[ENGLISH]

Hello, Cloud Gurus,

and welcome to this lecture, which is going to focus

on comparing the different kinds of workflow

that are available with Step Functions.

And we'll begin with Step Functions Workflows.

We'll take a look at Standard Workflows

and Express Workflows.

We'll explore the differences

between Synchronous and Asynchronous Express Workflows

and my exam tips.

Now, Step Functions provides various types of state machine.

And if you remember, a state machine

is simply a workflow in Step Functions,

and they provide various types of state machine

that featured different workflows

to cater for a variety of tasks

that you would like to orchestrate.

And the kind of task that you want to orchestrate

is going to determine the type of workflow

that you should select.

Now, first of all, we have Standard Workflows.

So what are these good for?

Well, these are great

for long-running, durable, and auditable workflows

that may run for up to 1 year.

And with these, full execution history

is available for up to 90 days

after the workflow has completed.

Standard Workflows use an at-most-once model.

And this means that tasks are never executed more than once

unless you explicitly specify retry actions in your code

or in your workflow.

Standard Workflows are designed for non-idempotent actions;

for example, when processing payments,

and you only want a payment to be processed once,

not multiple times.

But what do we mean by non-idempotent?

Well, a request is non-idempotent

if it always causes a change in state.

So let's say we repeat an identical request,

will it always cause a change in state?

And with our example of a payment,

it is always going to cause a change in state

if we repeat an identical payment.

Or another example is sending the same email multiple times.

It's always going to cause a change in state

because you will end up with multiple emails in your inbox.

Next, we have Express Workflows

and these are designed for short-lived workflows

that last up to 5 minutes.

So they are great

for high volume event-processing-type workloads.

They use an at-least-once model.

So the workflow runs at least once,

but it could run more than once.

So Express Workflows are ideal

if there's a possibility

that an execution might be run more than once

or that you require multiple concurrent executions.

And these are great for idempotent actions;

for example, transforming some input data

and then storing the result in DynamoDB.

And a request is idempotent

if an identical request has no side effect.

So it's considered idempotent if an identical request

can be made once or several times in a row

with no additional side effects.

For example, reading data from a database or an S3 bucket,

or for example,

taking some input data and running an identical process

to transform that data

and then store the result in DynamoDB.

The result will always be the same.

Now there are 2 different types of Express Workflows.

We've got synchronous and asynchronous

and you'll need to understand the differences.

So with a synchronous workflow, first of all,

it begins the workflow.

It waits until the workflow is completed

and then returns the result.

So this is great for operations

that are going to be performed one at a time.

So the workflow must complete before the next step begins.

For example,

you should wait for your customer to complete their payment

before you send them their order.

Whereas with an Asynchronous Express Workflow,

it begins a workflow,

it confirms the workflow has started,

but it is not going to return any result to you.

Instead, the result of the workflow

can be found in CloudWatch Logs.

So it's not waiting for the workflow to complete

before moving on to the next task.

So this is great if your services or operations

don't depend on the completion

and the result of your workflow.

For example, think of a messaging system

where you just want to send a message

and then move on to the next task

without waiting for an acknowledgement.

Now it's important that you get your head

around this idea of synchronous or asynchronous.

So let's think about it a different way.

What about a conversation? For example,

a face-to-face job interview,

where the interviewer asks you a question

and waits for your response

before moving onto the next question.

Would you describe that as synchronous or asynchronous?

Well, I would describe it as synchronous

because you're waiting for a response before moving on.

So what about sending an email?

So you send an email to a coworker

providing a link to a shared folder,

containing files that you'll be collaborating on together.

Then you move on to your next task of the day

without waiting for them to respond.

Would you say that's synchronous or asynchronous?

Well, I would describe that as asynchronous

because you fire off a task

and then move onto the next thing

without waiting for a confirmation or a response.

And as a general rule of thumb,

Asynchronous Workflows often run in the background

while synchronous workflows hold you up

until they all complete.

So for my exam tips,

make sure that you understand the differences

between the different types of Step Functions Workflow.

So we've got Standard Workflows,

which are for long-running workflows

running for up to 1 year.

They run at-most-once and they are non-idempotent.

So that means they always cause a change in state.

And if you were to make an identical request,

it is always going to cause another change in state.

