[ENGLISH]

Hello, Cloud Gurus,

and welcome to this lesson

on "Creating Dynamic Lambda Deployments

Using API Gateway Stages and Staging Variables."

We'll begin by creating two Lambda functions,

name them "MyProdFunction" and "MyDevFunction."

Next, we'll configure a REST API in API gateway,

and we'll configure an HTTP GET method

that references a stage variable called "lmbfunc."

We'll create prod and dev stages in API gateway,

and for each stage, we'll add the stage variable

that is going to identify our function.

Then we can test that it's all working

by using the Invoke URL for each stage.

And we'll have one URL for prod and one for dev.

So if you're ready to get your hands dirty with API gateway,

I'll see you in the console.

First, search for Lambda.

Create our first function, call it "MyProdFunction."

We'll accept all the defaults and Create Function.

After the function is created,

we're gonna modify the code in the function body,

and Deploy.

After we've done that,

we're gonna do the same again for our dev function.

So come back to Functions, Create Function,

call it "MyDevFunction," and Create Function.

Once again, we'll modify the code

in the function body and Deploy.

Next, search for API Gateway,

and we're gonna create a public REST API.

So choose an API type, scroll down and find REST API.

Make sure you select the public one,

not the one that says "Private."

So, select Build.

If you see this message, just click Okay.

It's gonna be a REST API, and we're creating a New API.

API name is "MyAPI," and then Create API.

Then under Resources, select Actions, Create Method,

from the dropdown, select GET,

because we're creating a new HTTP GET method,

then make sure you select this tick to add the method.

Integration type is gonna be a Lambda Function,

because we'll be calling one of our functions.

And then down here, we need to provide our function name,

and this is how we add the stage variable.

So it's $, stageVariables,

make sure you get the capitalization right,

then dot, and then the name of my stage variable

which is gonna be "lmbfunc."

Once you've done that, hit Save, and it will provide you

with a command to use that's gonna set up the permissions

that API gateway needs to invoke both of your functions.

However, we need to edit the command to make it correct.

So first of all, we need to add our function name

instead of this stage variable.

But then after adding my function name,

I actually found that this command still didn't work,

and I'll show you what I mean.

So to get it to work,

I'm just gonna copy everything in this command,

and then paste the command into my text editor.

First of all, I'm gonna replace this stage variable name

with the name of my function.

But then somehow, in providing this command,

it's messed up the spaces.

So, you need to delete all of these extra spaces.

First of all, there's an extra space in here,

which it's not going to like.

Then, anywhere you see this double hypen,

it seems to have added an additional space

that we don't want.

So there's one here and there's one here as well,

and there's one over here as well.

So once you've done that,

once you've added your function name,

got rid of any extra spaces that appear

near these double hyphens,

let's copy the command and see if we can run it.

Select the CloudShell and paste in your command,

and hit Enter.

And there we go, if it's all worked correctly,

this is what you should see.

Now if you are struggling to get this command to work,

check the resources for this lesson,

because I've provided a working version of the command

that you can use instead.

All you need to do is update it with the ARN

of your own function and the ARN of your own API gateway.

Once you've got this command working for prod,

you will need to do it for your dev function as well.

So I'm just gonna copy this command,

add in the name of "MyDevFunction,"

and then copy the remaining and enter.

And there we go, we've added the permission for API gateway

to invoke both "MyProdFunction" and "MyDevFunction."

Now before we can go any further, we need to deploy our API.

So I'm gonna close down the CloudShell.

Under Resources, select Actions,

Deploy API.

Under deployment stage, select New Stage,

we need to give it a name.

I'm just gonna call it "test," and Deploy.

Now we can create our prod stage and our dev stage.

So in Stages, select Create, stage name will be "prod,"

select the Deployment and Create.

Then select Stage Variables, Add Stage Variable,

put in the name of our stage variable

that we need to define, and the value is MyProdFunction.

So that's the name of our Lambda function,

and then on the right hand side,

you need to select this tick to save.

After you've done that,

you can test using the Invoke URL that appears up here.

So this is our URL for the prod stage.

