[ENGLISH]

Hello Cloud Gurus, and welcome to this lecture

where we're going to take a look at CI and CD,

and all of the services that AWS provides

to enable developers to adopt CI/CD practices.

So what is CI/CD?

Well, it's a software development best practice,

and it stands for Continuous Integration

and Continuous Delivery or Deployment,

and sometimes it's delivery, sometimes it's deployment,

these terms are frequently used interchangeably,

but there is a difference,

which we're going to find out about later.

And one of the main characteristics of CI/CD

is that it requires you to make

small changes and automate everything.

So we make small incremental code changes,

and we automate as much as possible.

For example, code integration, build, test,

and deployment tasks, and by doing that,

it's possible to make very frequent

code releases on a regular basis,

adding new features for our customers to enjoy,

and fixing bugs as well.

But why is it so cool?

Why does everyone love CI/CD so much?

Well, modern companies like AWS, Netflix,

Google, and Facebook have really pioneered this approach

to releasing code, successfully applying thousands of

changes per day without impacting their production systems.

And by doing that, they're able to quickly bring to market

brand new features and services,

enabling these companies to become

very successful market leaders.

So let's take a look at some of the main benefits

to using the CI/CD approach, and of course automation is

always a good thing when it comes to software development.

It makes things fast, repeatable, and scalable,

and it really enables rapid deployment of new software.

And of course, doing things manually is always a bad thing

when it comes to software development.

And while doing things manually or by hand

might be great if we were

creating a beautiful artisan loaf of bread,

but it's really not ideal for software development

because it's just slow, it's error-prone, it's inconsistent,

it's unscalable, and it creates complexity as well.

And ideally, we want our software to be built and deployed

in a consistent and repeatable way,

making it easier to support and maintain.

And finally, with CI/CD, we're making small,

incremental changes, and testing our changes

before integrating with the main code base.

And this allows you to catch bugs when they're small

and really simple to fix.

So let's take a look at an example

continuous integration workflow at a high level,

and it all starts of with a shared code repository.

So we have multiple developers contributing to a

shared code repository like Git, frequently merging

or integrating their code updates.

And typically, that would be on a daily basis as a minimum.

We then have an automated build stage,

so when changes appear in the code repository,

this triggers an automated build of the new code,

and finally we also have an automated test step as well.

So we run automated tests to check the code locally

before it gets committed into the master code repository.

So moving on to our continuous delivery

and continuous deployment workflow.

After successful testing,

the code gets merged into the master repository.

It then gets prepared for deployment,

so it gets built, tested, and packaged up,

ready for deployment.

And then with continuous delivery,

there's usually a manual decision.

So humans may be involved in the decision

where and when to deploy the code,

and that is known as continuous delivery.

However, that stage can also be fully automated as well.

So the system can automatically deploy the new code

as soon as it's been prepared for deployment,

and that is known as continuous deployment.

So let's take a look at some of the developer tools

that AWS provides to enable developers

to get started with their CI/CD workflow.

Well first of all, we've got CodeCommit,

and that provides source and version control for our code.

So it's a managed source control service,

enabling teams to collaborate.

And you can store anything from code to HTML pages,

scripts, images, and binaries, et cetera.

Basically, everything it takes to make your software run.

And you can store all of that in CodeCommit,

and have it perform source control

and version control for you.

And it's actually a fully managed private Git repository,

so if you've ever used Git before,

then you'll understand exactly what CodeCommit is.

We then have CodeBuild, which is an automated build service,

and that compiles the source code,

it can run automated tests, and produces packages

which are ready to deploy in your environment.

We then have CodeDeploy,

and that provides automated deployment,

so it automates code deployments,

and that works with any instance or server type,

including EC2, Lambda,

and it can also work with your on-premises systems,

so systems located in your own data center.

And finally, we have CodePipeline,

which can manage the entire end to end workflow,

orchestrating the whole build, test,

and deployment pipeline.

It integrates with all of the other tools,

so CodeCommit, CodeBuild, and CodeDeploy,

and you can actually configure it to deploy

the new version of your application every time

there's a code change, and new code appears

in your CodeCommit repository.

And later on in the course, we're going to get

our hands dirty with all of these tools,

so that you can get plenty of hands-on experience

and be really well prepared

for any questions you see about these in the exam.

