.

Nombre de la política	Tipo	Entidades asociadas
AWSLambda_FullAccess	Administrada por AWS	5
CloudWatchFullAccess	Administrada por AWS	3

Dentro del fichero JSON de la aplicación, debemos indicar una nueva key llamada **function-role** apuntando a
Nuestro Role recién creado

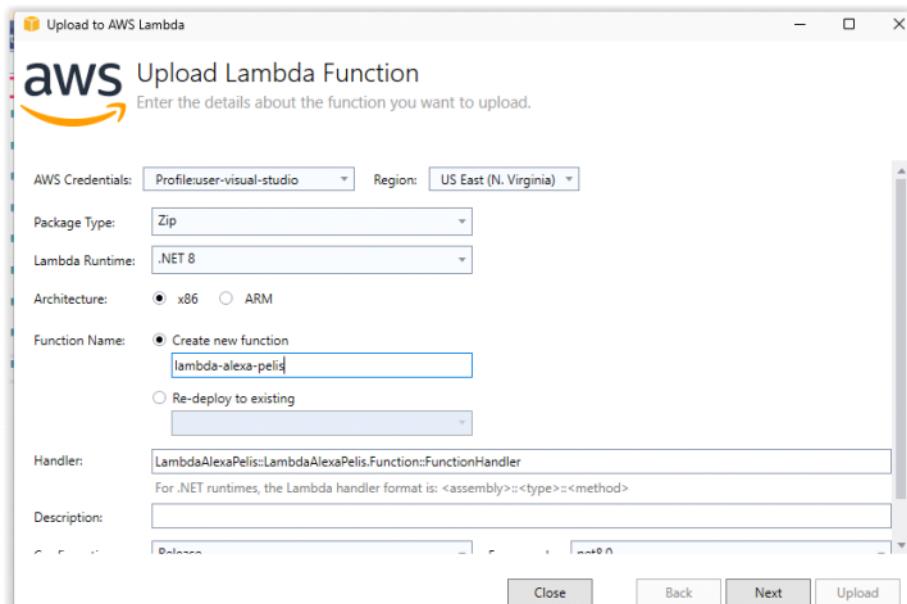
[role-execution-lambda-function](#)

```

"function-runtime": "dotnet8",
"function-memory-size": 512,
"function-timeout": 30,
"function-handler": "LambdaAlexaPelis::LambdaAlexaPelis.FunctionHandler",
"function-role": "role-execution-lambda-function"
}

```

Publicamos nuestra función incluyendo un nuevo usuario en nuestro Role



Abrimos el portal de Amazon Developer y creamos un nuevo Skill

<https://developer.amazon.com/>

The screenshot shows the 'Alexa Skills Kit' section of the Amazon Developer portal. At the top, there's a navigation bar with links for Dashboard, Apps & Services, Alexa, Login with Amazon, and Amazon Data. A dropdown menu for 'Alexa Skills Kit' is open, and a tooltip for 'Introducing the Small Business Program' is visible. Below the navigation, there are three numbered steps: 1. Name, Locale, 2. Experience, Model, Hosting service, and 3. Templates. Step 1 is currently selected. The main area is titled 'Enter a name for your skill. This is not the same as the skill invocation name, which you'll set up after you complete this step.' A 'Skill name' input field contains 'alexa pelis paco', with a character count of '16/50 characters'. A note below says: 'Brand names are only allowed if you provide proof of rights in the testing instructions or if you use the brand name in a referential way (for example, "I'm sorry, I don't have that information for you") or if you use the brand name for referential usage: unofficial, unauthorized, fan, fandom, for, about).' A large button at the bottom of this section says '+ Add Intent'.

2. Choose a primary locale

A locale refers to a language and the country in which it's spoken. Build your skill in your primary locale. You can always change it later.

Spanish (Spain)

El siguiente paso es crear un nuevo Intent.

Un Intent es la interacción que hacemos con Alexa y cómo queremos activar ciertos elementos.

Creamos un nuevo intent llamado **preguntapelis**

The screenshot shows the 'Intents' section of the Alexa Skills Kit. On the left, a sidebar has 'CUSTOM' selected and lists 'Invocations', 'Interaction Model', 'Intents (5)', 'Annotation Sets', 'Intent History', and 'JSON Editor'. 'Intents (5)' is currently active. The main area is titled 'Intents' and shows a list with a blue '+ Add Intent' button. Below the button is a 'NAME' input field. At the top right are 'Save', 'Evaluate model', and 'Update skill' buttons.

