

MLOps (Machine Learning Operations)

Contexto: Vivimos en una era de **datos masivos, procesamiento asequible y avances rápidos en ML**, lo que impulsa a las empresas a desarrollar modelos predictivos para generar valor comercial.

¿Qué es MLOps?

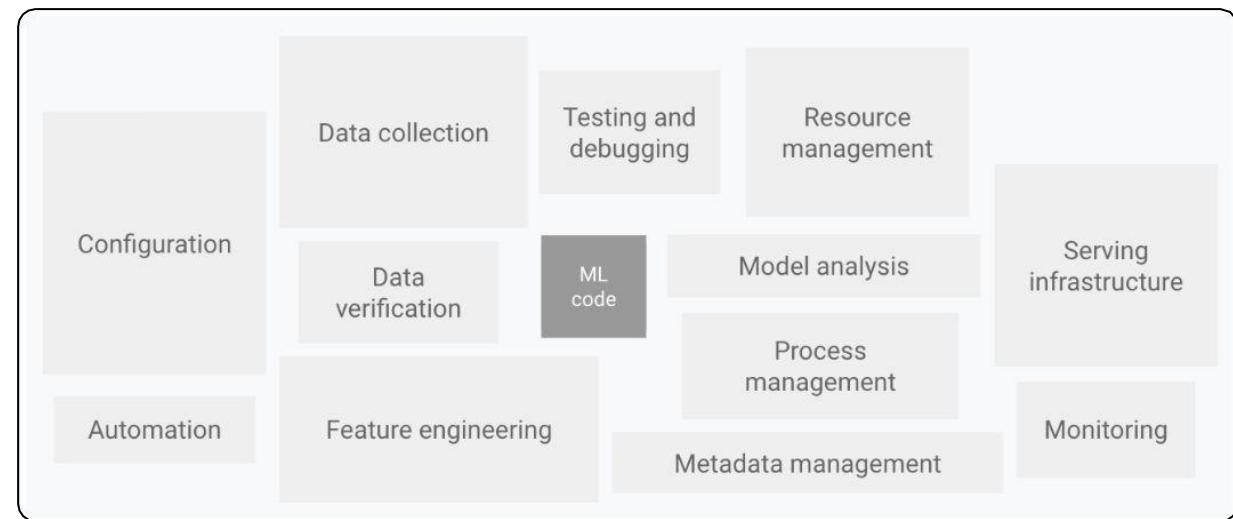
Un enfoque que **integra principios de DevOps en ML**, buscando la automatización y supervisión en todas las fases del ciclo de vida de ML, desde el desarrollo hasta la operación.

Un sistema de ML real **aborda mucho más que solo el código**

ML. Los componentes críticos que lo rodean incluyen configuración, automatización, gestión de datos, pruebas, y mucho más.

Para manejar la complejidad de estos sistemas, se aplican principios de DevOps adaptados al ML, conocidos como MLOps. Esto incluye prácticas esenciales como: **CI (Integración Continua)**, **CD (Despliegue Continuo)** y **CT (Entrenamiento Continuo)**.

Desafíos: Más allá de desarrollar un modelo con buen rendimiento, el reto está en **crear y mantener sistemas de ML que operen eficientemente en producción**.



MLOps (Machine Learning Operations)

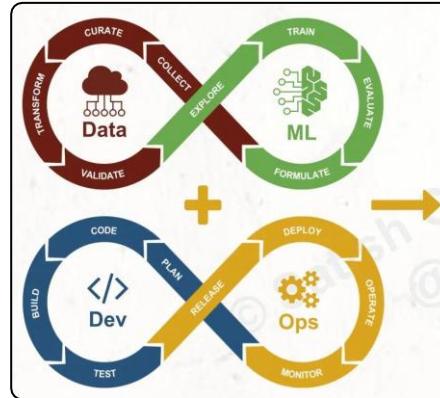
DevOps en Software Tradicional: Enfatiza la integración continua (CI) y la entrega continua (CD) para acelerar el desarrollo, mejorar la implementación, y garantizar actualizaciones confiables.

Diferencias Clave:

Integración Continua (CI): En ML, implica la validación de código, datos, esquemas, y modelos.

Entrega Continua (CD): En ML, abarca el despliegue automatizado de sistemas completos de entrenamiento y predicción.

Entrenamiento Continuo (CT): Aspecto único de MLOps que se enfoca en el reentrenamiento y despliegue automáticos de modelos.



MLOps en Sistemas de ML

Aplica prácticas de DevOps adaptadas a las peculiaridades de los sistemas de ML, incluyendo:

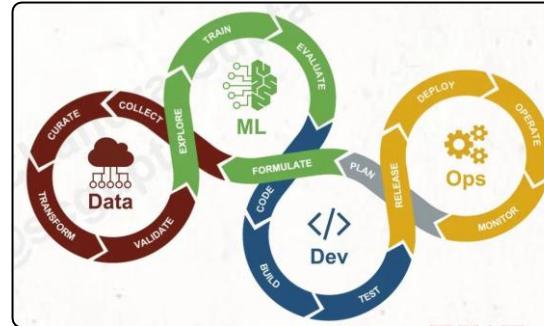
Equipos Multidisciplinarios: Integración de científicos de datos y expertos en ML que pueden no tener experiencia en ingeniería de software a nivel de producción.

Desarrollo Experimental: Proceso iterativo y experimental para explorar diferentes funciones, algoritmos, y técnicas de modelado, enfatizando la importancia de la reproducibilidad y la reutilización del código.

Pruebas Complejas: Validación de datos y modelos más allá de las pruebas de integración y unidad típicas en software convencional.

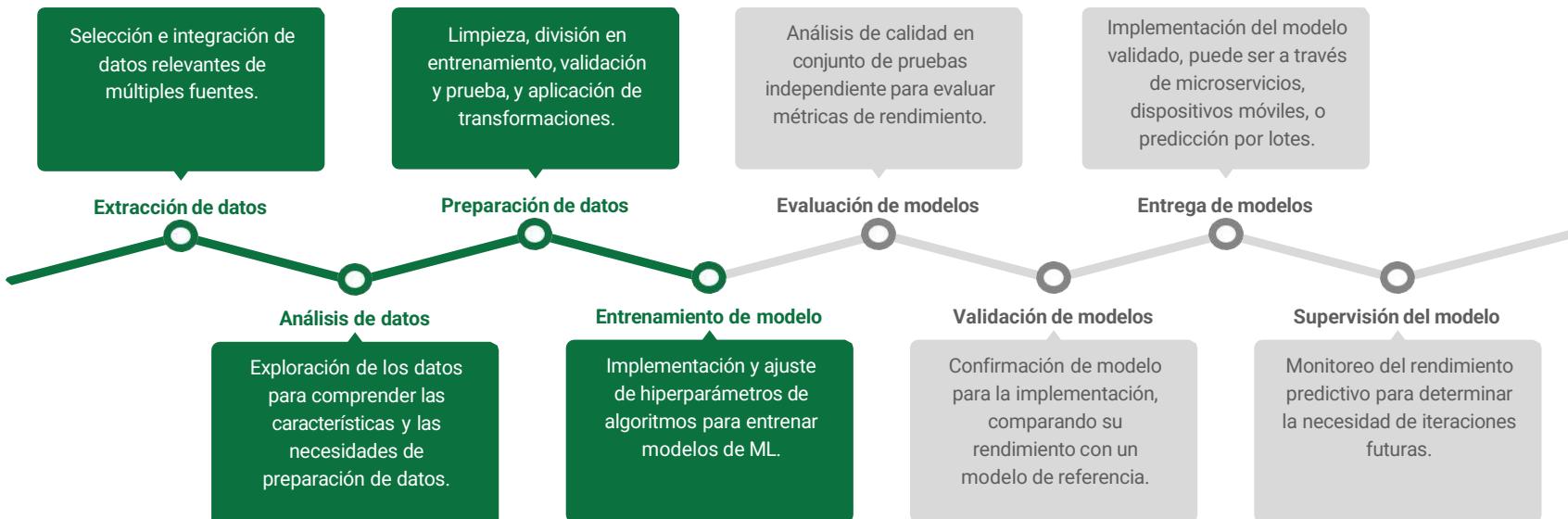
Implementación Desafiante: Necesidad de automatizar completamente las canalizaciones de entrenamiento y despliegue de modelos, aumentando la complejidad.

Mantenimiento en Producción: Supervisión continua del rendimiento del modelo y la calidad de los datos para manejar el decaimiento del modelo.



MLOps (Machine Learning Operations)

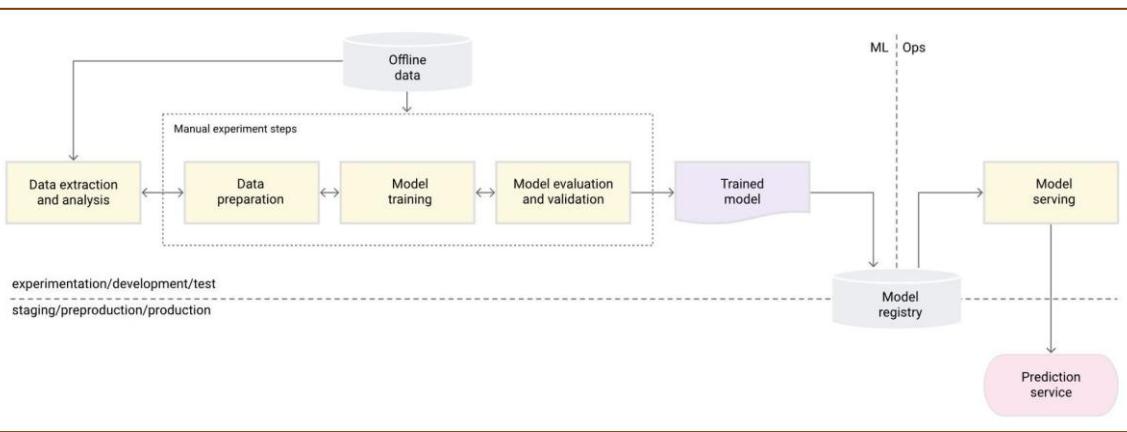
En cualquier proyecto de ML, **después de definir el caso de uso empresarial y establecer los criterios de éxito**, el proceso de entrega de un modelo de ML a la producción implica los siguientes pasos.



MLOps (Machine Learning Operations)

MLOps Nivel 0 - Proceso Manual

MLOps nivel 0 se caracteriza por un **proceso completamente manual** para desarrollar e implementar modelos de machine learning, representando el **nivel de madurez básico**.



Foco en la Implementación de Servicio de Predicción: El énfasis está en desplegar modelos como servicios de predicción, sin establecer prácticas de CI/CD.

