[ENGLISH]

Hello, Cloud Gurus

and welcome to this lecture.

Which is a summary of the developer theory section

of the course, including all of my exam tips.

And this lecture is split into two parts

to keep it short and sweet.

So let's start off with the developer tools,

beginning with CodeCommit.

And CodeCommit is a source and version control service

which enables teams to collaborate

on code, HTML pages, scripts, images, and binaries.

We then have CodeBuild, which is an automated build service

and CodeBuild compiles source code, runs tests

and produces packages that are ready to deploy.

We then have CodeDeploy

which is an automated deployment service.

And CodeDeploy automates code deployments

to EC2 instances, Lambda

and on-premises system as well.

And then we have CodePipeline.

And we can use CodePipeline to manage our build

and deployment workflow.

So it's an end-to-end solution allowing you to build, test

and deploy your application.

Or somatically every time there is a change

in your code base.

Onto continuous integration and continuous deployment.

And these are software development, best practices.

So with continuous integration,

this is all about integrating or merging

your code changes frequently at least once per day.

And if you see anything

in the exam relating to continuous integration

then I want you to think of CodeCommit.

Continuous delivery is all about automating

the build, test and deployment functions.

And if you see anything

in the exam relating to continuous delivery

then I want you to think about CodeBuild

for the build and test stages,

and CodeDeploy for the deployment stage.

And then we also have continuous deployment.

And this is all about fully automating the release process.

So code is deployed into staging or production as soon

as it has successfully passed through the release pipeline.

And if you see anything

in the exam relating to continuous deployment

then I want you to think of CodePipeline.

And I would strongly recommend reviewing

this AWS Whitepaper.

It's called practicing continuous integration

and continuous deployment on AWS.

And it does a really good job of explaining the features

and benefits of CI, CD and the AWS tooling

that is available.

So definitely check that one out before you take the exam.

Onto CodeCommit, and CodeCommit

is a centralized at code repository.

So it's a central place to store everything

that your software needs to run.

So the source code, binaries, libraries, images

and HTML files, et cetera.

And it is based on Git.

It enables collaboration for your teams

and it's able to manage updates from multiple users.

And it also provides version control.

So it tracks and manages changes

and maintains a version history

of all of your files.

Onto CodeDeploy.

And you'll need to understand the differences

between an in-place deployment and a blue-green deployment.

So with an in-place deployment, remember we take one or more

of our instances offline while we run the deployment.

So that means that capacity is going to be reduced

during the deployment.

Lambda is not supported.

Rolling back, involves a re-deploy

but it's a great one to use

if you are deploying your application

for the very first time.

Whereas with blue-green deployments

this is where we provision additional servers

and we install the new version

of our code on these new servers.

So there is no capacity reduction

with a blue-green deployment.

The green instances can be created ahead of time.

If you need to roll back it's very easy to switch

between the old and new environment.

However, you will pay for two environments running

in parallel until you terminate the old servers.

And of course the blue-green deployment is the safest option

because there's no capacity reduction.

We're not touching the existing environment,

and we are deploying our new version

of the application to a brand new environment.

And with CodeDeploy, we have the Appspec file

which is a configuration file,

which defines the parameters to be used by CodeDeploy.

For example, the operating system we're deploying to,

any files which need to be copied over

during the deployment and lifecycle event hooks.

Which are the scripts that you want to run

during your deployment.

Your deployment folder should look like this.

So we have the appspec file saved into the root

of the revision, and then everything else will be organized

into the relevant folders.

But the most important thing to note

is that the appspec file must be in the root of the folder.

Because if you store it in one of these folders

like the config folder, for example

then CodeDeploy will not be able to find it

and your deployment will fail.

And then we have the hooks section

which is another really important section

of the appspec file.

And these are our lifecycle event hooks.

So any scripts that we want to run

during the life cycle of the deployment, and remember

that they have a very specific run order.

And here is my example of an appspec file.

So this is where we reference the operating system,

this where we specify the files that we want to copy over

during the deployment.

So we've got the source

and the destination that we want to copy them to.

And here is the hooks section.

So these are all the scripts that we want to run

at each stage of the deployment.

And once again the lifecycle event hooks are run

in a very specific order and it's known as the run order.

And the most important one that you need to remember

is the in-place deployment.

And with an in-place deployment

there are broadly three phases.

Firstly de-registering our instances with a load balancer

then the installation itself.

And then once the application is healthy re-registering

our instances with the load balancer.

And for the exam you will need to understand

the logical flow and it's worth maybe one or two marks

in the exam.

So that's it for this lecture.

If you have any questions, please let me know.

And if you are ready to move on to part two

I will see you in the next lecture.

Thank you.

[SPANISH]

Hola, gurús de la nube

y bienvenidos a esta conferencia.

Que es un resumen de la sección de teoría del desarrollador

del curso, incluidos todos mis consejos para el examen.

Y esta conferencia se divide en dos partes.

para que sea breve y dulce.

Entonces, comencemos con las herramientas para desarrolladores,

comenzando con CodeCommit .

Y CodeCommit es un servicio de control de código fuente y versión.

que permite a los equipos colaborar

en código, páginas HTML, scripts, imágenes y archivos binarios.

Luego tenemos CodeBuild , que es un servicio de compilación automatizado.

y CodeBuild compila el código fuente, ejecuta pruebas

y produce paquetes que están listos para implementar.

Entonces tenemos CodeDeploy

que es un servicio de implementación automatizado.

Y CodeDeploy automatiza las implementaciones de código

a instancias EC2, Lambda

y el sistema local también.

Y luego tenemos CodePipeline .