Add Intent

An intent represents an action that fulfills a user's spoken request. [Learn more](#) about intents.

Create custom intent ②

preguntapelis

Create custom intent

Use an existing intent from Alexa's built-in library ③

Intents / preguntapelis

Locale: Spanish (ES)

Sample Utterances (3)

Bulk Edit Export

What might a user say to invoke this intent?



dame una peli



Hola caracola



Hala Madrid



1 – 3 of 3

En el código hemos puesto "algo" llamado **idpelicula**

```
private int GetSlotValue(string dataJson)
{
    var jsonObject = JObject.Parse(dataJson);
    var data = (JObject)jsonObject["idpelicula"];
    var nombre = (string)data["name"];
    var id = (string)data["value"];
    return int.Parse(id);
}
```

Necesitamos enviar dicho **idpelicula** desde nuestro Intent

Intent Slots (1)

ORDER	NAME	SLOT TYPE	ACTIONS
1	idpelicula	AMAZON.NUMBER	Edit Dialog Delete
2	Create a new slot	Select a slot type	Edit Dialog Delete

Intents / preguntapelis

Sample Utterances (3)

dame la peli {idpelicula}

dame una peli

Hola caracola {idpelicula}

Hala Madrid {idpelicula}

A continuación, enlazamos nuestro Skill con Lambda

Copiamos el ARN del Endpoint del Skill

Endpoints here. You can host your own HTTP web service endpoint as long as the service meets the requirements described here.

Endpoint

- Build History
- MODELS**
- TOOLS**
 - Monetize Your Skill
 - Account Linking
 - Permissions

Service Endpoint Type

Select how you will host your skill's service endpoint. Best practices in choosing lambda regions. [Learn more here](#)

AWS Lambda ARN (Recommended)

Your Skill ID ? amzn1.ask.skill.2e42922e-4c92-4943-a648-111b3efda92f

Default Region ? (Required) arn:aws:lambda:us-east-1:318883658404:function:Reflec

Abrimos nuestro Lambda y creamos un nuevo Trigger

Agregar desencadenador

Configuración del desencadenador Información

Alexa alexa iot voice

Elige un producto de Alexa

Kit de habilidades de Alexa Alexa Smart Home

La verificación del ID de la habilidad es un sencillo mecanismo para comprobar el identificador de habilidad de una solicitud de entrada procedente de una habilidad. Para configurarlo, especifique el ID de la habilidad que encontrará en el panel del conjunto de habilidades de Alexa (también denominado ID de la aplicación). [Obtenga más información](#)

Verificación del ID de la habilidad

Habilitar (recomendado) Deshabilitar

ID de la habilidad

amzn1.ask.skill.2e42922e-4c92-4943-a648-111b3efda92f

Copiamos el ARN de la función Lambda y lo asociamos con el Skill

lambda-alex-pelis

Se agregó correctamente el desencadenador de 2e42922e-4c92-4943-a648-111b3efda92f a la función lambda-alex-pelis. Ahora, la función recibe eventos del desencadenador.

▼ Información general de la función Información

Diagrama | Plantilla

Descripción

-

Última modificación
hace 32 minutos

ARN de la función
arn:aws:lambda:us-east-1:77205622:5:function:lambda-alex-pelis

URL de la función | Información

-

Service Endpoint Type

Select how you will host your skill's service endpoint. Best practices in choosing lambda regions. [Learn more here](#)

AWS Lambda ARN ?
(Recommended)

Your Skill ID ?

amzn1.ask.skill.2e42922e-4c92-4943-a648-111b3efda92f
Copy to Clipboard

Default Region ?
(Required)

da:us-east-1:772056227005:function:lambda-alexa-pelis

North America ?
(Optional)

arn:aws:lambda:us-east-1:<aws_account_id>:function:<lambda-alexa-pelis>

Realizamos un Build Skill y a jugar o llorar

Y lo tenemos!!!!

alexa developer console

All ? Search

Your Skills alexa pelis paco Build Code Test Distribution Certification Analytics

Skill testing is enabled in: Development Development Skill I/O Device Display Device Log

Alexa Simulator Manual JSON Voice & Tone

Spanish (ES) Type or click and hold the mic

Soy tu Alexa privado. Pideme una película...
¿Qué número de Película quieres?