Falta de Supervisión Activa: La ausencia de monitoreo continuo del modelo en producción puede ocasionar que la degradación del rendimiento pase desapercibida.

Características

Proceso Manual Interactivo: Las tareas de análisis y validación se realizan a mano usando código en notebooks, lo que implica un proceso interactivo y basado en scripts.

Desconexión entre Desarrollo y Operaciones: Existe una separación clara entre los equipos que crean modelos y los que los implementan, con una transferencia manual de los modelos para su despliegue.

Iteraciones de Versiones Poco Frecuentes: Los modelos se actualizan o reentrenan raramente, resultando en la implementación de nuevas versiones solo ocasionalmente.

Ausencia de CI/CD: No se practica la integración ni la entrega continua, principalmente actualizaciones y cambios. Baja frecuencia de

MLOps (Machine Learning Operations)

MLOps Nivel 0 - Proceso Manual

El nivel 0, común en **empresas que inician con ML**, presenta retos significativos, como la falta de adaptación de los modelos a los cambios en el entorno y los datos, lo que a menudo lleva a un rendimiento deficiente en producción.

Pasos para Mantener la Exactitud del Modelo

Supervisión Activa: Monitorear constantemente la calidad y el rendimiento del modelo en producción para identificar y corregir la degradación y otros problemas.

Reentrenamiento Frecuente: Actualizar los modelos con los datos más recientes para reflejar patrones emergentes y cambios en las tendencias.

Experimentación Continua: Probar continuamente nuevas implementaciones, técnicas de ingeniería de atributos, arquitecturas de modelos y ajustes de hiperparámetros para mejorar la exactitud.

MLOps para Mejora Continua

Implementación de CI/CD y CT: Adoptar prácticas de MLOps para establecer procesos automáticos de integración continua (CI), entrega continua (CD) y entrenamiento continuo (CT), permitiendo la rápida prueba, compilación, y despliegue de modelos.

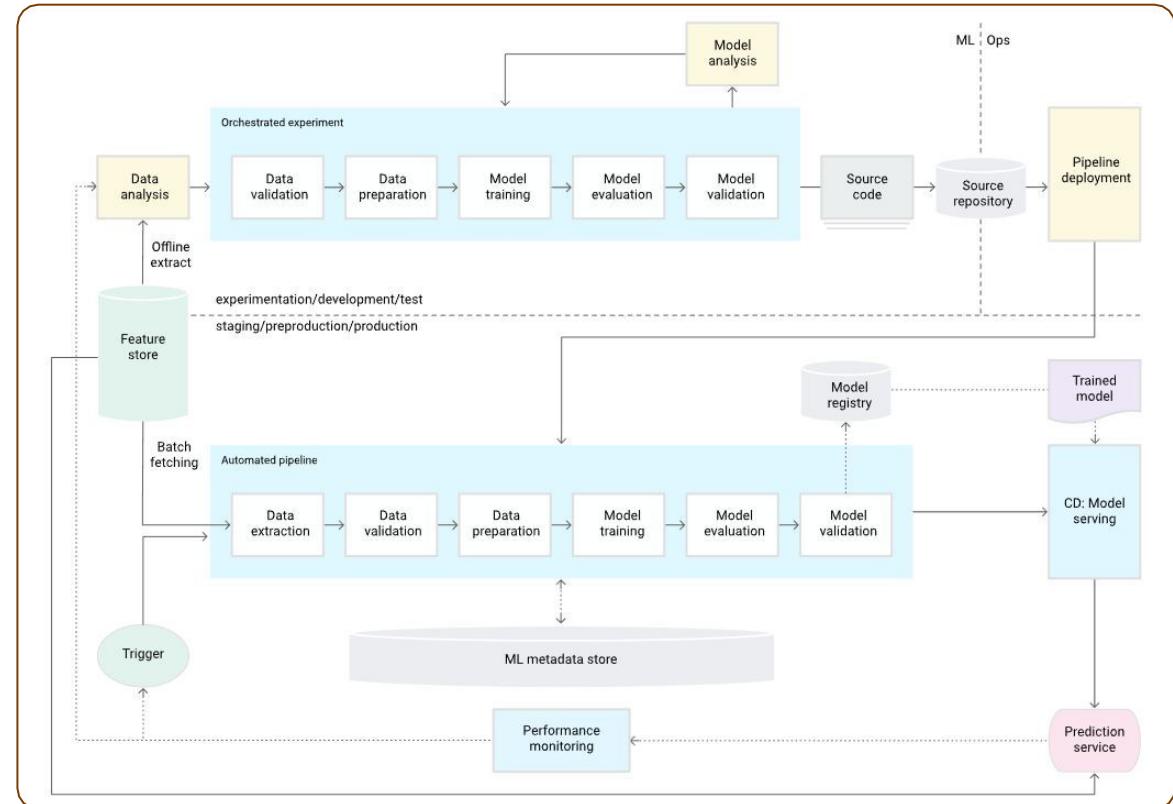
Canalización de Entrenamiento Automatizada: Crear una canalización automatizada de entrenamiento de ML para facilitar el reentrenamiento y la implementación de modelos con eficiencia y precisión.

MLOps (Machine Learning Operations)

Nivel 1 de MLOps: Automatización de la canalización de ML

El objetivo del nivel 1 es realizar un **entrenamiento continuo** del modelo mediante la automatización de la canalización de ML, lo que te permite lograr una **entrega continua** del servicio de predicción del modelo.

Para automatizar el proceso de uso de datos nuevos a fin de volver a entrenar los modelos en la producción, debes **ingresar los datos automatizados** y los pasos de **validación de modelos** en la canalización, y los **activadores de canalización** y la **administración de metadatos**.



MLOps (Machine Learning Operations)

Nivel 1 de MLOps: Automatización de la canalización de ML

Características

Experimentación Rápida: Los pasos experimentales son organizados y automatizados, facilitando iteraciones rápidas y una transición eficiente hacia la producción.

Entrenamiento Continuo en Producción (CT): Automatización del reentrenamiento de modelos con datos recientes en producción, utilizando activadores de canalización.

Simetría Experimental-Operacional: Unificación de las canalizaciones entre desarrollo y producción, clave para la práctica de MLOps y para asegurar consistencia y eficiencia operativa.

Implementación de Canalización Completa: Avance del Nivel 0, implementando no solo el modelo sino toda la canalización de entrenamiento de manera automática y recurrente para mantener los servicios de predicción actualizados.

Entrega Continua de Modelos: Implementación automatizada de modelos como servicios de predicción, utilizando actualizados, facilitando la entrega continua de servicios de predicción.

Código Modular para Componentes y Canalizaciones: Modularización del código fuente para reutilización y acoplamiento fácil en canalizaciones, con componentes organizados en contenedores.

MLOps (Machine Learning Operations)

Nivel 1 de MLOps: Automatización de la canalización de ML

Componentes Clave para el Entrenamiento Continuo en MLOps Nivel 1

Procesos automáticos esenciales para asegurar la calidad y relevancia de los modelos entrenados con datos nuevos.

Validación de Datos: Detecta sesgos y anomalías en los datos antes del entrenamiento, evaluando si los datos cumplen con el esquema esperado y si reflejan cambios significativos que requieren reentrenamiento.

Validación de Modelos: Evalúa la calidad predictiva y la coherencia del rendimiento del modelo en diversos segmentos de datos antes de su despliegue en producción.

Feature Store: Repositorio centralizado para gestionar y reutilizar características de datos, facilitando la consistencia entre entrenamiento y predicción, y apoyando tanto el entrenamiento continuo como las predicciones en tiempo real.

Administración de Metadatos: Registro detallado de la ejecución de canalizaciones para facilitar el linaje, la reproducibilidad, y la depuración de modelos. Incluye información sobre versiones de canalización, argumentos de parámetros, artefactos producidos, y métricas de evaluación.

Activadores de Canalización: Mecanismos para iniciar el reentrenamiento automático de modelos basado en diversas condiciones, como:

- A Demanda: Ejecución manual según sea necesario.
- Programación Regular: Basada en la disponibilidad sistemática de datos nuevos.
- Disponibilidad de Datos Nuevos: Reentrenamiento desencadenado por la llegada de nuevos datos.
- Deterioro del Rendimiento del Modelo: Iniciado por una caída notable en el rendimiento del modelo.
- Desvío de Conceptos: Cambios significativos en las distribuciones de datos que indican la necesidad de reentrenamiento.

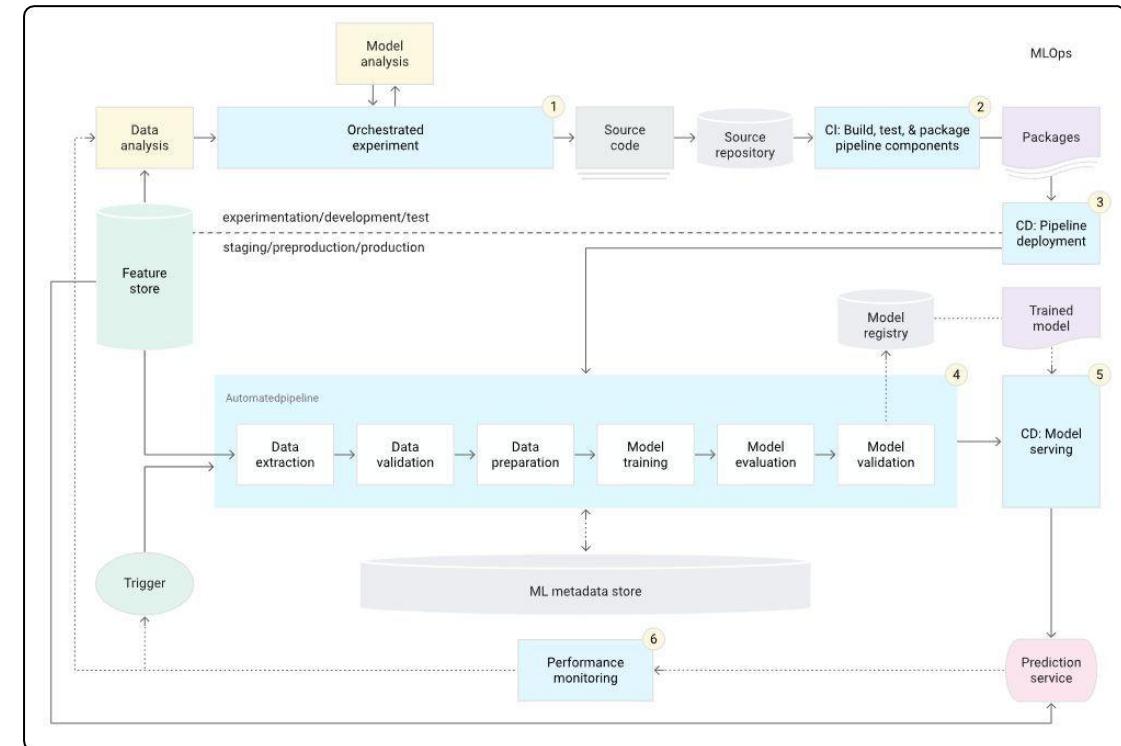
MLOps (Machine Learning Operations)

Nivel 2 de MLOps: Automatización de la canalización de CI/CD

Un sistema de **CI/CD robusto y automatizado** es crucial para la actualización rápida y fiable de canalizaciones de ML en producción.

