[ENGLISH]

Hello, Cloud Gurus, and welcome to this lesson

where we'll be reviewing

some example serverless architectures,

beginning with a quick recap

on the characteristics of event-driven architecture.

We'll then take a look at an example banking application,

an image processing application,

plus an application that processes streaming data.

And as usual, we'll go over my exam tips as well.

So without further ado, let's get started.

Now, when it comes to event-driven architectures,

they are asynchronous, and that means

that events and asynchronous communication

is used to loosely couple application components.

And what that really means

is that an event or message might trigger an action

but no response is necessarily expected or required.

So this means that the components

of an application are not dependent on a response

before they can continue to do the next step.

Services and components are loosely coupled,

which means they operate

and scale independently of each other.

And these kind

of architectures usually include single-purpose functions.

So stateless functions performing a short-lived task.

Now, hopefully, all of that sounds pretty familiar to you.

And when we think about event-driven architecture,

hopefully you're already thinking

about serverless architectures and Lambda.

So when it comes to building applications,

we can use AWS services

as building blocks for our applications,

just like LEGO bricks.

Many AWS services provide inherent scalability,

performance, and reliability.

And this means they are also flexible

and require minimal maintenance by you.

And of course,

it is always good practice to avoid writing extra code

to do something that AWS already offers as a service.

So what kind of components typically make up

an event-driven architecture?

Well, usually, there's an event source.

For example, an event can be generated

if an object is uploaded to S3

or if an item in a DynamoDB table is modified.

There's an event destination, for example, Lambda.

And there may be an event router

in between like EvenBbridge, coordinating everything

if the system is complex, for instance, if it's doing more

than just triggering a single Lambda function.

So now let's consider our example

of a simple banking application.

Imagine a customer making a withdrawal from an ATM machine

and the customer request hits an API Gateway.

This triggers a Lambda function

to process the cash withdrawal

and then write an update to a DynamoDB table.

The table update then triggers an event

because the customer has gone overdrawn.

The event is routed by EventBridge, first of all,

to trigger an overdraft notification function,

which is gonna send the message

to an SQS queue to begin the workflow

to offer the customer an increased overdraft.

And EventBridge is also going to send an SNS notification

to the customer's phone,

letting them know that they are using their overdraft.

So that is one example

of how we can integrate a number of different AWS services

to create an event-driven application.

Let's take a look at another example.

And this time, it's an image processing application.

Now, the workflow starts when a customer uploads an image

to an S3 bucket.

When an object is uploaded,

this creates a trigger that invokes a Lambda function

whenever a new image is added to the S3 bucket.

The Lambda function performs some processing

on the uploaded image

and it's gonna resize it as a thumbnail.

After processing the image,

the function then stores the processed image

in another S3 bucket

and then it's gonna store image metadata

in a DynamoDB table.

And then all of the processed images

are served by CloudFront.

And then in my final example,

we have an application that processes streaming data.

And this application is processing data

from a social media feed.

The social media data is loaded

into Kinesis Firehose in real time.

Then a Lambda function consumes the data

from Kinesis and performs some processing on it.

For instance, it might perform sentiment analysis

on the social media posts.

It then saves the results in a DynamoDB table

so that another team can query the data.

And that is my final example.

Now, in the exam, you may be asked about the best way

to design an event-driven application.

So it's worth spending a little time to understand

which services can be brought together

to create event-driven applications.

So for the exam, just remember that serverless architectures

are event-driven and asynchronous.

So an event or message might trigger an action

but no response is necessarily expected or required.

Think of AWS services as building blocks

that can be integrated together to build an application,

like building a model using LEGO.

Event-driven architectures are loosely coupled.

Services like SQS allow you

to queue messages and EventBridge helps you handle events

and route them to application components.

And finally, all of this gives you flexibility

and scalability as well.

And this is because services and components can operate

and scale independently of each other.

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 , Cloud Gurus , y bienvenidos a este lección

dónde estaremos revisando \_

alguno ejemplo sin servidor arquitecturas ,

comienzo con un rápido resumen

en el características de evento conducido arquitectura \_

Bien entonces echa un vistazo a un ejemplo bancario aplicación ,

un imagen Procesando aplicación ,

más un solicitud eso procesos transmisión de datos.

Y como de costumbre, vamos a ir encima mi examen consejos también . \_

Así que sin más ado , vamos conseguir empezó \_

Ahora , cuando se trata de evento conducido arquitecturas ,

son asincrónicos , y eso medio

eso eventos y asíncronos comunicación

es usado a flojamente pareja solicitud componentes \_

y que eso en realidad medio

es eso un evento o mensaje podría desencadenar un acción

pero no hay respuesta necesariamente esperado o requerido \_

entonces esto medio eso el componentes

de un aplicación no son dependiente en una respuesta

antes pueden continuar \_ para hacer el siguiente paso

Los servicios y componentes están vagamente acoplado ,

cual medio ellos funcionar

y escala independientemente de cada otro \_

y estos amable

de arquitecturas generalmente incluyen un solo propósito funciones \_

Tan apátrida funciones realizando un breve tarea \_

Ahora , con suerte , todos de eso sonidos bastante familiar para tu \_

y cuando nosotros pensar acerca de evento conducido arquitectura ,

con un poco de suerte estás ya pensamiento

acerca de sin servidor Arquitecturas y Lambda.

entonces cuando se trata de edificio aplicaciones ,

podemos usar los servicios de AWS

como bloques de construcción para nuestro aplicaciones ,

justo como ladrillos LEGO .