+ hala madrid quince

Franky es una ladrón de diamantes que tiene que entregar una enorme pieza a su jefe Avi, pero en el camino es tentado por Boris para apostar en un combate ilegal de boxeo. En realidad, es una trampa para atracarle por lo

Personalization

```
1 [{ 2   "version": "1.0", 3     "session": { 4       "new": false, 5       "sessionId": "amzn1.echo-api.session.c", 6       "application": { 7         "applicationId": "amzn1.ask.skill.", 8         "attributes": {}, 9         "user": { 10           "userId": "amzn1.ask.account.AMAYV", 11         }, 12         "affiliatedResources": [] 13       }, 14       "context": { 15         "viewports": [ 16           { 17             "type": "APL", 18             "id": "medHub", 19           } 20         ] 21       } 22     } 23   }]
```

AWS ELASTICACHE

martes, 27 de mayo de 2025 11:37

Es la base de datos temporal de AWS para múltiples dispositivos.

Lo que contiene en su interior es un Cluster que puede ser replicado en múltiples zonas.

Tenemos dos tipos:

- 1) ElastiCache Redis: Es lo mismo que en Azure
- 2) ElastiCache Memcached: Es lo mismo que el anterior solamente que el Cluster Interno funciona de forma más eficiente. Este tipo de Cache se puede combinar Con más bases de datos temporales.

Un "problemita" que tenemos es que No podemos utilizar Cache fuera del entorno De AWS.

Para probar esto, vamos a realizar el siguiente laboratorio:

- 1) App MVC para almacenar objetos favoritos dentro de Cache Redis.
- 2) Una máquina EC2 para publicar esta App
- 3) Un Servicio ElastiCache para comprobar toda la funcionalidad en producción.

Comenzamos desplegando una nueva máquina EC2 llamada **ec2-elastic-cache**
De tipo **Amazon Linux 2** con el Bash

Creamos un grupo y abrimos HTTP y HTTPS

Nos conectamos a la máquina y habilitamos HTTPS

```
cd /etc/pki/tls/certs  
sudo ./make-dummy-cert localhost.crt
```

Abrimos en AWS ElastiCache

The screenshot shows the AWS ElastiCache service page. On the left, there's a sidebar with 'Panel' and two main sections: 'Recursos' (Resources) and 'Configuraciones' (Configurations). Under 'Recursos', 'Cachés de Valkey' is selected and highlighted with a blue border. The main content area has a dark header 'Servicios de bases de datos' and a large title 'Amazon ElastiCache'. Below it, the sub-header 'Rendimiento en tiempo real para aplicaciones en tiempo real' is displayed. A descriptive paragraph follows, mentioning Valkey, Memcached, and Redis OSS. To the right, there are two boxes: 'Introducción' (Introduction) which says 'Cree una caché sin servidores' (Create a cache without servers) and 'Precios' (Prices) which lists 'Amazon ElastiCache anual' (Annual price). At the bottom, a large form titled 'Configuración' (Configuration) is shown. It asks to 'Elija una de las siguientes opciones para crear una nueva caché.' (Select one of the following options to create a new cache). The 'Motor' (Engine) section has three radio buttons: 'Valkey - recomendado' (Valkey - recommended), 'Memcached', and 'Redis OSS', with 'Valkey' selected. The 'Opción de implementación' (Implementation option) section has two radio buttons: 'Sin servidor' (No server) and 'Diseñe su propia caché' (Design your own cache), with 'Diseñe su propia caché' selected. The 'Método de creación' (Creation method) section has three radio buttons: 'Creación sencilla' (Simple creation), 'Caché del clúster' (Cluster cache), and 'Restauración a partir de copias de seguridad' (Restore from backup), with 'Caché del clúster' selected. A note below 'Caché del clúster' says: 'Establezca todas las opciones de configuración correspondientes al nuevo clúster.' (Set all configuration options for the new cluster).

Modo de clúster

Escale el clúster de manera dinámica sin tiempo de inactividad.

Habilitado

El modo de clúster permite la replicación en varias particiones para mejorar la escalabilidad y la disponibilidad.

Desactivado

El clúster de Valkey tendrá una única partición (grupo de nodos) con un nodo principal y hasta 5 réplicas de lectura.