Facilita a los científicos de datos la **experimentación y aplicación de nuevas ideas** en **ingeniería de atributos, arquitecturas de modelos y ajustes de hiperparámetros**.

Este enfoque permite la **compilación, evaluación y despliegue automáticos** de nuevos componentes de la canalización en el entorno de producción, asegurando la eficiencia y agilidad en el desarrollo y operación de modelos de ML.



MLOps (Machine Learning Operations)

Nivel 2 de MLOps: Automatización de la canalización de CI/CD

Características

Desarrollo y Experimentación: Inicio con la prueba iterativa de algoritmos y modelos nuevos, cuyos resultados se organizan y envían a un repositorio de código.

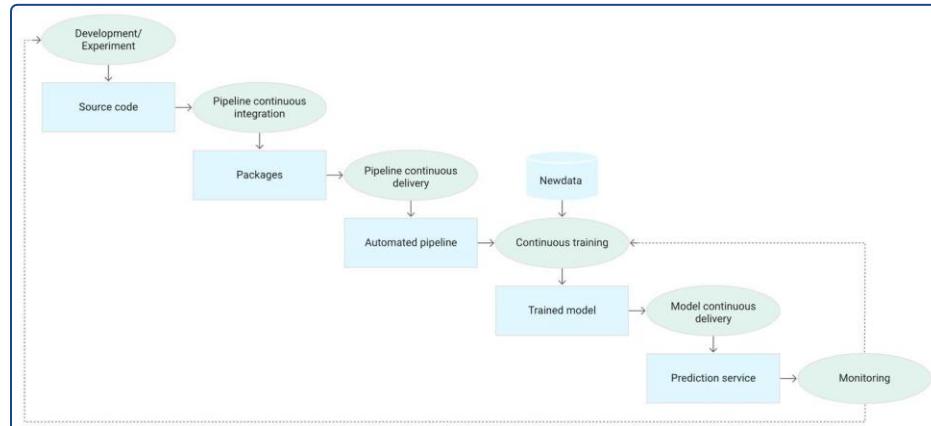
Integración Continua (CI) de la Canalización: Compilación del código fuente y ejecución de pruebas para generar componentes de canalización, que incluyen paquetes, ejecutables y artefactos.

Entrega Continua (CD) de la Canalización: Despliegue de los artefactos de CI en el entorno de destino, resultando en la implementación de una canalización actualizada con el modelo nuevo.

Activación Automática: Ejecución automática de la canalización en producción, ya sea por programación o activadores específicos, produciendo un modelo entrenado para el registro de modelos.

Entrega Continua de Modelos: Implementación del modelo entrenado como un servicio de predicción, finalizando con el despliegue del servicio de predicción del modelo.

Supervisión: Monitoreo del rendimiento del modelo en producción para identificar la necesidad de reentrenamientos o ajustes basados en el análisis de datos activos.



MLOps (Machine Learning Operations)

Nivel 2 de MLOps: Automatización de la canalización de CI/CD

Integración Continua (CI)

Compilación y Empaqueado: Al confirmarse nuevo código, la canalización y sus componentes se compilan, evalúan y empaquetan automáticamente, incluyendo la creación de paquetes, imágenes de contenedores y ejecutables.

Pruebas de CI Incluyen:

Pruebas de Unidad para Feature Engineering: Verificar la lógica detrás de la creación y transformación de características.

Pruebas de Unidad para Métodos del Modelo: Evaluar funciones específicas del modelo, como la correcta implementación de codificación one-hot para datos categóricos.

Convergencia del Entrenamiento de Modelos: Confirmar que la pérdida del modelo disminuye adecuadamente a lo largo de las iteraciones sin sobreajustarse.

Validación de Valores del Modelo: Asegurar que el entrenamiento del modelo no resulte en valores NaN por errores como división por cero.

Producción de Artefactos Esperados: Comprobar que cada componente de la canalización genera los artefactos previstos.

Integración de Componentes de la Canalización: Evaluar la correcta interacción y funcionamiento conjunto de los distintos componentes de la canalización.

MLOps (Machine Learning Operations)

Nivel 2 de MLOps: Automatización de la canalización de CI/CD

Entrega Contínua (CD)

Automatizar el despliegue de nuevas canalizaciones en el entorno de destino, proporcionando servicios de predicción con modelos recién entrenados.

Consideraciones Clave para la Entrega Continua

Compatibilidad del Modelo con Infraestructura: Asegurar que los requisitos del modelo, como paquetes, memoria y recursos de procesamiento, estén disponibles en el entorno de destino.

Validación de la API de Servicio: Probar la API con datos de entrada esperados para verificar las respuestas correctas del servicio de predicción.

Evaluación del Rendimiento del Servicio de Predicción: Realizar pruebas de carga para medir métricas críticas, incluyendo consultas por segundo (QPS) y latencia.

Validación de Datos para Reentrenamiento o Predicción por Lotes: Verificar la adecuación y calidad de los datos utilizados.

Cumplimiento de Objetivos de Rendimiento Predictivo: Asegurar que los modelos satisfacen los estándares de rendimiento antes de su implementación.

Estrategias de Implementación

Automatizada en Entorno de Prueba: Despliegue activado automáticamente con el envío de código a la rama de desarrollo.

Semiatomatizada en Entorno de Preproducción: Despliegue activado tras la fusión del código en la rama principal y aprobación de los revisores.

Manual en Entorno de Producción: Implementación cuidadosa en producción tras pruebas exitosas en entornos anteriores.



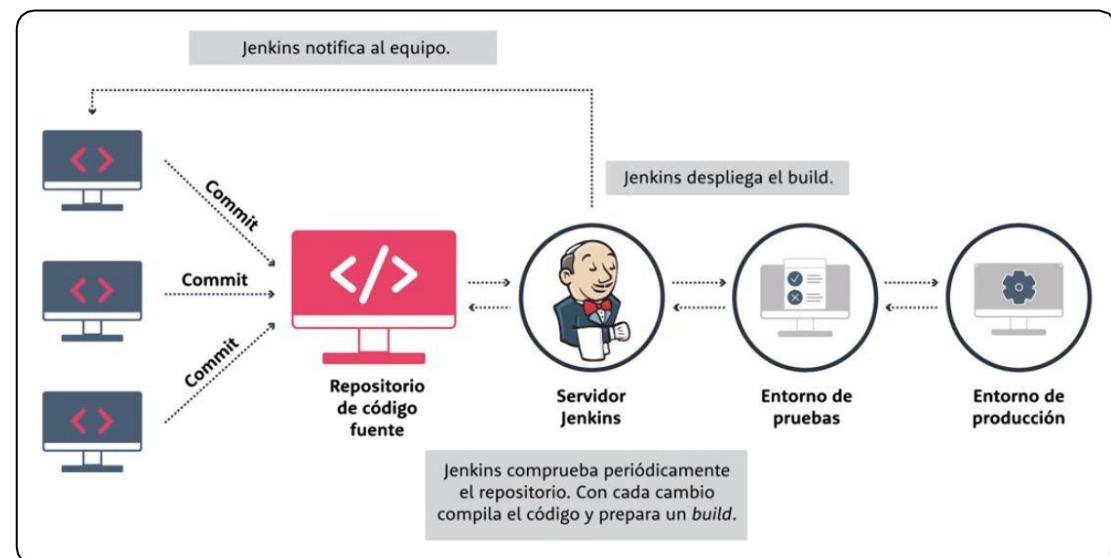
CI/CD con Jenkins para el Desarrollo Local

Jenkins es un **servidor de automatización de código abierto** que se usa ampliamente para **automatizar partes del desarrollo de software** relacionadas con la construcción, prueba e implementación, facilitando la integración y entrega continuas (CI/CD).

En el contexto de MLOps, que se centra en **automatizar y mejorar el ciclo de vida de los modelos de aprendizaje automático**, Jenkins permite lo siguiente:

Jenkins automatiza el **proceso de integración continua (CI) para modelos de aprendizaje automático (ML)** mediante la gestión de cambios de código de múltiples contribuyentes en un único proyecto de software.

Jenkins automatiza la **entrega continua (CD) de modelos de ML**, garantizando su implementación confiable en diferentes entornos (desarrollo, puesta en escena, producción) con las configuraciones adecuadas.





CI/CD con Jenkins para el Desarrollo Local

Instalar Jenkins en máquina local

Utilizar la versión de LTS (Long-Term-Support) de Jenkins como la imagen base

Preparar un entorno de Jenkins que pueda trabajar eficientemente con Docker

Instalar AWS CLI para interactuar con los Servicios Web de Amazon

Instalar el instalador de paquetes de Python3, pip

Instalar GnuPG para la gestión de claves y operaciones de seguridad

Añadir el usuario de jenkins al grupo docker

```
FROM jenkins/jenkins:lts
USER root

RUN /usr/local/bin/install-plugins.sh
    docker-workflow:1.26

RUN apt-get update -qq \
    && apt-get install -qyy apt-transport-https ca-certificates curl
gnupg2 software-properties-common
RUN curl -fSSL https://download.docker.com/linux/debian/gpg \
apt-key add -

RUN add-apt-repository \
    "deb [arch=amd64] https://download.docker.com/linux/debian\n$ (lsb_release -cs) \
    stable"

RUN apt-get update -qq \
    && apt-get install -y docker-ce docker-ce-cli containerd.io

RUN apt install awscli -y
RUN apt-get install python3-pip -y
RUN apt-get install -y gnupg
RUN usermod -aG docker jenkins
```

Dockerfile

plugins.txt	U	●	Dock
Jenkins_CI-CD > Installation >			
1	docker-workflow		
2	git:latest		



CI/CD con Jenkins para el Desarrollo Local

Instalar Jenkins en máquina local

Definir la **versión de la sintaxis de Docker Compose** que se está utilizando.

