[ENGLISH]

Hello, Cloud Gurus,

and welcome to this lecture,

where we're going to take a closer look at CodePipeline.

And of course, CodePipeline is a fully managed CI/CD service

which orchestrates the build, test,

and deployment stages of your software release process.

And you can configure CodePipeline to trigger the pipeline

every time there's a change to your code.

For example, if you're using CodeCommit

to store your source code

and you make some updates to your code,

then this can trigger CodePipeline to begin the build, test,

and deployment processes automatically,

just like a conductor in an orchestra,

controlling everything, directing and orchestrating

the entire process so you don't have to.

And by using an automated release process like this,

it means that your release process is going to be faster,

more consistent, and with fewer mistakes,

and it really enables the quick release of new features

and bug fixes in your software.

But how does it do all of this?

Well, it actually integrates

with a whole bunch of other AWS tools

as well as some third-party open source tools.

So think CodeCommit, CodeBuild,

and CodeDeploy for the AWS developer tools.

It also integrates with open source tools

like GitHub and Jenkins.

And if you haven't used Jenkins before,

it's an open source application which can be used

to build, test, and deploy your application.

CodePipeline also integrates with Elastic Beanstalk

and CloudFormation,

as well as Lambda and Elastic Container Service.

And if you haven't used Elastic Container Service before,

it's a fully managed platform which allows you to run

containers like Docker as well as Windows containers in AWS.

So let's take a look at an example CodePipeline workflow,

and you'll see how CodePipeline integrates

with all the other developer tools

that we've been talking about.

So once your workflow has been defined,

it all begins when a change is detected

in your code repository.

And in this case, the code repository is CodeCommit.

So new code appears in our CodeCommit repo,

and that is detected,

and immediately CodeBuild comes in

and compiles the source code,

runs any tests that you've defined,

and produces packages which can be used

to deploy your software.

And then finally, CodeDeploy comes in,

and the newly built application is then deployed

into either a staging or production environment.

So all of this can be done automatically,

and this is an example of a CI/CD,

continuous integration and continuous deployment

or delivery workflow,

all orchestrated using CodePipeline.

So let's take a look at some of our CodePipeline exam tips.

First of all, just remember it's a CI/CD service,

so it orchestrates your end-to-end software release process

based on a workflow that you define.

It can be configured to automatically trigger your pipeline

as soon as a change is detected

in your source code repository.

And finally, it integrates with lots of AWS tools,

as well as some third-party tools.

So think CodeCommit, CodeBuild,

and CodeDeploy for the AWS developer tools,

and for the third-party open source tools,

it also integrates with GitHub and Jenkins,

as well as things like Elastic Beanstalk, CloudFormation,

Lambda, and Elastic Container Service.

So that is the end of this lecture.

If you have any questions, please let me know.

If not, feel free to move on to the next lecture.

Thank you.

[SPANISH]

Hola, gurús de la nube,

y bienvenidos a esta conferencia,

donde vamos a echar un vistazo más de cerca a CodePipeline .

Y, por supuesto, CodePipeline es un servicio de CI/CD completamente administrado.

que orquesta la construcción, prueba,

y las etapas de implementación de su proceso de lanzamiento de software.

Y puede configurar CodePipeline para activar la canalización

cada vez que hay un cambio en su código.

Por ejemplo, si usa CodeCommit

para almacenar su código fuente

y haces algunas actualizaciones a tu código,

entonces esto puede activar CodePipeline para comenzar la compilación, prueba,

y procesos de implementación automáticamente,

como un director de orquesta,

controlándolo todo, dirigiendo y orquestando

el proceso para que usted no tenga que hacerlo.

Y mediante el uso de un proceso de lanzamiento automatizado como este,

significa que su proceso de lanzamiento será más rápido,

más consistente, y con menos errores,

y realmente permite el lanzamiento rápido de nuevas características

y correcciones de errores en su software.

Pero, ¿cómo hace todo esto?

Bueno, en realidad se integra

con un montón de otras herramientas de AWS

así como algunas herramientas de código abierto de terceros.

Así que piense en CodeCommit , CodeBuild ,

y CodeDeploy para las herramientas de desarrollo de AWS.

También se integra con herramientas de código abierto.

como GitHub y Jenkins.

Y si no ha usado Jenkins antes,

es una aplicación de código abierto que se puede utilizar

para construir, probar e implementar su aplicación.

CodePipeline también se integra con Elastic Beanstalk

y formación de nubes ,

así como Lambda y Elastic Container Service.

Y si no ha usado antes Elastic Container Service,

es una plataforma completamente administrada que le permite ejecutar

contenedores como Docker , así como contenedores de Windows en AWS.

Así que echemos un vistazo a un ejemplo de flujo de trabajo de CodePipeline ,

y verá cómo se integra CodePipeline

con todas las demás herramientas de desarrollo

que hemos estado hablando.

Entonces, una vez que haya definido su flujo de trabajo,

todo comienza cuando se detecta un cambio

en su repositorio de código.

Y en este caso, el repositorio de código es CodeCommit .

Aparece un nuevo código en nuestro repositorio de CodeCommit ,

y eso se detecta,

e inmediatamente aparece CodeBuild

y compila el código fuente,

ejecuta cualquier prueba que haya definido,

y produce paquetes que se pueden utilizar

para implementar su software.

Y finalmente, aparece CodeDeploy ,

y luego se implementa la aplicación recién construida

en un entorno de ensayo o de producción.

Así que todo esto se puede hacer automáticamente,

y este es un ejemplo de un CI/CD,

continua y despliegue continuo

o flujo de trabajo de entrega,

todo orquestado usando CodePipeline .

Así que echemos un vistazo a algunos de nuestros consejos para el examen de CodePipeline .

En primer lugar, recuerde que es un servicio de CI/CD,

por lo que organiza su proceso de lanzamiento de software de extremo a extremo

basado en un flujo de trabajo que defina.

Se puede configurar para activar automáticamente su canalización

tan pronto como se detecte un cambio

en su repositorio de código fuente.

Y finalmente, se integra con muchas herramientas de AWS,

así como algunas herramientas de terceros.

Así que piense en CodeCommit , CodeBuild ,

y CodeDeploy para las herramientas de desarrollo de AWS,

y para las herramientas de código abierto de terceros,

se integra con GitHub y Jenkins,

así como cosas como Elastic Beanstalk, CloudFormation ,

Lambda y servicio de contenedor elástico.

Así que ese es el final de esta conferencia.

Si tiene alguna pregunta, por favor hágamelo saber.

Si no, siéntase libre de pasar a la siguiente lección.

Gracias.