[ENGLISH]

Hi, and welcome back to part 2.

So now we are ready to upload our application code

to our S3 bucket.

So head to Services and select S3.

Select your S3 bucket

where you're going to be storing the code.

Upload, Add Files,

and we are going to upload version 1 of our application,

and you'll find it in this folder here.

And it's the ZIP file that we need to upload.

So select that and Upload.

So now let's deploy this code using CodeDeploy,

and we'll just make sure that everything is working

with our CodeDeploy permissions.

So head to Services, Developer Tools,

and select CodeDeploy,

and I'm going to open it in a new tab.

Select Getting Started,

Create Application, and we'll call our application mywebapp.

Compute platform is going to be EC2

and select Create Application.

Next we'll create a deployment group.

We'll call it mydg.

Under Service Role,

this is where we select the service role

that was created by the CloudFormation template.

So just go ahead and select that.

As we only have 1 EC2 instance,

we'll use an in-place deployment type.

Environment configuration will be EC2 instances,

and then down here, we'll select our instance.

Now the CloudFormation template

has added a tag for us

and the tag was Name

and the value is CodeDeployDemo.

Scroll down until you get to load balancer,

and we're going to uncheck Enable Load Balancing

because we don't have a load balancer.

And if you don't remember to uncheck this,

it will not work.

So now Create Deployment Group.

And then to start our deployment,

just Create Deployment.

Our deployment group should already be selected,

and it's mydg.

The revision, or the version of our application code,

is stored in S3.

And we need to provide the S3 location

where our revision is stored.

So if you come back to your S3 bucket

and select your file

and then select Copy S3 URI,

paste that into your revision location,

scroll down to the bottom, and Create Deployment.

So that is our deployment created

and it's in progress at the moment.

And it will just take a few moments to complete.

And there we go,

it's completed and we have a status of succeeded.

So before we move on,

let's test that it really worked as expected.

So if you head to Services and select EC2,

I'm going to open it in a new tab,

select your instance ID,

copy the public IP address.

And we'll open a new tab,

paste the IP into our browser and hit Enter.

And there we go.

There is our application.

So far, so good.

So CodeDeploy is all working

and now we are ready

to tie it all together with CodePipeline.

And we are going to use CodePipeline

to manage the deployment

of a new version of our application.

So first of all,

we need to upload version 2 of our code to S3.

So come back to your S3 bucket.

And if you remember,

we have version control enabled on our bucket,

so we don't need to delete this old version.

We can just go ahead and upload the new one,

even though it's going to have

exactly the same name.

So select Add Files

and we want to select version 2 of our code

from this directory.

So select mywebapp.zip from the version 2 directory.

Once your file has been uploaded,

you can select Show Versions

and it will show you the 2 versions of our file.

So now let's head back to our CodeDeploy tab

and you'll find CodePipeline,

down here on the left-hand side,

under the developer tools.

So select Getting Started, Create Pipeline.

We'll call it mypipeline.

Stick with the default

of creating a New Service Role for CodePipeline.

Hit Next.

The source provider is going to be S3.

We'll select our bucket name,

and we need to provide the name of our application revision.

And it's going to be mywebapp.zip.

Down here we can select a detection option.

So it says, choose a detection mode

to automatically start your pipeline

when a change occurs in the source code.

So it's going to use CloudWatch events

to automatically start the pipeline

when a change occurs to my code.

Hit Next.

Under Build Provider,

CodePipeline integrates with both CodeBuild

and Jenkins as well,

if you have code that needs to be built

as part of your pipeline,

but we don't, so we're just going to select,

Skip Build Stage and just acknowledge.

Under Deployment Provider,

this is going to be CodeDeploy.

Make sure you've got the correct region.

It should be US East 1.

Under Application Name, select mywebapp.

Under deployment group, select mydg.

Hit Next.

We can review our settings and Create Pipeline.

And there is our pipeline.

And it does just take a few minutes

to complete everything.

So just be patient and hopefully it should all work.

So there we go.

It's located the source code

and it's triggered the CodeDeploy deployment,

which is in progress now.

So there we go, it has succeeded.

So we've manually

triggered our deployment using CodePipeline.

So let's go back to our little webpage.

We can refresh the screen.

And there we go,

it's showing version 2 of our application,

which we deployed manually using CodePipeline.

So now, finally, we get to do the really fun bit,

and we're going to upload version 3 of our code to S3,

and fingers crossed, if it's all working correctly,

then CodePipeline will be alerted

that the new code has appeared

and it will automatically deploy

the new version to our instance.

So let's head over to our S3 bucket.

We'll upload version 3,

and here it is in my version 3 folder,

and quickly come back to CodePipeline

and you should see straight away

a new pipeline has been triggered.

It's located the new source code

and CodeDeploy is deploying the new code.

It's still in progress

and there we go. It has succeeded.