Dentro de **services**, se define un servicio llamado jenkins. En Docker Compose, un servicio es básicamente un contenedor en ejecución que forma parte de la aplicación.

image: jenkins-img indica que el servicio debe usar una imagen llamada jenkins-img.
container_name: jenkins-container asigna un nombre específico al contenedor que se va a crear a partir de este servicio.

build: ./ instruye a Docker Compose a construir la imagen del servicio usando el Dockerfile ubicado en el directorio actual (./)

Los **volumenes** permiten compartir archivos entre el contenedor y el sistema anfitrión o entre contenedores. En este caso, se montan dos volúmenes:

/var/run/docker.sock:/var/run/docker.sock monta el socket de Docker del anfitrión dentro del contenedor. Esto permite que el contenedor de Jenkins controle el **daemon de Docker** en el anfitrión, lo cual es necesario para que Jenkins pueda crear y gestionar contenedores Docker.

./jenkins_home:/var/jenkins_home monta un directorio llamado jenkins_home en el directorio actual del sistema anfitrión al directorio /var/jenkins_home dentro del contenedor. Esto es utilizado para **almacena los datos de Jenkins** (como configuraciones, trabajos, y plugins) de manera persistente, permitiendo que los datos sobrevivan reinicios del contenedor.

version: "3.0"
services:
 jenkins:
 image: jenkins-img
 container_name: jenkins-container
 build: ./
 volumes:
 - /var/run/docker.sock:/var/run/docker.sock
 - ./jenkins_home:/var/jenkins_home
 ports:
 - "8081:8080"

docker-compose.yml



CI/CD con Jenkins para el Desarrollo Local

Instalar Jenkins en máquina local

"**8081:8080**" indica que el puerto 8080 del contenedor (donde Jenkins escucha por defecto) se mapea al puerto 8081 del sistema anfitrión. Esto permite acceder a la interfaz web de Jenkins navegando a <http://localhost:8081> en el navegador del sistema anfitrión.

El comando **docker-compose up** se utiliza para iniciar y ejecutar toda la aplicación definida por un archivo docker-compose.yml.

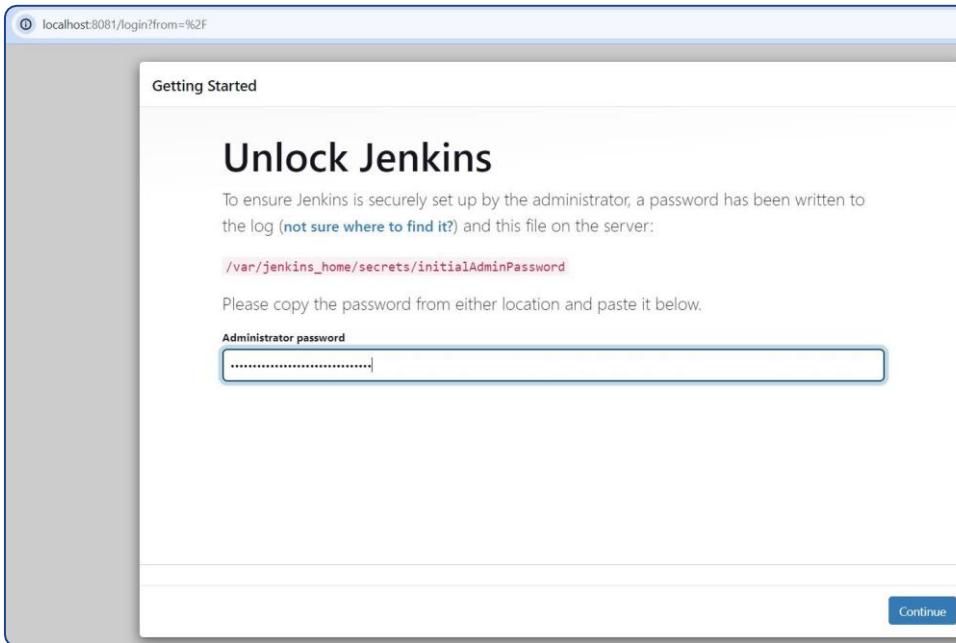
```
version: "3.0"
services:
  jenkins:
    image: jenkins-img
    container_name: jenkins-container
    build: .
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
      - ./jenkins_home:/var/jenkins_home
    ports:
      - "8081:8080"
```

```
PS D:\DMC\Jenkins_CI-CD\Installation> docker-compose up
[+] Running 1/1
  ! Jenkins Warning
[+] Building 8.9s (16/16) FINISHED
=> [jenkins internal] load .dockerignore
=> => transferring context: 2B
=> [jenkins internal] load build definition from Dockerfile
=> => transferring dockerfile: 1.99kB
=> [jenkins internal] load metadata for docker.io/jenkins/je
=> [jenkins internal] load build context
=> => transferring context: 32B
=> [jenkins 1/11] FROM docker.io/jenkins/jenkins:lts
```

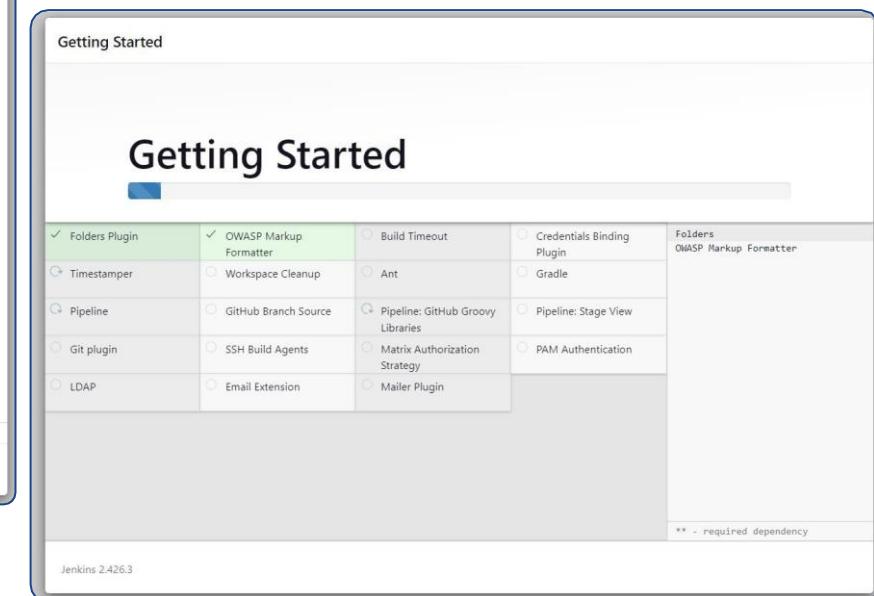


CI/CD con Jenkins para el Desarrollo Local

Instalar Jenkins en máquina local



The screenshot shows the Jenkins 'Getting Started' screen with the title 'Unlock Jenkins'. It instructs the user to ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server: `/var/jenkins_home/secrets/initialAdminPassword`. Below this, it says 'Please copy the password from either location and paste it below.' A redacted password field is shown under 'Administrator password'. At the bottom right is a blue 'Continue' button.



The screenshot shows the Jenkins 'Getting Started' screen with the title 'Getting Started'. It lists several Jenkins plugins with checkboxes:

<input checked="" type="checkbox"/> Folders Plugin	<input checked="" type="checkbox"/> OWASP Markup Formatter	<input type="checkbox"/> Build Timeout	<input type="checkbox"/> Credentials Binding Plugin
<input type="checkbox"/> Timestamper	<input type="checkbox"/> Workspace Cleanup	<input type="checkbox"/> Ant	<input type="checkbox"/> Gradle
<input type="checkbox"/> Pipeline	<input type="checkbox"/> GitHub Branch Source	<input checked="" type="checkbox"/> Pipeline: GitHub Groovy Libraries	<input type="checkbox"/> Pipeline: Stage View
<input type="checkbox"/> Git plugin	<input type="checkbox"/> SSH Build Agents	<input type="checkbox"/> Matrix Authorization Strategy	<input type="checkbox"/> PAM Authentication
<input type="checkbox"/> LDAP	<input type="checkbox"/> Email Extension	<input type="checkbox"/> Mailer Plugin	

A sidebar on the right lists the selected plugins: 'Folders' and 'OWASP Markup Formatter'. At the bottom right is a note: '** - required dependency'.



CI/CD con Jenkins para el Desarrollo Local

Instalar Jenkins en máquina local

Getting Started

Create First Admin User

Usuario

Contraseña

Confirma la contraseña

Nombre completo

Dirección de email

Jenkins 2.426.3

Skip and continue as admin

Getting Started

Instance Configuration

Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.426.3

Not now



CI/CD con Jenkins para el Desarrollo Local

Instalar Jenkins en máquina local

Getting Started

Jenkins is ready!

Your Jenkins setup is complete.

[Start using Jenkins](#)

The screenshot shows the Jenkins dashboard with the following elements:

- Header:** Jenkins logo, search bar, user info (José Enrique Mejía Gamarra), and navigation links.
- Left Sidebar:** Buttons for "Nueva Tarea", "Personas", "Historial de trabajos", "Administrar Jenkins", and "Mis vistas".
- Middle Section:**
 - Trabajos en la cola:** Message "No hay trabajos en la cola".
 - Estado del ejecutor de construcciones:** Shows 1 Inactivo and 2 Inactivo.
 - Bienvenido a Jenkins!** Message: "This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project."
 - Start building your software project** button.
 - Create a job** button.
 - Set up a distributed build** section with "Set up an agent", "Configure a cloud", and "Learn more about distributed builds".
- Bottom Right:** REST API and Jenkins 2.42 links.



CI/CD con Jenkins para el Desarrollo Local

Crear Job - Pipeline

Enter an item name

my-first-job
» Required field

 **Crear un proyecto de estilo libre**
Esta es la característica principal de Jenkins, la de ejecutar el proyecto combinando cualquier tipo de repositorio de software (SCM) con cualquier modo de construcción o ejecución (make, ant, mvn, rake, script ...). Por tanto se podrá tanto compilar y empaquetar software, como ejecutar cualquier proceso que requiera monitorización.

 **Pipeline**
Gestiona actividades de larga duración que pueden abarcar varios agentes de construcción. Apropiado para construir pipelines (conocidas anteriormente como workflows) y/o para la organización de actividades complejas que no se pueden articular fácilmente con tareas de tipo freestyle.

 **Crear un proyecto multi-getConfiguración**
Adequado para proyectos que requieran un gran número de configuraciones diferentes, como testear en múltiples entornos, ejecutar sobre plataformas concretas, etc.

 **Folder**
A container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

OK



CI/CD con Jenkins para el Desarrollo Local

Crear Job - Pipeline

Esta opción permite a Jenkins **administrar automáticamente el historial de ejecuciones** del pipeline, eliminando las más antiguas según ciertos criterios configurados.