So let's take a look at our exam tips for CI and CD.

So first of all, continuous integration

is all about integrating or merging code changes frequently,

at least once per day,

and the service they provide for this is CodeCommit.

Just remember,

that is just like a managed private Git repository.

Secondly, we have continuous delivery,

and that's all about automating the build, test,

and deployment functions of your release process.

So think CodeBuild an CodeDeploy.

And finally, we've got continuous deployment,

and that's the fully automated release process

where code is deployed into staging or production

as soon as it's successfully

passed through the release pipeline.

And the service they provide

for that is called CodePipeline.

And then the last thing I wanted to tell you about

is this AWS Whitepaper, and it's called

Practicing Continuous Integration

and Continuous Deployment on AWS.

And this is a great Whitepaper because it explains

the features and benefits of using continuous integration

and continuous delivery or deployment,

and discusses all the AWS tooling that you can use

in your software development environment.

And there's a link to this Whitepaper

in the resources section of the course,

so if you have time,

it's definitely worth reading this Whitepaper.

And that brings us to 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 Cloud Gurus, y bienvenidos a esta conferencia.

donde vamos a echar un vistazo a CI y CD,

y todos los servicios que proporciona AWS

para permitir que los desarrolladores adopten prácticas de CI/CD.

Entonces, ¿qué es CI/CD?

Bueno, es una mejor práctica de desarrollo de software,

y significa Integración Continua

y entrega continua o implementación,

y a veces es entrega, a veces es implementación,

estos términos se usan frecuentemente indistintamente,

pero hay una diferencia,

del que nos enteraremos más adelante.

Y una de las principales características de CI/CD

es que requiere que hagas

pequeños cambios y automatizar todo.

Así que hacemos pequeños cambios de código incrementales,

y automatizamos tanto como sea posible.

Por ejemplo, integración de código, compilación, prueba,

y tareas de implementación, y al hacerlo,

es posible hacer muy frecuente

de código de forma regular,

agregando nuevas características para que nuestros clientes disfruten,

y corregir errores también.

Pero, ¿por qué es tan genial?

¿Por qué todo el mundo ama tanto a CI/CD?

Bueno, las empresas modernas como AWS, Netflix,

Google y Facebook realmente han sido pioneros en este enfoque.

a la liberación de código, aplicando con éxito miles de

cambios por día sin afectar sus sistemas de producción.

Y al hacerlo, pueden llevar rápidamente al mercado

nuevas características y servicios,

permitir que estas empresas se conviertan

líderes de mercado muy exitosos.

Así que echemos un vistazo a algunos de los principales beneficios.

al uso del enfoque CI/CD y, por supuesto, la automatización es

siempre es algo bueno cuando se trata de desarrollo de software.

Hace que las cosas sean rápidas, repetibles y escalables,

y realmente permite la implementación rápida de nuevo software.

Y por supuesto, hacer las cosas manualmente siempre es algo malo.

cuando se trata de desarrollo de software.

Y mientras hace las cosas manualmente o a mano

podría ser genial si fuéramos

creando una hermosa hogaza de pan artesanal,

pero realmente no es ideal para el desarrollo de software

porque es lento, es propenso a errores, es inconsistente,

es no escalable , y crea complejidad también.

E idealmente, queremos que nuestro software se construya e implemente

de manera consistente y repetible,

haciéndolo más fácil de soportar y mantener.

Y finalmente, con CI/CD, estamos haciendo pequeños,

incrementales y probando nuestros cambios

antes de integrarse con el código base principal.

Y esto te permite atrapar insectos cuando son pequeños.

y realmente simple de arreglar.

Así que echemos un vistazo a un ejemplo.

continua a un alto nivel,

y todo comienza con un repositorio de código compartido.

Así que tenemos varios desarrolladores que contribuyen a un

repositorio de código compartido como Git , que se fusiona con frecuencia

o integrando sus actualizaciones de código.

Y, por lo general, eso sería diariamente como mínimo.

Luego tenemos una etapa de construcción automatizada,

de modo que cuando aparezcan cambios en el repositorio de código,

esto desencadena una compilación automatizada del nuevo código,

y finalmente también tenemos un paso de prueba automatizado.