Muchos servicios de AWS proporcionar inherente escalabilidad ,

rendimiento y fiabilidad .

y esto medio también son flexibles \_

y requieren mínimo mantenimiento por tu \_

y de curso ,

él es siempre bien práctica a evitar escribir código adicional

hacer algo \_ que AWS ya ofrece como servicio .

Entonces que amable de componentes típicamente maquillar \_

un evento conducido arquitectura ?

Bueno , por lo general , hay un evento fuente \_

Para ejemplo , un Se puede generar un evento

si un objeto es subido a S3

o si un elemento en una tabla de DynamoDB es modificado \_

hay un evento destino , por ejemplo , lambda.

y allí puede ser un evento enrutador

en el medio como EvenBbridge , coordinador todo

si el sistema es complejo , por instancia , si es haciendo más

que justo activando una única función Lambda .

entonces ahora vamos considerar nuestro ejemplo

de una simple banca aplicación \_

Imagina un cliente hacer un retiro de un cajero automático

y el cliente la solicitud llega a una API Gateway.

Este activa una función Lambda

a proceso el retiro de efectivo

y luego escribir un actualizar a una tabla de DynamoDB .

La actualización de la tabla entonces disparadores un evento

porque el el cliente se ha ido descubierto \_

El evento es enrutado por EventBridge , primero de todo ,

a desencadenar un sobregiro notificación función ,

cual es ir a enviar el mensaje

a una cola SQS a comenzar el flujo de trabajo

a oferta el cliente un aumentó sobregiro \_

Y EventBridge es también yendo a enviar una notificación SNS

a el clientes teléfono ,

dejar a ellos saber eso ellos están usando su sobregiro \_

Entonces eso es uno ejemplo

de cómo podemos integrar un numero \_ de diferentes servicios de AWS

a crear un evento conducido aplicación \_

Vamos echa un vistazo a otro ejemplo \_

Y esta vez, es un imagen Procesando aplicación \_

Ahora , el flujo de trabajo empieza cuando un cliente cargas un imagen

a un cubo S3 .

Cuando un objeto es subido ,

este crea un disparador eso invoca una función Lambda

cada vez que una nueva imagen es agregado a el cubo S3 .

La función lambda realiza alguno Procesando

en el subido imagen

y es ir a cambiar el tamaño como una miniatura .

Después del procesamiento el imagen ,

el función entonces historias el procesada imagen

en otro cubo S3

y luego es voy a almacenar la imagen metadatos

en una tabla de DynamoDB .

y luego todo de el procesada imágenes

se sirven por frente a la nube .

Y luego en mi ejemplo final ,

nosotros tener un solicitud eso procesos transmisión de datos.

y esto solicitud es procesamiento de datos

de un feed de redes sociales .

Los datos de las redes sociales son cargado

en Kinesis Firehose en tiempo real.

Luego, una función Lambda consume los datos.

de Kinesis y realiza alguno Procesando en eso \_

Para instancia , lo podría llevar a cabo sentimiento análisis

en las publicaciones en las redes sociales.

Él entonces ahorra el da como resultado una tabla de DynamoDB

para que otro el equipo puede consultar los datos

y eso es mi último ejemplo .

Ahora , en el examen , tu se puede preguntar acerca de el mejor forma

a diseño un evento conducido aplicación \_

entonces es valer dedicando un poco de tiempo a entender

cual se pueden traer servicios juntos

a crear evento conducido aplicaciones \_

Entonces para el examen , solo recordar eso sin servidor arquitecturas

son impulsados por eventos y asincrónicos .

entonces un evento o mensaje podría desencadenar un acción

pero no hay respuesta necesariamente esperado o requerido \_

Pensar de los servicios de AWS como bloques de construcción

que se puede integrar juntos a construir un aplicación ,

como construyendo un modelo utilizando LEGO.

Evento conducido las arquitecturas son sueltas acoplado \_

Servicios como SQS permitir tú

a cola mensajes y EventBridge ayuda tú manejar eventos

y ruta a ellos a solicitud componentes \_

Y finalmente , todo de este da tú flexibilidad

y escalabilidad también . \_

y esto es porque los servicios y componentes pueden operar

y escala independientemente de cada otro \_

Entonces eso es él para este lección \_

Si tú tener cualquier preguntas , por favor házmelo saber \_ \_

De lo contrario , lo haré ver tu en el próximo lección \_

Agradecer tu \_