Desechar ejecuciones antiguas ?

Strategy

Log Rotation

Número de días para mantener ejecuciones de proyectos
si no está vacío, sólo se mantendrán las ejecuciones con una edad inferior a este número de días

5

Número máximo de ejecuciones para guardar
si no está vacío, sólo se guardarán un número de ejecuciones inferior a este valor

5

Avanzado ▾

- Desechar ejecuciones antiguas ?
- Do not allow concurrent builds
- Do not allow the pipeline to resume if the controller restarts
- Esta ejecución debe parametrizarse ?
- GitHub project
- Pipeline speed/durability override ?
- Preserve stashes from completed builds ?
- Throttle builds ?



CI/CD con Jenkins para el Desarrollo Local

Crear Job - Pipeline

Si esta opción está activada, **previene que se ejecuten múltiples** instancias del mismo pipeline al mismo tiempo.

Impide que un pipeline que estaba en ejecución al momento de reiniciar el controlador (Jenkins) **se reanude automáticamente.**

Do not allow concurrent builds

Abort previous builds ?

Do not allow the pipeline to resume if the controller restarts

- Desechar ejecuciones antiguas ?
- Do not allow concurrent builds
- Do not allow the pipeline to resume if the controller restarts
- Esta ejecución debe parametrizarse ?
- GitHub project
- Pipeline speed/durability override ?
- Preserve stashes from completed builds ?
- Throttle builds ?



CI/CD con Jenkins para el Desarrollo Local

Crear Job - Pipeline

Permite **definir parámetros para el pipeline**, los cuales pueden ser utilizados en la ejecución para modificar el comportamiento del pipeline basado en valores de entrada proporcionados al iniciar la construcción.

Esta ejecución debe parametrizarse ?

≡ Parámetro de cadena ?

Defines a simple text parameter, where users can enter a string value, which you can use during a build, either as an environment variable, or through variable substitution in some other parts of the configuration.

Nombre ?
name_container

Valor por defecto ?
my-model-api-container

Descripción ?

Plain text [Visualizar](#)

Trim the string ?

- Desechar ejecuciones antiguas ?
- Do not allow concurrent builds
- Do not allow the pipeline to resume if the controller restarts
- Esta ejecución debe parametrizarse ?
- GitHub project
- Pipeline speed/durability override ?
- Preserve stashes from completed builds ?
- Throttle builds ?



CI/CD con Jenkins para el Desarrollo Local

Crear Job - Pipeline

Esta opción permite **asociar el pipeline con un proyecto** específico en GitHub

Permite ajustar la configuración de **durabilidad del pipeline, equilibrando entre el rendimiento y la capacidad de sobrevivir a reinicios inesperados.**

Esta configuración permite conservar los **stashes** creados durante una ejecución del pipeline para que puedan ser **usados en ejecuciones futuras**. Los stashes son **colecciones temporales de archivos** que pueden ser guardados y restaurados en diferentes etapas del pipeline.

Permite **limitar el número de ejecuciones del pipeline** durante un período de tiempo determinado o asegurar que solo un cierto número de ejecuciones puedan correr simultáneamente dentro de una categoría de pipelines.

- Desechar ejecuciones antiguas ?
- Do not allow concurrent builds
- Do not allow the pipeline to resume if the controller restarts
- Esta ejecución debe parametrizarse ?
- GitHub project
- Pipeline speed/durability override ?
- Preserve stashes from completed builds ?
- Throttle builds ?



CI/CD con Jenkins para el Desarrollo Local

Crear Job - Pipeline

Esta opción permite configurar el pipeline para que se **ejecute automáticamente después de que otro proyecto o pipeline específico se haya construido.**

Permite configurar el pipeline para que **se ejecute en intervalos regulares**, especificados mediante una sintaxis cron.

Ejecutar periódicamente ?

Programador ?

```
15 18 * * *
```

⚠ Spread load evenly by using 'H 18 * * *' rather than '15 18 * * *'
Would last have run at Saturday, February 3, 2024 at 6:15:05 PM Coordinated Universal Time; would next run at Sunday, February 4, 2024 at 6:15:05 PM Coordinated Universal Time.

Build Triggers

Construir tras otros proyectos ?

Ejecutar periódicamente ?

GitHub hook trigger for GITScm polling ?

Consultar repositorio (SCM) ?

Periodo de espera ?

Lanzar ejecuciones remotas (ejem: desde 'scripts') ?



CI/CD con Jenkins para el Desarrollo Local

Crear Job - Pipeline

Este disparador es utilizado para iniciar automáticamente el pipeline cuando se **detectan cambios en un repositorio de GitHub**.

Con esta opción, Jenkins **revisará periódicamente el repositorio de control de versiones** (SCM) para detectar cambios.

Este disparador le permite a Jenkins **esperar un cierto tiempo después de que se detectan cambios** en el SCM antes de iniciar la construcción.

Esta opción permite iniciar la construcción de Jenkins de **manera remota a través de una llamada HTTP GET o POST** a una URL específica de Jenkins.

Build Triggers

- Construir tras otros proyectos ?
- Ejecutar periódicamente ?
- GitHub hook trigger for GITScm polling ?
- Consultar repositorio (SCM) ?
- Periodo de espera ?
- Lanzar ejecuciones remotas (ejem: desde 'scripts') ?



CI/CD con Jenkins para el Desarrollo Local

Crear Job - Pipeline

Pipeline

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

https://github.com/EnriqueMejia96/Jenkins_CI-CD.git

Credentials ?

- none -

+ Add ▾

Avanzado ▾

The screenshot shows the Jenkins Pipeline creation interface. The 'Definition' dropdown is set to 'Pipeline script from SCM'. Under 'SCM', 'Git' is selected. In the 'Repositories' section, a single repository is configured with the URL 'https://github.com/EnriqueMejia96/Jenkins_CI-CD.git'. The 'Credentials' dropdown is currently empty ('- none -'). There is also a '+ Add' button for adding new credentials. At the bottom, there is an 'Avanzado' (Advanced) section with a dropdown menu.



CI/CD con Jenkins para el Desarrollo Local

Crear Job - Pipeline

Branches to build ?

Branch Specifier (blank for 'any') ?

main ×

Add Branch

Navegador del repositorio ?

(Auto) ▼

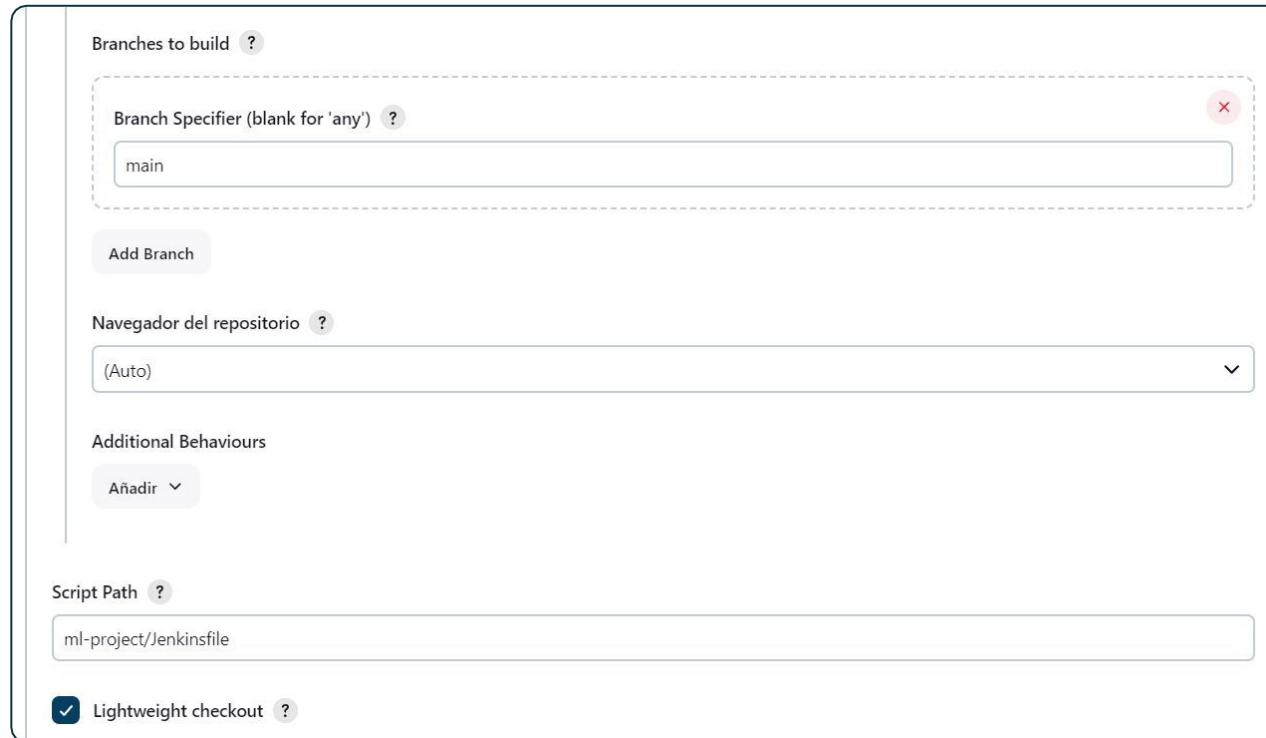
Additional Behaviours

Añadir ▼

Script Path ?