Entonces ejecutamos pruebas automatizadas para verificar el código localmente

antes de que se confirme en el repositorio de código maestro.

Así que pasando a nuestra entrega continua

y flujo de trabajo de implementación continua.

Después de una prueba exitosa,

el código se fusiona con el repositorio principal.

Luego se prepara para el despliegue,

para que se construya, pruebe y empaquete,

listo para el despliegue.

Y luego con la entrega continua,

generalmente hay una decisión manual.

Entonces los humanos pueden estar involucrados en la decisión.

dónde y cuándo desplegar el código,

y eso se conoce como entrega continua.

Sin embargo, esa etapa también se puede automatizar por completo.

Para que el sistema pueda implementar automáticamente el nuevo código.

tan pronto como se haya preparado para el despliegue,

y eso se conoce como despliegue continuo.

Así que echemos un vistazo a algunas de las herramientas para desarrolladores

que proporciona AWS para permitir a los desarrolladores

para comenzar con su flujo de trabajo de CI/CD.

Bueno, antes que nada, tenemos CodeCommit ,

y eso proporciona control de fuente y versión para nuestro código.

Entonces es un servicio de control de fuente administrado,

permitir que los equipos colaboren.

Y puede almacenar cualquier cosa, desde código hasta páginas HTML,

scripts , imágenes y binarios, etcétera.

Básicamente, todo lo que se necesita para que su software funcione.

Y puede almacenar todo eso en CodeCommit ,

y hacer que realice control de fuente

y control de versiones para usted.

Y en realidad es un repositorio Git privado completamente administrado,

así que si alguna vez has usado Git antes,

entonces comprenderá exactamente qué es CodeCommit .

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

y que compila el código fuente,

puede ejecutar pruebas automatizadas y produce paquetes

que están listos para implementar en su entorno.

Luego tenemos CodeDeploy ,

y que proporciona implementación automatizada,

por lo que automatiza las implementaciones de código,

y que funciona con cualquier instancia o tipo de servidor,

incluyendo EC2, Lambda,

y también puede funcionar con sus sistemas locales,

sistemas ubicados en su propio centro de datos.

Y finalmente, tenemos CodePipeline ,

que puede gestionar todo el flujo de trabajo de principio a fin,

orquestando toda la construcción, prueba,

y tubería de implementación.

Se integra con todas las demás herramientas,

entonces CodeCommit , CodeBuild y CodeDeploy ,

y en realidad puede configurarlo para implementar

la nueva versión de su aplicación cada vez

hay un cambio de código y aparece un nuevo código

en su repositorio de CodeCommit .

Y más adelante en el curso, vamos a obtener

nuestras manos sucias con todas estas herramientas,

para que pueda obtener mucha experiencia práctica

y estar muy bien preparado

para cualquier pregunta que vea sobre estos en el examen.

Así que echemos un vistazo a nuestros consejos de examen para CI y CD.

En primer lugar, la integración continua

se trata de integrar o fusionar cambios de código con frecuencia,

al menos una vez al día,

y el servicio que brindan para esto es CodeCommit .

Solo recuerda,

eso es como un repositorio Git privado administrado .

En segundo lugar, tenemos entrega continua,

y eso es todo acerca de la automatización de la construcción, prueba,

y funciones de implementación de su proceso de liberación.

Así que piensa en CodeBuild un CodeDeploy .

Y finalmente, tenemos implementación continua,

y ese es el proceso de lanzamiento totalmente automatizado

donde el código se implementa en la puesta en escena o la producción

tan pronto como sea exitoso

pasado a través de la canalización de liberación.

Y el servicio que brindan

para eso se llama CodePipeline .

Y luego lo último que quería contarte sobre

es este documento técnico de AWS, y se llama

Practicando la Integración Continua

e implementación continua en AWS.

Y este es un gran Whitepaper porque explica

las características y ventajas de utilizar la integración continua

y entrega o implementación continua,

y analiza todas las herramientas de AWS que puede utilizar

en su entorno de desarrollo de software.

Y hay un enlace a este Whitepaper

en la sección de recursos del curso,

así que si tienes tiempo,

definitivamente vale la pena leer este Whitepaper.

Y eso nos lleva al final de esta conferencia.

Si tiene alguna pregunta, por favor hágamelo saber.

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

Gracias.