Y podemos usar CodePipeline para administrar nuestra compilación

y flujo de trabajo de implementación.

Es una solución integral que le permite crear, probar

e implementar su aplicación.

O somáticamente cada vez que hay un cambio

en su código base.

Hacia la integración continua y el despliegue continuo.

Y estas son las mejores prácticas de desarrollo de software.

Así que con la integración continua,

se trata de integrar o fusionar

su código cambia con frecuencia al menos una vez al día.

Y si ves algo

en el examen relativo a la integración continua

entonces quiero que pienses en CodeCommit .

La entrega continua tiene que ver con la automatización

las funciones de compilación, prueba y despliegue.

Y si ves algo

en el examen relativo a la entrega continua

entonces quiero que pienses en CodeBuild

para las etapas de construcción y prueba,

y CodeDeploy para la etapa de implementación.

Y luego también tenemos el despliegue continuo.

Y esto se trata de automatizar completamente el proceso de lanzamiento.

Por lo tanto, el código se implementa en la puesta en escena o la producción tan pronto

ya que ha pasado con éxito a través de la tubería de lanzamiento.

Y si ves algo

en el examen relativo al despliegue continuo

entonces quiero que piense en CodePipeline .

Y recomendaría encarecidamente revisar

este documento técnico de AWS.

Se llama practicar la integración continua.

e implementación continua en AWS.

Y hace un muy buen trabajo al explicar las características.

y beneficios de CI, CD y las herramientas de AWS

que está disponible.

Así que definitivamente échale un vistazo antes de tomar el examen.

En CodeCommit y CodeCommit

es un repositorio de código centralizado.

Así que es un lugar central para almacenar todo.

que su software necesita para funcionar.

Así que el código fuente, binarios, bibliotecas, imágenes

y archivos HTML, etcétera.

Y está basado en Git .

Permite la colaboración para sus equipos.

y es capaz de administrar actualizaciones de múltiples usuarios.

Y también proporciona control de versiones.

Por lo tanto, rastrea y administra los cambios.

y mantiene un historial de versiones

de todos sus archivos.

En CodeDeploy .

Y tendrás que entender las diferencias.

entre una implementación en el lugar y una implementación azul-verde.

Entonces, con una implementación en el lugar, recuerde que tomamos uno o más

de nuestras instancias sin conexión mientras ejecutamos el deployment.

Entonces eso significa que la capacidad se va a reducir.

durante el despliegue.

Lambda no es compatible.

Retroceder, implica volver a implementar

pero es genial para usar

si está implementando su aplicación

por primera vez

Mientras que con las implementaciones azul-verde

aquí es donde aprovisionamos servidores adicionales

e instalamos la nueva versión

de nuestro código en estos nuevos servidores.

Así que no hay reducción de capacidad.

con un despliegue azul-verde.

Las instancias verdes se pueden crear con anticipación.

Si necesita retroceder, es muy fácil cambiar

entre el viejo y el nuevo entorno.

Sin embargo, pagará por dos entornos que se ejecutan

en paralelo hasta que finalice los servidores antiguos.

Y, por supuesto, el despliegue azul-verde es la opción más segura.

porque no hay reducción de capacidad.

No estamos tocando el entorno existente,

y estamos implementando nuestra nueva versión

de la aplicación a un nuevo entorno.

Y con CodeDeploy , tenemos el archivo Appspec

que es un archivo de configuración,

que define los parámetros que usará CodeDeploy .

Por ejemplo, el sistema operativo que estamos implementando,

cualquier archivo que necesite ser copiado

durante la implementación y los enlaces de eventos del ciclo de vida.

Cuáles son los scripts que desea ejecutar

durante su despliegue.

Su carpeta de implementación debería verse así.

Entonces tenemos el archivo appspec guardado en la raíz

de la revisión, y luego todo lo demás se organizará

en las carpetas correspondientes.

Pero lo más importante a tener en cuenta

es que el archivo appspec debe estar en la raíz de la carpeta.

Porque si lo guardas en una de estas carpetas

como la carpeta de configuración , por ejemplo

entonces CodeDeploy no podrá encontrarlo

y su implementación fallará.

Y luego tenemos la sección de ganchos

que es otra sección muy importante

del archivo appspec .

Y estos son nuestros ganchos de eventos de ciclo de vida.

Cualquier script que queramos ejecutar

durante el ciclo de vida de la implementación y recuerde

que tienen un orden de ejecución muy específico.

Y aquí está mi ejemplo de un archivo appspec .

Así que aquí es donde hacemos referencia al sistema operativo,

aquí donde especificamos los archivos que queremos copiar

durante el despliegue.

Así que tenemos la fuente

y el destino al que queremos copiarlos.

Y aquí está la sección de ganchos.

Estos son todos los scripts que queremos ejecutar.

en cada etapa del despliegue.

Y una vez más se ejecutan los ganchos de eventos del ciclo de vida

en un orden muy específico y se conoce como el orden de ejecución.

Y el más importante que debes recordar

es la implementación en el lugar.

Y con una implementación en el lugar

hay tres fases.

En primer lugar, anular el registro de nuestras instancias con un balanceador de carga

luego la instalación en sí.

Y luego, una vez que la aplicación esté en buen estado, vuelva a registrarse

nuestras instancias con el balanceador de carga.

Y para el examen necesitarás entender

el flujo lógico y vale quizás una o dos marcas

en el examen

Así que eso es todo por esta conferencia.

Si tiene alguna pregunta, por favor hágamelo saber.

Y si estás listo para pasar a la segunda parte

Te veré en la próxima conferencia.

Gracias.