ml-project/Jenkinsfile

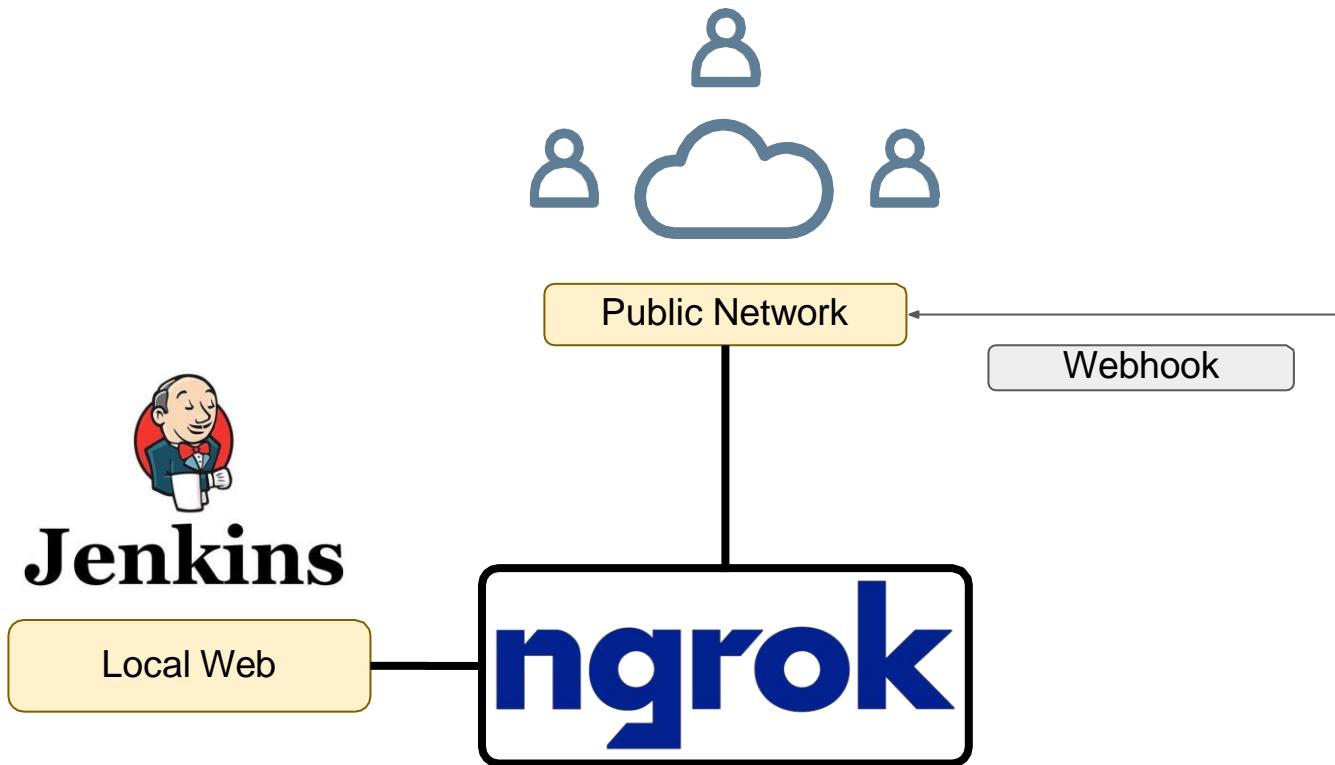
Lightweight checkout ?





CI/CD con Jenkins para el Desarrollo Local

Trigger: Webhook GitHub





CI/CD con Jenkins para el Desarrollo Local

Trigger: Webhook GitHub

```
ngrok http 8081
```

ngrok

Build better APIs with ngrok. Early access: ngrok.com/early-access

Session Status

Account

Version

Region

Latency

Web Interface

Forwarding

Connections

online

enriquemejia@gmail.com (Plan: Free)

3.5.0

United States (us)

172ms

<http://127.0.0.1:4040>

<https://f4eb-2800-4b0-4500-a037-a0ac-f65e-6f59-1d98.ngrok-free.app> -> <http://localhost:8081>

ttl	opn	rtt1	rtt5	p50	p90
0	0	0.00	0.00	0.00	0.00

Webhooks

Add webhook

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our [Webhooks Guide](#).

●

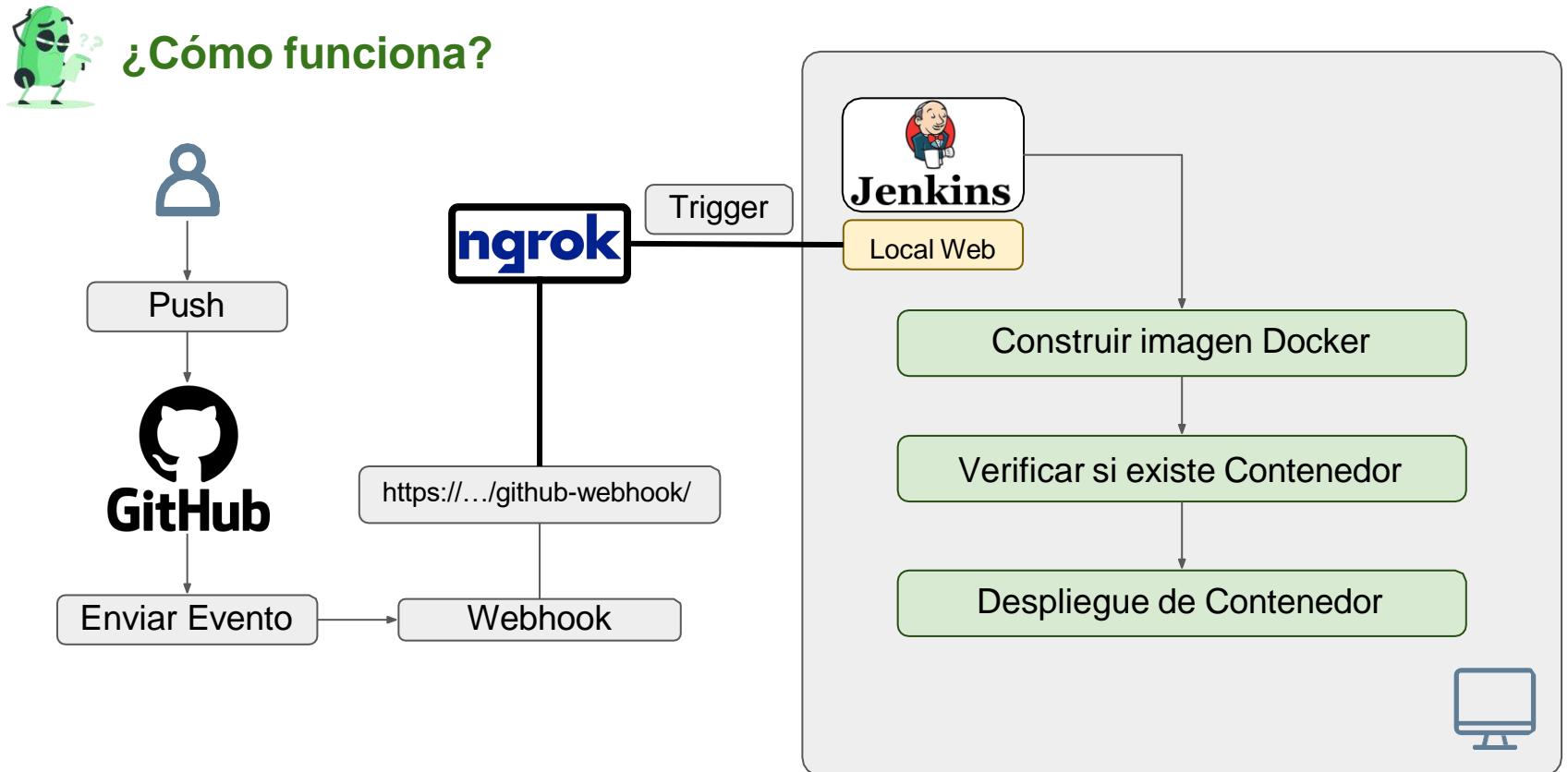
Edit

Delete

The screenshot shows the GitHub Settings interface with the 'Webhooks' section selected. On the left, a sidebar lists various settings like Environments, Codespaces, Pages, Security, Code security and analysis, Deploy keys, Secrets and variables, Integrations, and GitHub Apps. The main area is titled 'Webhooks / Add webhook'. It contains fields for 'Payload URL' (containing a placeholder URL), 'Content type' (set to 'application/json'), and 'Secret' (an empty field). Under 'SSL verification', the 'Enable SSL verification' option is selected. At the bottom, there's a section for selecting events with radio buttons for 'Just the push event.', 'Send me everything.', and 'Let me select individual events.'



CI/CD con Jenkins para el Desarrollo Local





CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 0: Guardar imagen en AWS ECR

0.0. Crear imagen local [Docker Desktop]

```
docker build -t my-predict-api:latest .
```

0.1. Iniciar sesión en AWS ECR

```
aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin  
533267339745.dkr.ecr.us-east-1.amazonaws.com
```

0.2. Crear repositorio en AWS ECR

```
aws ecr create-repository --repository-name my-predict-api --region us-east-1
```

0.3. Etiquetar imagen docker

```
docker tag my-predict-api:latest 533267339745.dkr.ecr.us-east-1.amazonaws.com/my-predict-api:latest
```

0.4. Hacer push a la imagen docker -> AWS ECR

```
docker push 533267339745.dkr.ecr.us-east-1.amazonaws.com/my-predict-api:latest
```



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 1: Crear entorno en AWS Elastic Beanstalk

Configure environment Info

Environment tier Info
Amazon Elastic Beanstalk has two types of environment tiers to support different types of web applications.

Web server environment
Run a website, web application, or web API that serves HTTP requests. [Learn more](#)

Worker environment
Run a worker application that processes long-running workloads on demand or performs tasks on a schedule. [Learn more](#)

Application information Info

Application name

Maximum length of 100 characters.

▶ Application tags (optional)

Environment information Info
Choose the name, subdomain and description for your environment. These cannot be changed later.

Environment name

Must be from 4 to 40 characters in length. The name can contain only letters, numbers, and hyphens. It can't start or end with a hyphen.
This name must be unique within a region in your account.

Domain
.us-east-1.elasticbeanstalk.com
 my-predict-api.us-east-1.elasticbeanstalk.com is available

Elastic Beanstalk x Elastic Beanstalk > Environments

Environments (1) Info

Actions Create environment

Platform Info

Platform type
 Managed platform
Platforms published and maintained by Amazon Elastic Beanstalk. [Learn more](#)

Custom platform
Platforms created and owned by you. This option is unavailable if you have no platforms.

Platform

Platform branch

Platform version

Application code Info

Sample application

Existing version
Application versions that you have uploaded.

Upload your code
Upload a source bundle from your computer or copy one from Amazon S3.

Version label
Unique name for this version of your application code.

Source code origin. Maximum size 500 MB

Local file



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 1: Crear entorno en AWS Elastic Beanstalk

The screenshot shows the AWS Elastic Beanstalk interface. On the left, there's a sidebar with 'Applications' and 'Environments' selected. The main area is titled 'Environments (1) Info'. It shows one environment named 'My Environment'. At the top right, there are buttons for 'Actions', 'Create environment', and navigation arrows.

This screenshot shows the 'Configure service access' step of the AWS Elastic Beanstalk wizard. It includes sections for 'Service access', 'Service role', 'Existing service roles', 'EC2 key pair', 'EC2 instance profile', and 'Environment properties'. A red box highlights the 'Skip to review' button at the bottom.

Presets [Info](#)
Start from a preset that matches your use case or choose custom configuration to unset recommended values and use the service's default values.