💡 Si elige desactivar el modo de clúster, no podrá cambiar el número de particiones. La configuración admite todos los comandos y funcionalidades de Valkey, pero limita el tamaño y el rendimiento máximos de la caché. [Más información](#)

Información del clúster

Utilice las siguientes opciones para configurar el clúster.

Nombre

cache-coches

El nombre puede tener hasta 40 caracteres y no debe contener espacios.

Descripción - opcional

Descripción

Ubicación

Elija si desea alojar el clúster en la nube de AWS o en las instalaciones.

Ubicación

Nube de AWS

Utilice la nube de AWS para las instancias de ElastiCache.

En las instalaciones

Cree instancias de ElastiCache en un Outpost (a través de AWS Outposts). Debe crear primero un ID de subred en un Outpost.

Multi-AZ

Habilitar

Multi-AZ proporciona una alta disponibilidad mejorada a través de la comutación por error automática a una réplica de lectura, entre zonas de disponibilidad, en caso de comutación por error de un nodo principal.

Commutación por error automática

Habilitar

La comutación por error automática de ElastiCache proporciona una alta disponibilidad mejorada a través de la comutación por error automática a una réplica de lectura en caso de que se produzca una comutación por error en el nodo principal.

Configuración de la caché

Utilice las siguientes opciones para configurar el clúster.

Versión del motor

Compatibilidad de la versión del motor de Valkey que se ejecutará en los nodos.

8.0



Puerto

El número de puerto en el que los nodos aceptan conexiones.

6379



Grupos de parámetros

Los grupos de parámetros controlan las propiedades del tiempo de ejecución de los nodos y clústeres.

default.valkey8



Tipo de nodo

El tipo de nodo a implementar y su tamaño de memoria asociado.

cache.t3.micro

0.5 GiB memory Up to 5 Gigabit network performance



Número de réplicas

Ingrese el número de réplicas entre 0 y 5. Cero réplicas no habilitarán un clúster mejorado con roles principal o de réplica.

1

Conectividad

Elija las versiones de IP que admitirá este clúster. A continuación, seleccione un grupo de subredes existente o cree uno nuevo.

Tipo de red

Elija entre IPv4, pila doble e IPv6

IPv4

Sus recursos se comunicarán únicamente a través del protocolo IPv4.



Grupos de subredes

Elija un grupo de subredes existente

Cree un nuevo grupo de subredes

Nombre

grupo-cache-redis

El nombre puede tener hasta 255 caracteres y no debe contener espacios.

Descripción - opcional

Descripción

Vamos a habilitar solamente la red para 1e (mi máquina)

Instancias (1) [Información](#) Última actualización Hace less than a minute [Conectar](#) [Estado de la instancia](#) ▾ [Acciones](#) ▾ [Lanz.](#)

Buscar Instancia por atributo o etiqueta (case-sensitive)

[Todos los estados](#) ▾

inst...	Comprobación de	Estado de la al.	Zona de dispon...	DNS de IPv4 pública	Dirección IP...	IP elá...
2/2 comprobador	Ver alarmas +	us-east-1e	ec2-18-204-8-113.com...	18.204.8.113	-	-

Administrar subredes

Agregue o elimine subredes de la tabla siguiente.

Subredes (2/6)

Buscar subredes

<input type="checkbox"/>	Zona de disponibilidad	ID de subred	Nombre de subred	Bloque de CIDR
<input type="checkbox"/>	us-east-1a	subnet-0f0e0beb5f9484911	-	172.31.32.0/20
<input type="checkbox"/>	us-east-1b	subnet-0aaf692a83dad60f	-	172.31.0.0/20
<input type="checkbox"/>	us-east-1c	subnet-0ceeca22a653b9ded	-	172.31.80.0/20
<input type="checkbox"/>	us-east-1d	subnet-0b12db78543a57de2	-	172.31.16.0/20
<input checked="" type="checkbox"/>	us-east-1e	subnet-0f1c54ffe09094cb4	-	172.31.48.0/20

[Cancelar](#) [Elegir](#)

Ubicación de zonas de disponibilidad

Utilice los siguientes campos para configurar las ubicaciones correspondientes a las zonas de disponibilidad.