For example, processing a payment multiple times.

Express Workflows are short-lived workflows,

which last up to 5 minutes.

They run at least once and can run multiple times

and they are idempotent.

And that means that you can run a request more than once

and it will not have any additional side effects.

For example, reading from a database.

With Express Workflows, you've got 2 further options,

so you've got synchronous and asynchronous.

And with Synchronous Express Workflows,

the workflow must complete before the next step begins.

For example, you want to confirm a successful payment

before sending out an order.

Whereas with Asynchronous Express Workflows,

other tasks are not dependent

on the completion of the workflow.

For example, with a messaging system

where you send a message and then move on

without waiting for any confirmation or acknowledgement.

So that's it for this lecture.

If you have any questions, please let me know.

Otherwise I'll see you in the next lecture. Thank you.

[SPANISH]

Hola, gurús de la nube,

y bienvenidos a esta conferencia, que se va a centrar

sobre la comparación de los diferentes tipos de flujo de trabajo

que están disponibles con Step Functions.

Y comenzaremos con los flujos de trabajo de Step Functions.

Echaremos un vistazo a los flujos de trabajo estándar

y flujos de trabajo exprés.

Exploraremos las diferencias.

entre flujos de trabajo rápidos síncronos y asíncronos

y mis consejos para el examen.

Ahora, Step Functions proporciona varios tipos de máquinas de estado.

Y si recuerdas, una máquina de estados

es simplemente un flujo de trabajo en Step Functions,

y proporcionan varios tipos de máquina de estado

que presentaba diferentes flujos de trabajo

para atender una variedad de tareas

que le gustaría orquestar.

Y el tipo de tarea que desea orquestar

va a determinar el tipo de flujo de trabajo

que debes seleccionar.

Ahora, en primer lugar, tenemos flujos de trabajo estándar.

Entonces, ¿para qué sirven estos?

Bueno, estos son geniales

para flujos de trabajo de larga ejecución, duraderos y auditables

que puede durar hasta 1 año.

Y con estos, historial completo de ejecuciones.

está disponible por hasta 90 días

después de que se haya completado el flujo de trabajo.

Los flujos de trabajo estándar utilizan un modelo de una vez como máximo.

Y esto significa que las tareas nunca se ejecutan más de una vez.

a menos que especifique explícitamente acciones de reintento en su código

o en su flujo de trabajo.

Los flujos de trabajo estándar están diseñados para acciones no idempotentes;

por ejemplo, al procesar pagos,

y solo desea que un pago se procese una vez,

no varias veces.

Pero, ¿qué entendemos por no idempotente?

Bueno, una solicitud no es idempotente.

si siempre provoca un cambio de estado.

Entonces, digamos que repetimos una solicitud idéntica,

¿Siempre causará un cambio de estado?

Y con nuestro ejemplo de un pago,

siempre va a causar un cambio de estado

si repetimos un pago idéntico.

Otro ejemplo es enviar el mismo correo electrónico varias veces.

Siempre va a causar un cambio de estado.

porque terminará con múltiples correos electrónicos en su bandeja de entrada.

A continuación, tenemos Express Workflows

y estos están diseñados para flujos de trabajo de corta duración

que duran hasta 5 minutos.

Entonces son geniales

para cargas de trabajo de procesamiento de eventos de alto volumen.

Usan un modelo de al menos una vez.

Así que el flujo de trabajo se ejecuta al menos una vez,

pero podría funcionar más de una vez.

Así que los flujos de trabajo Express son ideales

si hay una posibilidad

que una ejecución puede ejecutarse más de una vez

o que requiere múltiples ejecuciones simultáneas.

Y estos son geniales para acciones idempotentes;

por ejemplo, transformar algunos datos de entrada

y luego almacenar el resultado en DynamoDB.

Y una solicitud es idempotente

si una solicitud idéntica no tiene efectos secundarios.

Entonces se considera idempotente si una solicitud idéntica

se puede hacer una o varias veces seguidas

sin efectos secundarios adicionales.

Por ejemplo, leer datos de una base de datos o un depósito S3,

o por ejemplo,

tomando algunos datos de entrada y ejecutando un proceso idéntico

para transformar esos datos

y luego almacene el resultado en DynamoDB.

