[ENGLISH]

Hello, Cloud Gurus,

and welcome to this lesson,

where we'll be getting our hands dirty using AWS SAM.

And we'll begin by launching the AWS CloudShell

because the SAM CLI comes pre-installed in the CloudShell.

Next, we'll create an S3 bucket,

and we'll use sam package

to package our code and save it in S3.

And then, finally, we're going to use sam deploy

to deploy a Lambda function using CloudFormation.

So if you'd like to join me in the console,

we'll get started using AWS SAM.

So, from the console,

we are going to run all of the commands for this lesson

using the AWS CloudShell.

So select CloudShell from up here.

We'll just close that down.

And while the environment is starting up,

I'm just going to show you a couple of files

that we will need for this lesson,

and these are all in the GitHub repository

that is linked as a resource for this lesson.

So we've got hello-from-lambda.js,

which is our function code,

and template.yml, which is our SAM template.

So let's take a look at the function code.

And this is just a simple function,

and it's going to return this message

when we launch the function.

And if I hit Back on the browser,

here's our template.yml,

and this is our SAM template.

So, first of all, we've got our template version,

we've got a description.

This transform section just tells CloudFormation

that we are using the serverless application model, or SAM.

Then, under Resources,

this is where our function is defined.

So here's our Lambda function.

The Runtime is nodejs.

Here's the Architecture, so x86 architecture,

MemorySize, and we've got a Timeout, as well,

so the function needs to complete within 100 seconds.

Here's the Description.

And under Policies,

the Lambda function is just getting the basic permissions,

and this will allow our Lambda function

to send logs to CloudWatch and nothing else.

And then, if we hit Back on the browser,

I've also included a text file

with all the commands that we need to run

from the CloudShell.

So the first thing we're going to do is copy our 2 files.

Here's the command to do that. I'm just going to copy that.

And then, from the CloudShell, just clear the screen,

and paste in the first command.

And this curl command just transfers the file from GitHub,

and it will save it locally in our CloudShell,

so hit Enter.

Now we'll download the second file

and hit Enter.

You type ls.

You should see that you've got both of those files

saved to your CloudShell.

So now, the next thing that we need to do

is create the S3 bucket that's going to store our code in.

So clear the screen, and type aws s3 mb

for make bucket,

then s3://.

And now we need to give our bucket a unique name.

So I'm going to call it cfsambucket,

and then just add some random numbers on the end

and hit Enter.

Then, if you run aws s3 ls, there is our bucket.

So now, we're ready to package and deploy our code.

And these are the commands that we need.

And this first command is just going to package our file,

so it's going to package our template.yml.

The output is going to be called sam-template.yml,

and it's going to store the package in our bucket,

but we will need to replace this bucket name

with the name of the bucket that we just created.

So, back in your CloudShell, paste that command,

and then just update the name of your bucket

so that it matches the one that you just created,

and hit Enter.

And if it's all worked, then this is what you should see.

And then, finally, to deploy the code,

this is the command we're going to use.

So copy the command. Come back to your CloudShell.

And in this command,

we are deploying the package we just created.

So we're using sam deploy.

Here's our template file;

the CloudFormation stack that it creates

is going to be called mystack.

This just allows CloudFormation to create IAM resources,

and when it creates the function,

it's also going to create an execution role

for our function, as well,

so we need to make sure that we give CloudFormation

the ability to do that,

and that's what this flag here does.

So, if you're happy with that, just hit Enter,

and it might just take a few moments to complete.

And if it's all been successful,

then this is what you should see.

And under the ResourceStatus,

we can see that it's created the IAM role,

and it's created the Lambda function,

and it has finished deploying the CloudFormation stack.

So now that's all completed,

we can head across to Lambda

and take a look at our function.

So in the search box, type lambda.

Here's our function.

If we scroll down and select Test,

we can create a new test event.

I'm going to call it te1,

and save, and then select Test.

So it's executed our function,

and here is the response from our function.

And then, if we search for CloudFormation,

select our stack,

and then select Resources,

here's the function that was created,

and here is the function execution role.

And we created all of that really quickly and easily

just using a couple of AWS SAM commands.

So, in summary, we used the AWS CloudShell

to run our SAM commands

because the SAM CLI is pre-installed.

We created an S3 bucket.

We used sam package to package up our code,

and it automatically saved it to the S3 bucket.

And then we used sam deploy

to deploy a Lambda function using CloudFormation.

So that is it for this lesson.

If you have any questions, please let me know.

Otherwise, I will see you in the next lesson.

Thank you.

[SPANISH]

Hola, gurús de la nube,

y bienvenidos a esta lección,

donde nos ensuciaremos las manos con AWS SAM.

Y comenzaremos lanzando AWS CloudShell

porque la CLI de SAM viene preinstalada en CloudShell .

A continuación, crearemos un depósito S3,

y usaremos el paquete sam

para empaquetar nuestro código y guardarlo en S3.

Y luego, finalmente, vamos a usar sam deployment

para implementar una función de Lambda mediante CloudFormation .

Entonces, si desea unirse a mí en la consola,

Comenzaremos a usar AWS SAM.

Entonces, desde la consola,

vamos a ejecutar todos los comandos para esta lección

utilizando AWS CloudShell .

Así que seleccione CloudShell desde aquí arriba.

Solo cerraremos eso.