Ubicación de zonas de disponibilidad

Al ubicar nodos en diferentes zonas de disponibilidad, reduce la posibilidad de que un error en una zona de disponibilidad, como un corte de energía, provoque un error en todo el sistema. Elija **Specify Availability Zones** (Especificar zonas de disponibilidad) si desea especificar zonas de disponibilidad para los nodos del clúster.

Especificar zonas de disponibilidad

Réplicas | Región

Principal	us-east-1e
Réplica 1	us-east-1e

Seguridad

Utilice la siguiente sección para configurar la seguridad de la red y de los datos para el clúster.

Cifrado en reposo

Habilitar

Habilita el cifrado de los datos almacenados en el disco.

Cifrado en tránsito | [Información](#)

Habilitar

Habilita el cifrado de datos que se mueven entre el servicio y el cliente.

Seleccionado grupos de seguridad (0)

[Administrar](#)

Un grupo de seguridad actúa como un firewall que controla el acceso a la red a los clústeres.

ID de grupo

▲ | Nombre

▼

Seleccionamos el grupo de seguridad de nuestra máquina

Administrar grupos de seguridad

Grupos de seguridad (1/2)

Buscar grupos de seguridad		
ID del grupo de seguridad	Nombre del grupo de seguridad	Descripción
<input checked="" type="checkbox"/> sg-084618ea499cef90f	grupo-seguridad-elastic	launch-wizard created 2025-05-27T
<input type="checkbox"/> sg-0e32dc63137690df7	default	default VPC security group

Cancelar

Elegir

Copia de seguridad

Puede utilizar copias de seguridad para restaurar un clúster o propagar un nuevo clúster. La copia de seguridad se compone de los metadatos del clúster, junto con todos los datos del clúster.

Habilitar copias de seguridad automáticas

ElastiCache creará automáticamente una copia de seguridad diaria de un conjunto de réplicas.

Mantenimiento

Configure los ajustes de mantenimiento correspondientes al clúster.

Período de mantenimiento

Especifique el intervalo de tiempo (UTC) para las actualizaciones, como por ejemplo la aplicación de parches a un sistema operativo, la actualización de controladores y la instalación de software o parches.

Sin preferencia

Especificar el periodo de mantenimiento

Actualización automática de versiones secundarias

Habilitar

Programe automáticamente la actualización del clúster a la versión secundaria más reciente, una vez que esta se encuentre disponible. La actualización del clúster solo se programará durante el periodo de mantenimiento.

Tema para la notificación de Amazon SNS

Elija un tema SNS de la lista o ingrese el nombre de recurso de Amazon (ARN) de un tema existente. Si no elige ningún tema, no se enviarán notificaciones.

[Desactivar notificaciones](#)

Registros

Especifique si desea proporcionar los registros lentos de Valkey o los registros del motor.

Registros lentos

Habilitar

Proporcione el registro lento para las consultas que superen un tiempo de puesta en marcha especificado.

Registros del motor

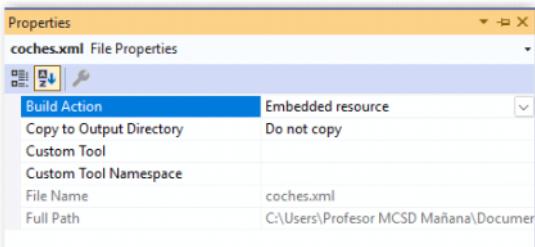
Habilitar

Proporcione el registro del motor para las consultas que superen un tiempo de ejecución especificado.

Creamos nuestra aplicación llamada **MvcAWSElastiCache**

Copiamos el fichero **coches.xml** dentro de una carpeta llamada **Documents**

Ponemos el fichero como **Recursos incrustado**



Sobre **Models** creamos una nueva clase llamada **Coche**

COCHE

```
public class Coche
{
    0 references
    public int IdCoche { get; set; }
    0 references
    public string Marca { get; set; }
    0 references
    public string Modelo { get; set; }
    0 references
    public string Imagen { get; set; }
}
```