So once everything is completed,

we can then just go back to our browser tab,

which is showing our application,

reload the page, and there we go.

There is version 3 of our application.

We updated our source code in S3.

It triggered the automatic deployment from CodePipeline,

which used CodeDeploy,

to deploy the new version of our code.

And if you're ready to delete everything,

you can just head back to your pipelines.

You can select your pipeline and delete it.

Under CodeDeploy, you can delete your application.

From CloudFormation, you can delete your stack.

And from S3, you can delete your bucket as well.

So for my CodePipeline exam tips,

just remember that we use CodePipeline for CI/CD.

So it's a continuous integration

and continuous delivery service,

and it orchestrates the end-to-end software release process.

And it allows you to automate everything,

so it avoids using manual deployments,

and automatically triggers your pipeline

when new source code appears.

So when we added new source code to our S3 bucket,

the pipeline was automatically triggered.

And it is integrated

with lots of different AWS and third-party tools as well.

So things like CodeCommit, CodeBuild, CodeDeploy,

which we used in the lab,

GitHub, Jenkins, Elastic Beanstalk, CloudFormation, Lambda

and Elastic Container Service as well.

So that's 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, y bienvenidos de nuevo a la parte 2.

Así que ahora estamos listos para cargar nuestro código de aplicación

a nuestro cubo S3.

Dirígete a Servicios y selecciona S3.

Seleccione su cubo S3

donde vas a almacenar el código.

Cargar, agregar archivos,

y vamos a subir la versión 1 de nuestra aplicación,

y lo encontrará en esta carpeta aquí.

Y es el archivo ZIP que necesitamos cargar.

Así que seleccione eso y Subir.

Así que ahora implementemos este código usando CodeDeploy,

y nos aseguraremos de que todo funcione

con nuestros permisos de CodeDeploy.

Así que dirígete a Servicios, Herramientas para desarrolladores,

y seleccione CodeDeploy,

y voy a abrirlo en una nueva pestaña.

Seleccione Primeros pasos,

Create Application, y llamaremos a nuestra aplicación mywebapp.

La plataforma informática será EC2

y seleccione Crear aplicación.

A continuación, crearemos un grupo de implementación.

Lo llamaremos mydg.

En Rol de servicio,

aquí es donde seleccionamos el rol de servicio

que fue creado por la plantilla de CloudFormation.

Así que sigue adelante y selecciona eso.

Como solo tenemos 1 instancia EC2,

usaremos un tipo de implementación en el lugar.

La configuración del entorno serán instancias EC2,

y luego aquí abajo, seleccionaremos nuestra instancia.

Ahora la plantilla de CloudFormation

ha agregado una etiqueta para nosotros

y la etiqueta era Nombre

y el valor es CodeDeployDemo.

Desplácese hacia abajo hasta llegar al equilibrador de carga,

y vamos a desmarcar Enable Load Balancing

porque no tenemos un balanceador de carga.

Y si no recuerdas desmarcar esto,

no funcionará

Así que ahora cree un grupo de implementación.

Y luego, para comenzar nuestro despliegue,

simplemente Crear Despliegue.

Nuestro grupo de implementación ya debería estar seleccionado,

y es mydg.

La revisión, o la versión de nuestro código de aplicación,

se almacena en S3.

Y necesitamos proporcionar la ubicación S3

donde se almacena nuestra revisión.

Entonces, si regresa a su cubo S3

y seleccione su archivo

y luego seleccione Copiar S3 URI,

péguelo en su ubicación de revisión,

desplácese hacia abajo hasta la parte inferior y Crear implementación.

Así que ese es nuestro despliegue creado.

y está en progreso en este momento.

Y solo tomará unos minutos completarlo.

Y ahí vamos,

se completó y tenemos un estado de éxito.

Así que antes de continuar,

probemos que realmente funcionó como se esperaba.

Entonces, si se dirige a Servicios y selecciona EC2,

Voy a abrirlo en una nueva pestaña,

seleccione su ID de instancia,

copiar la dirección IP pública.

Y abriremos una nueva pestaña,

pega la IP en nuestro navegador y presiona Enter.

Y ahí vamos.

Ahí está nuestra aplicación.

Hasta ahora, todo bien.

Entonces CodeDeploy está funcionando

y ahora estamos listos

para unirlo todo con CodePipeline.

Y vamos a usar CodePipeline

para gestionar el despliegue

de una nueva versión de nuestra aplicación.

Entonces, en primer lugar,

necesitamos subir la versión 2 de nuestro código a S3.

Así que vuelva a su cubo S3.

Y si recuerdas,

tenemos el control de versiones habilitado en nuestro cubo,

por lo que no necesitamos eliminar esta versión anterior.

Podemos seguir adelante y cargar el nuevo,

aunque va a tener

exactamente el mismo nombre.