So select that, and there we go.

This is what you should see.

It's calling our prod function.

Let's do the same again for development.

So select Create Stage, stage name is "dev."

Select our Deployment, select Create, under Stage Variables,

we're gonna add another stage variable.

This time it's gonna point to MyDevFunction.

Remember to save using the tick on the right,

and then we can test using the Invoke URL,

and this is the invoke URL for the dev stage.

And there we go, it didn't work the first time,

just took a little while to deploy.

So it's saying, "This is my dev function."

So we're using the stage variable that we created

to determine which endpoint API gateway serves.

So for the exam,

just be aware that "We can use API gateway stages

and stage variables to create dynamic Lambda deployments.

We can use stages and staging variables

to reference different backends,

for instance, different Lambda functions."

And that means that,

"We can deploy and test new functionality safely

before deploying to production."

Because we can deploy it to a different stage

and use that stage to test.

So that is it for this lesson.

If you have any questions, please let me know.

Otherwise, I will see you in the next one.

Thank you.

[SPANISH]

Hola, gurús de la nube,

y bienvenidos a esta lección

en "Creación de implementaciones dinámicas de Lambda

Uso de etapas de API Gateway y variables de preparación".

Comenzaremos creando dos funciones de Lambda,

asígneles el nombre "MyProdFunction" y "MyDevFunction".

A continuación, configuraremos una API REST en la puerta de enlace API,

y configuraremos un método HTTP GET

que hace referencia a una variable de etapa llamada "lmbfunc".

Crearemos etapas de producción y desarrollo en API Gateway,

y para cada etapa, agregaremos la variable etapa

que va a identificar nuestra función.

Entonces podemos probar que todo funciona.

utilizando la URL de invocación para cada etapa.

Y tendremos una URL para prod y otra para dev.

Entonces, si está listo para ensuciarse las manos con API Gateway,

Te veré en la consola.

Primero, busque Lambda.

Cree nuestra primera función, llámela "MyProdFunction".

Aceptaremos todos los valores predeterminados y Create Function.

Una vez creada la función,

vamos a modificar el código en el cuerpo de la función,

y desplegar.

Después de que hayamos hecho eso,

vamos a hacer lo mismo nuevamente para nuestra función de desarrollo.

Así que vuelve a Funciones, Crear Función,

llámelo "MyDevFunction" y Create Function.

Una vez más, modificaremos el código.

en el cuerpo de la función y Desplegar.

A continuación, busque API Gateway,

y vamos a crear una API REST pública.

Así que elija un tipo de API, desplácese hacia abajo y busque REST API.

Asegúrate de seleccionar el público,

no el que dice "Privado".

Por lo tanto, seleccione Construir.

Si ve este mensaje, simplemente haga clic en Aceptar.

Será una API REST y estamos creando una nueva API.

El nombre de la API es "MyAPI" y luego Crear API.

Luego, en Recursos, seleccione Acciones, Crear método,

en el menú desplegable, seleccione OBTENER,

porque estamos creando un nuevo método HTTP GET,

luego asegúrese de seleccionar esta marca para agregar el método.

El tipo de integración será una función Lambda,

porque estaremos llamando a una de nuestras funciones.

Y luego aquí abajo, necesitamos proporcionar el nombre de nuestra función,

y así es como agregamos la variable etapa.

Entonces es $, StageVariables,

asegúrese de obtener las mayúsculas correctas,

luego punto, y luego el nombre de mi variable de etapa

que va a ser "lmbfunc".

Una vez que haya hecho eso, presione Guardar y le proporcionará

con un comando para usar que configurará los permisos

esa puerta de enlace API necesita invocar ambas funciones.

Sin embargo, necesitamos editar el comando para que sea correcto.

Primero que nada, necesitamos agregar el nombre de nuestra función

en lugar de esta variable de etapa.

Pero luego, después de agregar el nombre de mi función,

De hecho, descubrí que este comando aún no funcionaba,

y te mostraré lo que quiero decir.

Así que para que funcione,

Voy a copiar todo en este comando,

y luego pegue el comando en mi editor de texto.