Sobre **Repositories** creamos una carpeta llamada **RepositoryCoches**

REPOSITORYCOCHES

```
public class RepositoryCoches
{
    private XDocument document;

    public RepositoryCoches()
    {
        string path = "MvcAWSElastiCache.Documents.coches.xml";
        Stream stream = this.GetType().Assembly
            .GetManifestResourceStream(path);
        this.document = XDocument.Load(stream);
    }

    public List<Coche> GetCoches()
    {
        var consulta = from datos in this.document.Descendants("coche")
            select new Coche
            {
                IdCoche = int.Parse(datos.Element("idcoche").Value),
                Marca = datos.Element("marca").Value,
                Modelo = datos.Element("modelo").Value,
                Imagen = datos.Element("imagen").Value
            };
        return consulta.ToList();
    }

    public Coche FindCoche(int idcoche)
    {
        return this.GetCoches().FirstOrDefault(x => x.IdCoche == idcoche);
    }
}
```

Sobre **Controllers** creamos un nuevo controlador llamado **CochesController**

COCHESCONTROLLER

```
public class CochesController : Controller
{
    private RepositoryCoches repo;

    public CochesController(RepositoryCoches repo)
    {
        this.repo = repo;
    }

    public IActionResult Index()
    {
        List<Coche> coches = this.repo.GetCoches();
        return View(coches);
    }

    public IActionResult Details(int id)
    {
        Coche car = this.repo.FindCoche(id);
        return View(car);
    }
}
```

Creamos nuestras vistas

Resolvemos las dependencias en **Program**

PROGRAM

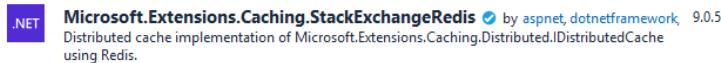
```
// Add services to the container.
builder.Services.AddTransient<RepositoryCoches>();
builder.Services.AddControllersWithViews();
```

El siguiente paso es utilizar Cache Redis como favorito.

La programación es casi igual a Azure.

También recordemos que los Redis almacenan las Keys por usuario, es decir, Si yo no "personalizo" las Keys, todos los usuarios verán lo mismo.

Agregamos los siguientes Nuget



La mayor ventaja es que no necesitamos permisos ni nada, simplemente la cadena de Conexión a Redis.

Creamos una carpeta llamada **Helpers** y una clase llamada **HelperCacheRedis**

HELPERCACHEREDIS

```
public class HelperCacheRedis
{
    private static Lazy<ConnectionMultiplexer>
        CreateConnection = new Lazy<ConnectionMultiplexer>(() =>
    {
        //AQUI IRA NUESTRA CADENA DE CONEXION
        return ConnectionMultiplexer.Connect("");
    });

    public static ConnectionMultiplexer Connection
    {
        get
        {
            return CreateConnection.Value;
        }
    }
}
```

El siguiente paso es crear una nueva carpeta **Services** y una clase llamada **ServiceCacheRedis**

```
public class ServiceCacheRedis
{
    private IDatabase cache;

    public ServiceCacheRedis()
    {
        this.cache = HelperCacheRedis.Connection.GetDatabase();
    }

    public async Task<List<Coche>> GetCochesFavoritosAsync()
    {
        //TENDREMOS ALMACENADO EN CACHE UN JSON DE COLECCION
        //DE COCHES
        string jsonCoches = await
            this.cache.StringGetAsync("favoritos");
        if (jsonCoches == null)
        {
            return null;
        }
        else
        {
            List<Coche> cars =
                JsonConvert.DeserializeObject<List<Coche>>(jsonCoches);
            return cars;
        }
    }

    public async Task AddFavoritoAsync(Coche car)
    {
        List<Coche> favoritos = await this.GetCochesFavoritosAsync();
        if (favoritos == null)
        {
            favoritos = new List<Coche>();
        }
        favoritos.Add(car);
        string jsonCoches = JsonConvert.SerializeObject(favoritos);
        await this.cache.StringSetAsync("favoritos", jsonCoches
            , TimeSpan.FromMinutes(30));
    }

    public async Task DeleteFavoritoAsync(int idcoche)
    {
        List<Coche> cars = await this.GetCochesFavoritosAsync();
        if (cars != null)
        {
            Coche carDelete = cars.FirstOrDefault(x => x.IdCoche == idcoche);
            cars.Remove(carDelete);
            if (cars.Count == 0)
            {
                await this.cache.KeyDeleteAsync("favoritos");
            }
        }
    }
}
```

```

        else
    {
        string jsonCoches = JsonConvert.SerializeObject(cars);
        await this.cache.StringSetAsync("favoritos",
            jsonCoches, TimeSpan.FromMinutes(30));
    }
}
}