El resultado siempre será el mismo.

Ahora hay 2 tipos diferentes de Express Workflows.

Tenemos sincrónicos y asincrónicos

y tendrá que entender las diferencias.

Entonces, con un flujo de trabajo síncrono, en primer lugar,

comienza el flujo de trabajo.

Espera hasta que se complete el flujo de trabajo.

y luego devuelve el resultado.

Así que esto es genial para las operaciones.

que se van a realizar de uno en uno.

Por lo tanto, el flujo de trabajo debe completarse antes de que comience el siguiente paso.

Por ejemplo,

debe esperar a que su cliente complete su pago

antes de enviarles su pedido.

Mientras que con un flujo de trabajo rápido asíncrono,

comienza un flujo de trabajo,

confirma que el flujo de trabajo ha comenzado,

pero no te va a devolver ningún resultado.

En cambio, el resultado del flujo de trabajo

se puede encontrar en CloudWatch Logs.

Por lo tanto, no está esperando a que se complete el flujo de trabajo

antes de pasar a la siguiente tarea.

Así que esto es genial si sus servicios u operaciones

no dependas de la terminación

y el resultado de su flujo de trabajo.

Por ejemplo, piense en un sistema de mensajería

donde solo quieres enviar un mensaje

y luego pasar a la siguiente tarea

sin esperar un acuse de recibo.

Ahora es importante que consigas tu cabeza

en torno a esta idea de síncrono o asíncrono.

Así que vamos a pensar en ello de una manera diferente.

¿Qué tal una conversación? Por ejemplo,

una entrevista de trabajo cara a cara,

donde el entrevistador te hace una pregunta

y espero tu respuesta

antes de pasar a la siguiente pregunta.

¿Lo describiría como sincrónico o asincrónico?

Bueno, lo describiría como sincrónico.

porque estás esperando una respuesta antes de continuar.

Entonces, ¿qué pasa con el envío de un correo electrónico?

Así que le envías un correo electrónico a un compañero de trabajo

proporcionando un enlace a una carpeta compartida,

que contienen archivos en los que colaborarán juntos.

Luego pasas a tu próxima tarea del día.

sin esperar a que respondan.

¿Dirías que es sincrónico o asincrónico?

Bueno, lo describiría como asíncrono.

porque disparas una tarea

y luego pasar a lo siguiente

sin esperar una confirmación o una respuesta.

Y como regla general,

Los flujos de trabajo asincrónicos a menudo se ejecutan en segundo plano

mientras los flujos de trabajo sincrónicos lo detienen

hasta que se completen todos.

Así que para mis consejos de examen,

asegúrese de entender las diferencias

entre los diferentes tipos de flujo de trabajo de Step Functions.

Tenemos flujos de trabajo estándar,

que son para flujos de trabajo de ejecución prolongada

funcionando hasta por 1 año.

Se ejecutan como máximo una vez y no son idempotentes.

Eso significa que siempre provocan un cambio de estado.

Y si tuviera que hacer una solicitud idéntica,

siempre va a causar otro cambio de estado.

Por ejemplo, procesar un pago varias veces.

Express Workflows son flujos de trabajo de corta duración,

que duran hasta 5 minutos.

Se ejecutan al menos una vez y pueden ejecutarse varias veces

y son idempotentes.

Y eso significa que puede ejecutar una solicitud más de una vez

y no tendrá efectos secundarios adicionales.

Por ejemplo, leer de una base de datos.

Con Express Workflows, tiene 2 opciones más,

así que tienes sincrónico y asincrónico.

Y con Synchronous Express Workflows,

el flujo de trabajo debe completarse antes de que comience el siguiente paso.

Por ejemplo, desea confirmar un pago exitoso

antes de enviar un pedido.

Mientras que con los flujos de trabajo rápidos asíncronos,

otras tareas no son dependientes

en la finalización del flujo de trabajo.

Por ejemplo, con un sistema de mensajería

donde envías un mensaje y luego sigues adelante

sin esperar ninguna confirmación o acuse de recibo.

Así que eso es todo por esta conferencia.

Si tiene alguna pregunta, por favor hágamelo saber.

De lo contrario, te veré en la próxima conferencia. Gracias.