Y mientras el entorno se pone en marcha,

solo les voy a mostrar un par de archivos

que necesitaremos para esta lección,

y estos están todos en el repositorio de GitHub

que está vinculado como recurso para esta lección.

Así que tenemos hello-from-lambda.js,

cuál es nuestro código de función,

y template.yml , que es nuestra plantilla SAM.

Así que echemos un vistazo al código de la función.

Y esto es solo una función simple,

y va a devolver este mensaje

cuando lanzamos la función.

Y si presiono Atrás en el navegador,

aquí está nuestra plantilla.yml ,

y esta es nuestra plantilla SAM.

En primer lugar, tenemos nuestra versión de plantilla,

tenemos una descripción.

Esta sección de transformación solo le dice a CloudFormation

que estamos utilizando el modelo de aplicación sin servidor , o SAM.

Luego, en Recursos,

aquí es donde se define nuestra función.

Así que aquí está nuestra función Lambda.

El tiempo de ejecución es nodejs .

Aquí está la arquitectura, por lo que la arquitectura x86,

MemorySize , y también tenemos un Timeout,

por lo que la función debe completarse en 100 segundos.

Aquí está la descripción.

Y en Políticas,

la función Lambda solo obtiene los permisos básicos,

y esto permitirá que nuestra función Lambda

para enviar registros a CloudWatch y nada más.

Y luego, si presionamos Atrás en el navegador,

También he incluido un archivo de texto.

con todos los comandos que necesitamos ejecutar

de CloudShell . \_

Así que lo primero que vamos a hacer es copiar nuestros 2 archivos.

Aquí está el comando para hacer eso. Solo voy a copiar eso.

Y luego, desde CloudShell , simplemente borre la pantalla,

y pegar en el primer comando.

Y este comando curl simplemente transfiere el archivo desde GitHub ,

y lo guardará localmente en nuestro CloudShell ,

así que presione Entrar.

Ahora vamos a descargar el segundo archivo.

y presione Entrar.

Escribes ls .

Deberías ver que tienes ambos archivos.

guardado en su CloudShell .

Así que ahora, lo siguiente que tenemos que hacer

es crear el depósito S3 que va a almacenar nuestro código.

Así que borre la pantalla y escriba aws s3 mb

para hacer balde,

luego s3://.

Y ahora necesitamos darle a nuestro cubo un nombre único.

Así que voy a llamarlo cfsambucket ,

y luego simplemente agregue algunos números aleatorios al final

y presione Entrar.

Luego, si ejecuta aws s3 ls , ahí está nuestro cubo.

Ahora estamos listos para empaquetar e implementar nuestro código.

Y estos son los comandos que necesitamos.

Y este primer comando solo va a empaquetar nuestro archivo,

entonces va a empaquetar nuestro template.yml .

La salida se llamará sam-template.yml ,

y va a almacenar el paquete en nuestro balde,

pero tendremos que reemplazar este nombre de cubo

con el nombre del depósito que acabamos de crear.

Entonces, de vuelta en su CloudShell , pegue ese comando,

y luego simplemente actualice el nombre de su cubo

para que coincida con el que acabas de crear,

y presione Entrar.

Y si todo funcionó, entonces esto es lo que deberías ver.

Y luego, finalmente, para implementar el código,

este es el comando que vamos a usar.

Así que copia el comando. Vuelva a su CloudShell .

Y en este mandato,

estamos implementando el paquete que acabamos de crear.

Así que estamos usando sam deployment.

Aquí está nuestro archivo de plantilla;

el Pila de CloudFormation que crea

se llamará mystack .

Esto solo permite que CloudFormation cree recursos de IAM,

y cuando crea la función,

va a crear un rol de ejecución

para nuestra función, también,

por lo que debemos asegurarnos de dar CloudFormation

la capacidad de hacer eso,

y eso es lo que hace esta bandera aquí.

Entonces, si está satisfecho con eso, simplemente presione Entrar,

y puede que solo tarde unos minutos en completarse.

Y si todo ha tenido éxito,

entonces esto es lo que deberías ver.

Y bajo ResourceStatus ,

podemos ver que se ha creado el rol de IAM,

y se crea la función Lambda,

y ha terminado de implementar la pila de CloudFormation .

Así que ahora que todo está completo,

podemos dirigirnos a Lambda

y echa un vistazo a nuestra función.

Entonces, en el cuadro de búsqueda, escriba lambda.

Aquí está nuestra función.

Si nos desplazamos hacia abajo y seleccionamos Prueba,

podemos crear un nuevo evento de prueba.

voy a llamarlo te1,

y guarde, y luego seleccione Prueba.

Así que ha ejecutado nuestra función,

y aquí está la respuesta de nuestra función.

Y luego, si buscamos CloudFormation ,

seleccione nuestra pila,

y luego seleccione Recursos,

aquí está la función que se creó,

y aquí está el rol de ejecución de la función.

Y creamos todo eso de manera muy rápida y sencilla.

simplemente usando un par de comandos de AWS SAM.

Entonces, en resumen, usamos AWS CloudShell

para ejecutar nuestros comandos SAM

porque la CLI de SAM está preinstalada.

Creamos un depósito S3.

Usamos sam package para empaquetar nuestro código,

y lo guardó automáticamente en el depósito S3.

Y luego usamos sam deployment

para implementar una función de Lambda mediante CloudFormation .

Eso es todo por esta lección.

Si tiene alguna pregunta, por favor hágamelo saber.

De lo contrario, te veré en la próxima lección.

Gracias.