```

Incluimos el servicio dentro de Program

PROGRAM

```

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddTransient<RepositoryCoches>();
builder.Services.AddTransient<ServiceCacheRedis>();
builder.Services.AddControllersWithViews();

```

Implementamos la funcionalidad dentro de **CochesController**

COCHESCONTROLLER

```

public class CochesController : Controller
{
    private RepositoryCoches repo;
    private ServiceCacheRedis service;

    public CochesController
        (RepositoryCoches repo,
        ServiceCacheRedis service)
    {
        this.repo = repo;
        this.service = service;
    }

    public async Task<IActionResult> Favoritos()
    {
        List<Coche> cars = await this.service.GetCochesFavoritosAsync();
        return View(cars);
    }

    public async Task<IActionResult> SeleccionarFavorito(int idcoche)
    {
        //BUSCAMOS EL COCHE EN XML
        Coche car = this.repo.FindCoche(idcoche);
        await this.service.AddFavoritoAsync(car);
        return RedirectToAction("Favoritos");
    }

    public async Task<IActionResult> DeleteFavorito(int idcoche)
    {
        await this.service.DeleteFavoritoAsync(idcoche);
        return RedirectToAction("Favoritos");
    }
}

```

```

</div>
<a asp-controller="Coches"
asp-action="SeleccionarFavorito"
asp-route-idcoche="@coche.IdCoche"
class="btn btn-warning">
    Favorito
</a>
</div>
...

```

Recuperamos la cadena de conexión a nuestro Cache Redis

cache-coches Información

▼ Detalles de clúster

Nombre del clúster cache-coches	Descripción -	Tipo de nodo cache.t3.micro
Motor Valkey	Versión del motor 8.0.1	Almacén de datos global -
Estado de actualización Hasta la fecha	Modo de clúster Desactivado	Particiones 1
Organización de datos en niveles Desactivado	Multi-AZ Desactivado	Conmutación por error automá Desactivado
Cifrado en reposo Desactivado	Grupo de parámetros default.valkey8	ARN de Outpost -
Punto de conexión principal  cache-coches.nilqz5.ng.0001.use1.cache.amazonaws.com:6379	Punto de conexión del lector  cache-coches-ro.nilqz5.ng.0001.use1.cache.amazonaws.com:6379	ARN  arn:aws:elasticache:us-east-05:replicationgroup:cache-coche

Referencia

```
public class HelperCacheRedis
{
    private static Lazy<ConnectionMultiplexer>
        CreateConnection = new Lazy<ConnectionMultiplexer>(() =>
    {
        string connectionString = @"cache-coches.nilqz5.ng.0001.us
//AQUI IRA NUESTRA CADENA DE CONEXION
        return ConnectionMultiplexer.Connect(connectionString);
    });
}
```

Al intentar conectar, no podremos hacerlo...

```
private static Lazy<ConnectionMultiplexer>
CreateConnection = new Lazy<ConnectionMultiplexer>(() =>
{
    string connectionString = @"cache-coches.nilqz5.ng.0001.use1.cac
//AQUI IRA NUESTRA CADENA DE CONEXION
    return ConnectionMultiplexer.Connect(connectionString);
```

Exception User-Unhandled



StackExchange.Redis.RedisConnectionException: 'It was not possible to connect to the redis server(s). Error connecting right now. To allow this multiplexer to continue retrying until it's able to connect, use abortConnect=false in your connection string or AbortOnConnectFail=false; in your code.'

Debemos desplegar nuestra App en la máquina EC2

Al desplegar la aplicación, tendremos el mismo Error, no se puede conectar...

An unhandled exception occurred while processing the request.

RedisConnectionException: It was not possible to connect to the redis server(s). Error connecting right now. To allow this multiplexer to connect, use abortConnect=false in your connection string or AbortOnConnectFail=false; in your code.