En primer lugar, voy a reemplazar este nombre de variable de etapa

con el nombre de mi función.

Pero luego, de alguna manera, al proporcionar este comando,

está desordenado los espacios.

Por lo tanto, debe eliminar todos estos espacios adicionales.

En primer lugar, hay un espacio extra aquí,

que no le va a gustar.

Luego, en cualquier lugar donde veas este doble guión,

parece haber añadido un espacio adicional

que no queremos.

Así que hay uno aquí y también hay uno aquí,

y hay uno aquí también.

Así que una vez que hayas hecho eso,

una vez que haya agregado el nombre de su función,

se deshizo de los espacios adicionales que aparecen

cerca de estos guiones dobles,

copiemos el comando y veamos si podemos ejecutarlo.

Seleccione CloudShell y pegue su comando,

y presione Entrar.

Y ahí vamos, si todo ha funcionado correctamente,

esto es lo que debes ver.

Ahora, si tiene dificultades para que este comando funcione,

revisa los recursos para esta lección,

porque he proporcionado una versión de trabajo del comando

que puedes usar en su lugar.

Todo lo que necesita hacer es actualizarlo con el ARN

de su propia función y el ARN de su propia puerta de enlace API.

Una vez que tenga este comando funcionando para prod,

también deberá hacerlo para su función de desarrollo.

Así que solo voy a copiar este comando,

agregue el nombre de "MyDevFunction",

y luego copie el resto y entre.

Y ahí vamos, hemos agregado el permiso para la puerta de enlace API

para invocar tanto "MyProdFunction" como "MyDevFunction".

Ahora, antes de que podamos continuar, debemos implementar nuestra API.

Así que voy a cerrar CloudShell.

En Recursos, seleccione Acciones,

Implementar API.

En la etapa de implementación, seleccione Nueva etapa,

tenemos que darle un nombre.

Solo lo llamaré "prueba" e implementación.

Ahora podemos crear nuestra etapa de producción y nuestra etapa de desarrollo.

Entonces, en Etapas, seleccione Crear, el nombre de la etapa será "prod"

seleccione Implementación y Crear.

Luego seleccione Variables de etapa, Agregar variable de etapa,

poner el nombre de nuestra variable de etapa

que necesitamos definir, y el valor es MyProdFunction.

Así que ese es el nombre de nuestra función Lambda,

y luego en el lado derecho,

debe seleccionar esta marca para guardar.

Después de que hayas hecho eso,

puede probar usando la URL de invocación que aparece aquí.

Así que esta es nuestra URL para la etapa de producción.

Así que seleccione eso, y ahí vamos.

Esto es lo que deberías ver.

Está llamando a nuestra función prod.

Hagamos lo mismo otra vez para el desarrollo.

Así que seleccione Create Stage, el nombre de la etapa es "dev".

Seleccione nuestra implementación, seleccione Crear, en Variables de etapa,

vamos a agregar otra variable de etapa.

Esta vez apuntará a MyDevFunction.

Recuerda guardar usando el tick de la derecha,

y luego podemos probar usando la URL de invocación,

y esta es la URL de invocación para la etapa de desarrollo.

Y ahí vamos, no funcionó la primera vez,

solo tomó un poco de tiempo para implementar.

Entonces dice: "Esta es mi función de desarrollo".

Así que estamos usando la variable de etapa que creamos

para determinar qué puerta de enlace API de punto final sirve.

Así que para el examen,

solo tenga en cuenta que "Podemos usar etapas de puerta de enlace API

y variables de etapa para crear implementaciones dinámicas de Lambda.

Podemos usar etapas y variables de puesta en escena

para hacer referencia a diferentes backends,

por ejemplo, diferentes funciones de Lambda".

Y eso significa que,

"Podemos implementar y probar nuevas funciones de forma segura

antes de implementarlo en producción".

Porque podemos implementarlo en una etapa diferente

y use esa etapa para probar.

Eso es todo por esta lección.

Si tiene alguna pregunta, por favor hágamelo saber.

De lo contrario, te veré en la próxima.

Gracias.