Configuration presets
 Single instance (free tier eligible)
 Single instance (using spot instance)
 High availability
 High availability (using spot and on-demand instances)
 Custom configuration

Next

Configure service access [Info](#)

Service access
IAM roles, assumed by Elastic Beanstalk as a service role, and EC2 instance profiles allow Elastic Beanstalk to create and manage your environment. Both the IAM role and instance profile must be attached to IAM managed policies that contain the required permissions. [Learn more](#)

Service role
 Create and use new service role
 Use an existing service role
Existing service roles
Choose an existing IAM role for Elastic Beanstalk to assume as a service role. The existing IAM role must have the required IAM managed policies.
aws-elasticbeanstalk-service-role [Change](#)

EC2 key pair
Select an EC2 key pair to securely log in to your EC2 instances. [Learn more](#)
Choose a key pair [Change](#)

EC2 instance profile
Choose an IAM instance profile with managed policies that allow your EC2 instances to perform required operations.
aws-elasticbeanstalk-ec2-role [Change](#)
[View permission details](#)

Environment properties
X-Ray enabled: Deactivated
No environment properties defined

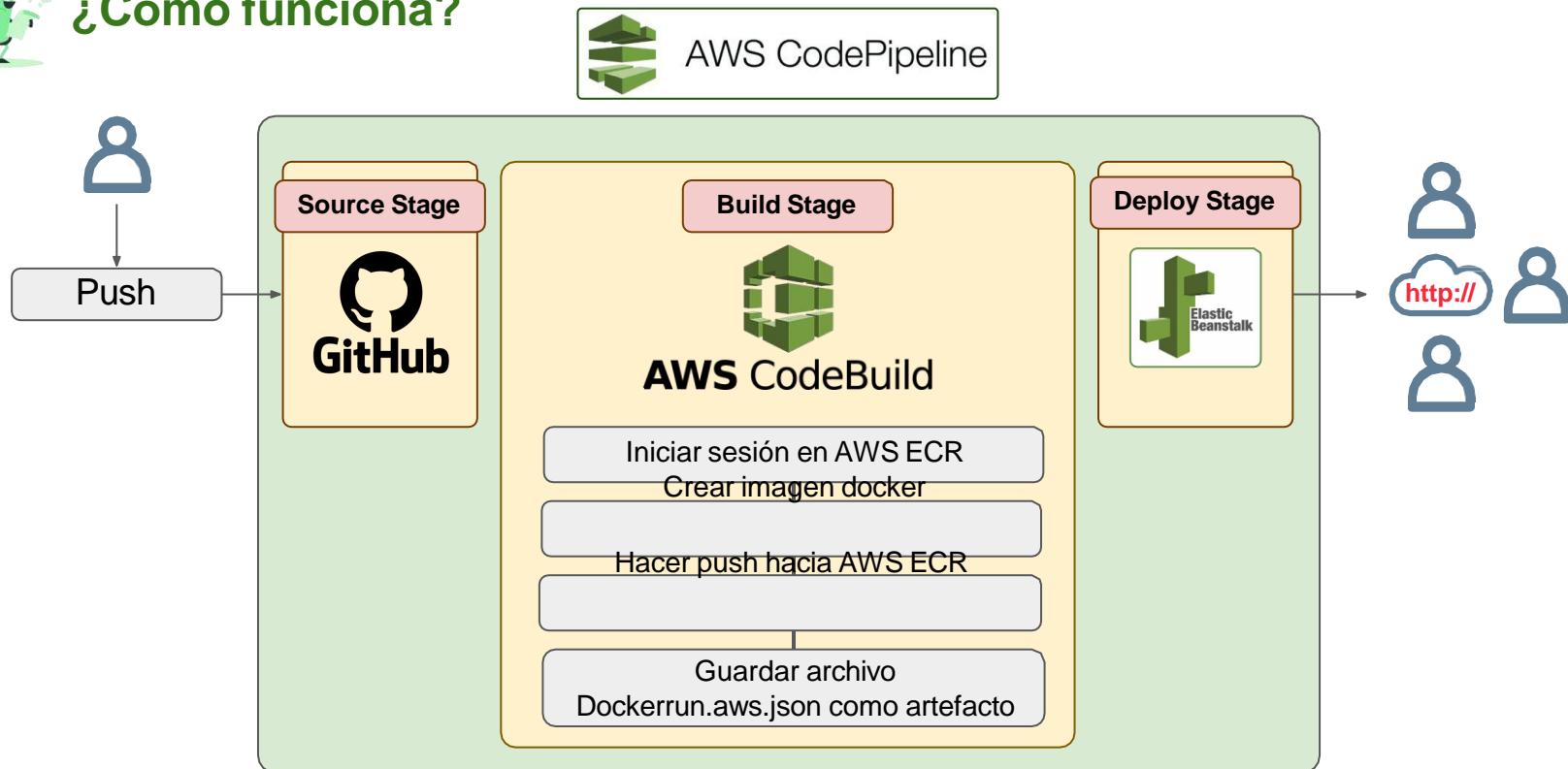
Next



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk



¿Cómo funciona?





CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 2: Crear pipeline en AWS CodePipeline

Choose pipeline settings Info

Step 1 of 5

Pipeline settings

Pipeline name

Enter the pipeline name. You cannot edit the pipeline name after it is created.

my-predict-api

No more than 100 characters.

Pipeline type

The pipeline type determines the pipeline structure and availability of parameters such as triggers. Pipeline type selection will impact features and pricing. [Which pipeline is right for me?](#)

V1

V2

Service role

New service role

Create a service role in your account

Existing service role

Choose an existing service role from your account

Role name

AWSCodePipelineServiceRole-us-east-1-my-predict-api

Type your service role name

Allow AWS CodePipeline to create a service role so it can be used with this new pipeline

AWS CodePipeline ofrece diferentes versiones (V1, V2)

V1: la versión original de CodePipeline, que puede carecer de algunas funciones más nuevas, pero es familiar y **estable para las implementaciones existentes**.

V2: una versión más nueva que podría ofrecer **capacidades mejoradas**, rendimiento mejorado o un modelo de **precios más flexible**. Sin embargo, las funciones específicas disponibles en V1 pueden ser diferentes o no estar disponibles en V2.

Define el **rol de IAM** (Administración de identidad y acceso) que asume CodePipeline al **acceder a los recursos de AWS** en su nombre. Este rol necesita permisos suficientes para acceder a los recursos utilizados en su canalización



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 3: AWS CodePipeline - Add Source Stage

Add source stage [Info](#)

Step 2 of 5

Source

Source provider

This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

GitHub (Version 2)

New GitHub version 2 (app-based) action
To add a GitHub version 2 action in CodePipeline, you create a connection, which uses GitHub Apps to access your repository. Use the options below to choose an existing connection or create a new one. [Learn more](#)

Connection

Choose an existing connection that you have already configured, or create a new one and then return to this task.

or [Connect to GitHub](#)

Repository name

Choose a repository in your GitHub account.

You can type or paste the group path to any project that the provided credentials can access. Use the format 'group/subgroup/project'.

Pipeline trigger

Source action can either use change detection (branch and commit) or a pipeline trigger configuration, such as Git tags, but not both. Note: Choosing a trigger that is not the branch option will not enable automated change detection for the pipeline.

Push in a branch
 Git tags
Pipeline type V2 required

None
Your pipeline only runs if you start it manually or on a schedule.

AWS | Services | [Developer Tools](#) > ... > Create connection

Create a connection [Info](#)

Create GitHub App connection [Info](#)

Connection name

github-jemg-connection

▶ Tags - optional

[Connect to GitHub](#)

AWS Connector for GitHub by [Amazon Web Services](#) would like permission to:

Verify your GitHub identity (EnriqueMejia96)
Know which resources you can access
Act on your behalf

[Learn more](#)

Learn more about AWS Connector for GitHub

[Cancel](#) [Authorize AWS Connector for GitHub](#)

Authorizing will redirect to <https://redirect.codestar.aws>

Not owned or operated by GitHub
 Created 4 years ago
 More than 1K GitHub users



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 3: AWS CodePipeline - Add Source Stage

Connect to GitHub

GitHub connection settings [Info](#)

Connection name
github-jemg-connection

GitHub Apps
GitHub Apps create a link for your connection with GitHub. To start, install a new app and save this connection.

or [Install a new app](#)

▶ Tags - optional

[Connect](#)

Install AWS Connector for GitHub

Install on your personal account José Enrique Mejía Gamarra

for these repositories:

All repositories
This applies to all current *and* future repositories owned by the resource owner.
Also includes public repositories (read-only).

Only select repositories
Select at least one repository.
Also includes public repositories (read-only).

with these permissions:

- Read access to issues and metadata
- Read and write access to administration, code, commit statuses, and pull requests

[Install](#) [Cancel](#)

Confirm access

Signed in as @EnriqueMejia96

Authentication code [?](#)

XXXXX [Verify](#)

Open your two-factor authenticator (TOTP) app or browser extension to view your authentication code.

Having problems?
[Use your password](#)



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 3: AWS CodePipeline - Add Source Stage

Connection
Choose an existing connection that you have already configured, or create a new one and then return to this task.

or [Connect to GitHub](#)

Ready to connect
Your GitHub connection is ready for use.

Repository name
Choose a repository in your GitHub account.

[X](#)

You can type or paste the group path to any project that the provided credentials can access. Use the format 'group/subgroup/project'.

Pipeline trigger
Source action can either use change detection (branch and commit) or a pipeline trigger configuration, such as Git tags, but not both. Note: Choosing a trigger that is not the branch option will not enable automated change detection for the pipeline.

Push in a branch
 Git tags
Pipeline type V2 required
 None
Your pipeline only runs if you start it manually or on a schedule.

Branch name
Choose a branch of the repository.

[X](#)

Output artifact format
Choose the output artifact format.

CodePipeline default
AWS CodePipeline uses the default zip format for artifacts in the pipeline. Does not include Git metadata about the repository.

Full clone
AWS CodePipeline passes metadata about the repository that allows subsequent actions to do a full Git clone. Only supported for AWS CodeBuild actions.

[Cancel](#) [Previous](#) [Next](#)



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 3: AWS CodePipeline - Add Build Stage

Build - optional

Build provider

This is the tool of your build project. Provide build artifact details like operating system, build spec file, and output file names.

AWS CodeBuild

Region

US East (N. Virginia)

Project name

Choose a build project that you have already created in the AWS CodeBuild console. Or create a build project in the AWS CodeBuild console and then return to this task.



or

Create project

Environment variables - optional

Choose the key, value, and type for your CodeBuild environment variables. In the value field, you can reference variables generated by CodePipeline. [Learn more](#)

Add environment variable

Build type

Single build

Triggers a single build.

Batch build

Triggers multiple builds as a single execution.

Create build project

Project configuration

Project name

my-predict-api-codebuild

A project name must be 2 to 255 characters. It can include the letters A-Z and a-z, the numbers 0-9, and the special characters - and _.