StackExchange.Redis.ConnectionMultiplexer.ConnectImpl(ConfigurationOptions configuration, TextWriter writer, Nullable<ServerType> serverType, EndPointCollection endpoints)
ConnectionMultiplexer.cs, line 723

Stack Query Cookies Headers Routing



RedisConnectionException: It was not possible to connect to the redis server(s). Error connecting right now. To allow this multiplexer to connect, use abortConnect=false in your connection string or AbortOnConnectFail=false

StackExchange.Redis.ConnectionMultiplexer.ConnectImpl(ConfigurationOptions configuration, TextWriter writer, Nullable<ServerType> serverType, EndPointCollection endpoints)
ConnectionMultiplexer.cs

Reglas de entrada Información

ID de la regla del grupo de seguridad	Tipo	Información	Protocolo	Intervalo de puertos	Origen	Información	Descripción: opcional
Información							
sgr-031c13650ee48a313	HTTP		TCP	80	P...	Q	<input type="text"/> 0.0.0.0/0 X
sgr-0ce3735c3fc8782b1	HTTPS		TCP	443	P...	Q	<input type="text"/> 0.0.0.0/0 X
sgr-0015a98ad4db0072c	SSH		TCP	22	P...	Q	<input type="text"/> 0.0.0.0/0 X
-	TCP personalizado		TCP	6379	A...	Q 0.0.0.0/0	<input type="text"/> 0.0.0.0/0 X

Y ya tendremos nuestra aplicación perfecta!!!

Volvemos a realizar un **dotnet run** (Gracias Mario!!!)

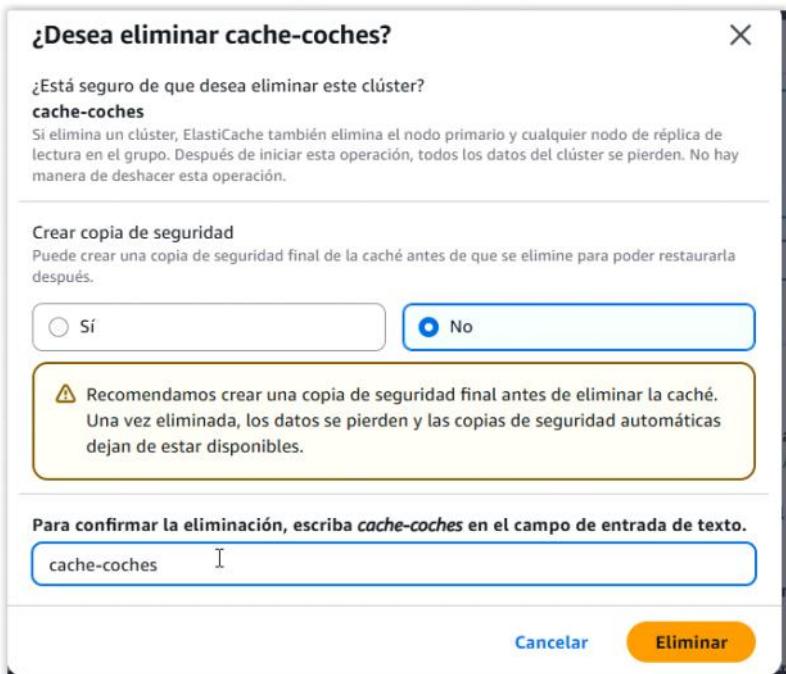
 Home Elasti Cache Privacy

Coches favoritos Cache Redis AWS

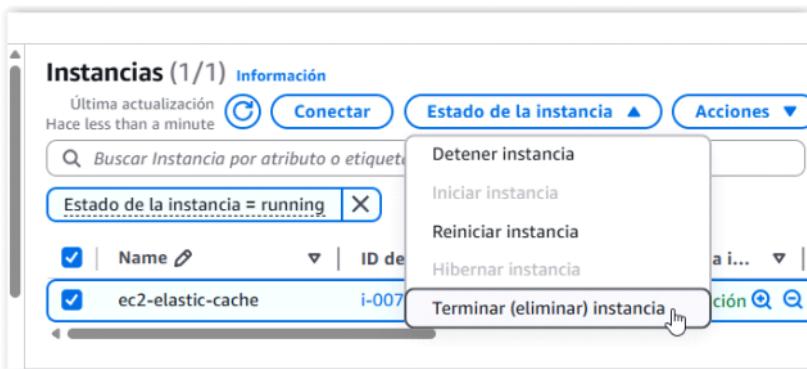

DMC DELOREAN
[Delete favorito](#)

Super importante:

- 1) Eliminamos Cache Redis y le indicamos que NO queremos copias de seguridad



2) Eliminamos nuestro EC2



3) Quitamos a nuestro usuario **user-visual** del grupo en IAM

4) Eliminar el VPC