Así que seleccione Agregar archivos

y queremos seleccionar la versión 2 de nuestro código

de este directorio.

Así que seleccione mywebapp.zip del directorio de la versión 2.

Una vez que su archivo ha sido cargado,

puede seleccionar Mostrar versiones

y te mostrará las 2 versiones de nuestro archivo.

Así que ahora volvamos a nuestra pestaña CodeDeploy

y encontrará CodePipeline,

abajo en el lado izquierdo,

bajo las herramientas del desarrollador.

Así que seleccione Primeros pasos, Crear canalización.

Lo llamaremos mypipeline.

Seguir con el valor predeterminado

de crear un nuevo rol de servicio para CodePipeline.

Presiona Siguiente.

El proveedor de origen será S3.

Seleccionaremos el nombre de nuestro cubo,

y necesitamos proporcionar el nombre de la revisión de nuestra aplicación.

Y va a ser mywebapp.zip.

Aquí abajo podemos seleccionar una opción de detección.

Entonces dice, elija un modo de detección

para iniciar automáticamente su canalización

cuando se produce un cambio en el código fuente.

Así que va a usar eventos de CloudWatch

para iniciar automáticamente la canalización

cuando se produce un cambio en mi código.

Presiona Siguiente.

En Proveedor de compilación,

CodePipeline se integra con CodeBuild

y Jenkins también,

si tienes un código que necesita ser construido

como parte de su tubería,

pero no lo hacemos, así que solo vamos a seleccionar,

Omita la etapa de compilación y simplemente reconozca.

En Proveedor de implementación,

esto va a ser CodeDeploy.

Asegúrate de tener la región correcta.

Debería ser EE.UU. Este 1.

En Nombre de la aplicación, seleccione mywebapp.

En grupo de implementación, seleccione mydg.

Presiona Siguiente.

Podemos revisar nuestra configuración y Crear Pipeline.

Y ahí está nuestra tubería.

Y solo toma unos minutos

para completar todo.

Así que ten paciencia y espero que todo funcione.

Así que ahí vamos.

Se encuentra el código fuente.

y activó la implementación de CodeDeploy,

que está en curso ahora.

Así que ahí vamos, ha tenido éxito.

Así que hemos manualmente

desencadenó nuestra implementación mediante CodePipeline.

Así que volvamos a nuestra pequeña página web.

Podemos refrescar la pantalla.

Y ahí vamos,

está mostrando la versión 2 de nuestra aplicación,

que implementamos manualmente usando CodePipeline.

Así que ahora, finalmente, podemos hacer la parte realmente divertida,

y vamos a subir la versión 3 de nuestro código a S3,

y los dedos cruzados, si todo funciona correctamente,

entonces CodePipeline será alertado

que ha aparecido el nuevo código

y se desplegará automáticamente

la nueva versión a nuestra instancia.

Así que vayamos a nuestro cubo S3.

Subiremos la versión 3,

y aquí está en mi carpeta de la versión 3,

y regrese rápidamente a CodePipeline

y deberías ver de inmediato

se ha activado una nueva canalización.

Se encuentra el nuevo código fuente.

y CodeDeploy está implementando el nuevo código.

todavía está en progreso

y ahí vamos Ha tenido éxito.

Así que una vez que todo esté terminado,

podemos volver a la pestaña de nuestro navegador,

que está mostrando nuestra aplicación,

recargar la página, y ahí vamos.

Existe la versión 3 de nuestra aplicación.

Actualizamos nuestro código fuente en S3.

Desencadenó la implementación automática de CodePipeline,

que utilizó CodeDeploy,

para implementar la nueva versión de nuestro código.

Y si estás listo para borrar todo,

puede regresar a sus tuberías.

Puede seleccionar su embudo y eliminarlo.

En CodeDeploy, puede eliminar su aplicación.

Desde CloudFormation, puede eliminar su pila.

Y desde S3, también puede eliminar su cubo.

Entonces, para mis consejos para el examen de CodePipeline,

solo recuerde que usamos CodePipeline para CI/CD.

Así que es una integración continua.

y servicio de entrega continua,

y organiza el proceso de lanzamiento de software de extremo a extremo.

Y te permite automatizar todo,

por lo que evita el uso de implementaciones manuales,

y activa automáticamente su canalización

cuando aparece un nuevo código fuente.

Entonces, cuando agregamos un nuevo código fuente a nuestro depósito S3,

la canalización se activó automáticamente.

Y está integrado

con muchas herramientas diferentes de AWS y de terceros también.

Entonces cosas como CodeCommit, CodeBuild, CodeDeploy,

que usamos en el laboratorio,

GitHub, Jenkins, Elastic Beanstalk, CloudFormation, Lambda

y Elastic Container Service también.

Eso es todo por esta lección.

Si tienes alguna pregunta,

por favor hágamelo saber

De lo contrario, te veré en la próxima lección.

Gracias.