Additional configuration

Description, Build badge, Concurrent build limit, tags



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 3: AWS CodePipeline - Add Build Stage

Environment

Provisioning model info

On-demand
Automatically provision build infrastructure in response to new builds.

Reserved capacity
Use a dedicated fleet of instances for builds. Fleet's compute and environment type will be used for the project.

Environment image

Managed image
Use an image managed by AWS CodeBuild

Custom image
Specify a Docker image

Compute

EC2
Optimized for flexibility during action runs

Lambda
Optimized for speed and minimizes the start up time of workflow actions.

Operating system Amazon Linux

Runtime(s) Standard

Image aws/codebuild/amazonlinux2-x86_64-standard:5.0

Image version Always use the latest image for this runtime version

Use GPU-enhanced compute

Service role

New service role
Create a service role in your account

Existing service role
Choose an existing service role from your account

Role name codebuild-my-predict-api-role
Type your service role name

Additional configuration
Timeout, certificate, VPC, compute type, environment variables, file systems

Buildspec

Build specifications

Use a buildspec file
Store build commands in a YAML-formatted buildspec file

Insert build commands
Store build commands as build project configuration

Buildspec name - optional
By default, CodeBuild looks for a file named buildspec.yml in the source code root directory. If your buildspec file uses a different name or location, enter its path from the source root here (for example, buildspec-two.yml or configuration/buildspec.yml).

buildspec.yml



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 3: AWS CodePipeline - Add Deploy Stage

Add deploy stage Info

Step 4 of 5

Deploy - optional

Deploy provider
Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.

AWS Elastic Beanstalk ▾

Region

US East (N. Virginia) ▾

Application name
Choose an application that you have already created in the AWS Elastic Beanstalk console. Or create an application in the AWS Elastic Beanstalk console and then return to this task.

X

Environment name
Choose an environment that you have already created in the AWS Elastic Beanstalk console. Or create an environment in the AWS Elastic Beanstalk console and then return to this task.

X

[Cancel](#) [Previous](#) [Skip deploy stage](#) **Next**



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 4: Agregar permisos ECR a Service Role - IAM

Identity and Access Management (IAM)

Search IAM

Dashboard

Access management

- User groups
- Users
- Roles**
- Policies
- Identity providers
- Account settings

Access reports

- Access Analyzer
- External access
- Unused access
- Analyzer settings
- Credential report
- Organization activity

IAM > Roles

Roles (10) Info

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Role name	Trusted entities	Last activity
aws-elasticbeanstalk-ec2-role	AWS Service: ec2	18 minutes ago
aws-elasticbeanstalk-service-role	AWS Service: elasticbeanstalk	22 minutes ago
AWSCodePipelineServiceRole-us-east-1-my-predict-api	AWS Service: codepipeline	-
AWSCodePipelineServiceRole-us-east-1-my-predict-api-pipeline	AWS Service: codepipeline	1 hour ago
AWSServiceRoleForAutoScaling	AWS Service: autoscaling (Service-Linked Role)	34 minutes ago
AWSServiceRoleForEC2Spot	AWS Service: spot (Service-Linked Role)	-
AWSServiceRoleForElasticLoadBalancing	AWS Service: elasticloadbalancing (Service-Linked Role)	3 days ago
AWSServiceRoleForSupport	AWS Service: support (Service-Linked Role)	-
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service-Linked Role)	-
codebuild-my-predict-api-role	AWS Service: codebuild	-



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 4: Agregar permisos ECR a Service Role - IAM

codebuild-my-predict-api-role [Info](#)

Summary

Creation date
February 04, 2024, 18:19 (UTC-05:00)

Last activity
-

ARN
[arn:aws:iam::533267339745:role/service-role/codebuild-my-predict-api-role](#)

Maximum session duration
1 hour

Permissions [Trust relationships](#) [Tags](#) [Access Advisor](#) [Revoke sessions](#)

Permissions policies (1) [Info](#)
You can attach up to 10 managed policies.

Filter by Type [All types](#)

Policy name [CodeBuildBasePolicy-my-predict-api-code-build-us-east-1](#)

Type Customer managed

Creation time February 04, 2024, 18:19 (UTC-05:00)

Edited time February 04, 2024, 18:19 (UTC-05:00)

ARN [arn:aws:iam::533267339745:policy/service-role/CodeBuildBasePolicy-my-predict-api-code-build-us-east-1](#)

CodeBuildBasePolicy-my-predict-api-code-build-us-east-1 [Info](#)
Policy used in trust relationship with CodeBuild

Policy details

Type Customer managed

Creation time February 04, 2024, 18:19 (UTC-05:00)

Edited time February 04, 2024, 18:19 (UTC-05:00)

ARN [arn:aws:iam::533267339745:policy/service-role/CodeBuildBasePolicy-my-predict-api-code-build-us-east-1](#)

Permissions [Entities attached](#) [Tags](#) [Policy versions \(1\)](#) [Access Advisor](#)

Permissions defined in this policy [Info](#)
Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it.

Search

Allow (3 of 403 services)

Service	Access level	Resource	Request condition
CloudWatch Logs	Limited: Write	Multiple	None
CodeBuild	Limited: Write	region string like us-east-1	None
S3	Limited: Read, Write	BucketName string like [codepipeline-us-east-1-*	None

[Edit](#) [Summary](#) [JSON](#)

Show remaining 400 services



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 4: Agregar permisos ECR a Service Role - IAM

Policy editor

```
16  {
17      "Effect": "Allow",
18      "Resource": [
19          "arn:aws:s3:::codepipeline-us-east-1-*"
20      ],
21      "Action": [
22          "s3:PutObject",
23          "s3:GetObject",
24          "s3:GetObjectVersion",
25          "s3:GetBucketAcl",
26          "s3:GetBucketLocation"
27      ]
28 },
29 {
30     "Effect": "Allow",
31     "Action": [
32         "codebuild:CreateReportGroup",
33         "codebuild:CreateReport",
34         "codebuild:UpdateReport",
35         "codebuild:BatchPutTestCases",
36         "codebuild:BatchPutCodeCovverages"
37     ],
38     "Resource": [
39         "arn:aws:codebuild:us-east-1:533267339745:report-group/my-predict-api-code-build-*"
40     ]
41 },
42 {
43     "Sid": "Statement1",
44 }
```

+ Add new statement

Visual JSON Actions ▾

Edit statement Statement1 Remove

Add actions Choose a service CONTAINER

Available App2Container EMR Containers

Elastic Container Registry

Elastic Container Registry Public

Elastic Container Service

Add a resource Add

Add a condition (optional) Add

5307 of 6144 characters remaining

Check for new access

JSON Ln 43, Col 21

Security: 0 Errors: 0 Warnings: 0 Suggestions: 2

Edit statement Statement1 Remove

Add actions All services > Elastic Container Registry

Filter actions

All actions (ecr:*)

Access level - list

DescribeImages Info

DescribePullThroughCacheRules Info

ListImages Info

Access level - read

BatchCheckLayerAvailability Info

BatchGetImage Info

BatchGetRepositoryScanningConfiguration

Add a resource Add

Add a condition (optional) Add



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 4: Agregar permisos ECR a Service Role - IAM

Edit statement Statement1 Remove

Add actions All services > Elastic Container Registry

Filter actions

All actions (ecr:*)

Access level - list

DescribeImages Info

DescribePullThroughCacheRules Info

ListImages Info

Access level - read

BatchCheckLayerAvailability Info

BatchGetImage Info

BatchGetRepositoryScanningConfiguration

Add a resource

Add a condition (optional)

5300 of 6144 characters remaining

Add resource X

Specify the resource type and ARN to add for the selected service.

Service: Elastic Container Registry

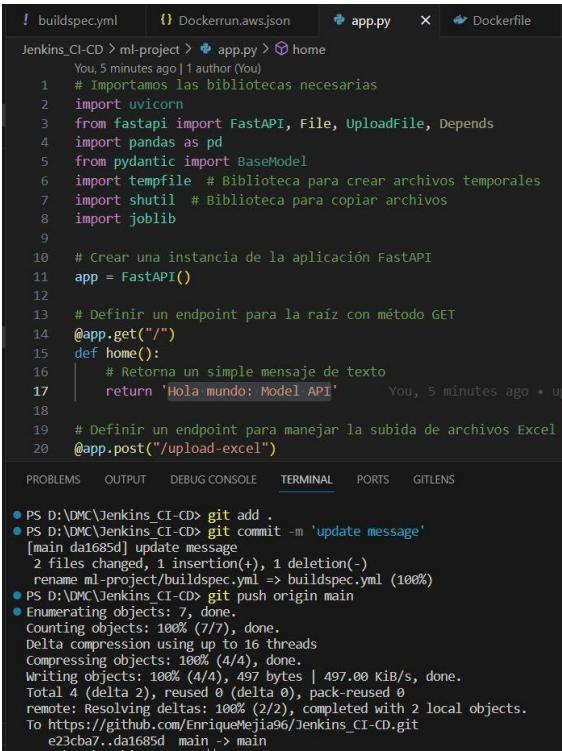
Resource type: All Resources

Resource ARN:



CI/CD con AWS CodePipeline + AWS Elastic Beanstalk

Paso 5: Probar Pipeline - CI/CD



```
! buildspec.yml Dockerrun.aws.json app.py Dockerfile
Jenkins_CI-CD > ml-project > app.py > home
You, 5 minutes ago | 1 author (You)
1 # Importamos las bibliotecas necesarias
2 import uvicorn
3 from fastapi import FastAPI, File, UploadFile, Depends
4 import pandas as pd
5 from pydantic import BaseModel
6 import tempfile # Biblioteca para crear archivos temporales
7 import shutil # Biblioteca para copiar archivos
8 import joblib
9
10 # Crean una instancia de la aplicación FastAPI
11 app = FastAPI()
12
13 # Definir un endpoint para la raíz con método GET
14 @app.get("/")
15 def home():
16     # Retorna un simple mensaje de texto
17     return "Hola mundo: Model API"
18
19 # Definir un endpoint para manejar la subida de archivos Excel
20 @app.post("/upload-excel")
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
PS D:\DMC\jenkins_CI-CD> git add .
PS D:\DMC\jenkins_CI-CD> git commit -m 'update message'
[main da1685d] update message
 2 files changed, 1 insertion(+), 1 deletion(-)
 rename ml-project/buildspec.yml => buildspec.yml (100%)
PS D:\DMC\jenkins_CI-CD> git push origin main
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 16 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 497 bytes | 497.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/EnriqueMejia96/Jenkins_CI-CD.git
 e23cba7..da1685d main -> main
```

