DESAFÍO 13 - PIPELINES

OBJETIVO

Construir un pipeline que se encargue del proceso de automatización de la construcción del *DockerFile*.

El pipeline se debe encargar de tomar el código de la aplicación, construir una imagen, levantar un contenedor y subir esa imagen a *DockerHub*.

JENKINS

1. Utilizar la sintaxis de tipo script o *JenkinsFile* (este archivo debe estar en un repositorio público, por ejemplo, G*itHub*).

2. El pipeline debe tener lo siguiente:

- Construcción de la imagen.
- Ejecución de un contenedor usando esa imagen.
- Publicar esa imagen en un repositorio de imágenes (*DockerHub*, por ejemplo).

3. Componentes adicionales, seleccionar al menos dos:

- Linting/Eficiencia/Seguridad: Usar herramientas como Hadolint, Dive u otras para asegurar de que la imagen siga buenas prácticas, para ello, agregar un paso en el pipeline que ejecute esta verificación.
- Testing: Algún tipo de *test* que verifique el correcto funcionamiento de la aplicación, no es necesario que sea muy complejo.
- Trigger: Como mínimo indispensable el pipeline se ejecuta de forma manual utilizando el botón de *build now*. Agregar una ejecución periódica o vía *webhook*.
- Plugins: Configurar algún complemento que permita realizar cambios en el código como *trigger*.

NOTA: Todo lo que sea relacionado a la administración de credenciales, secretos, etc., se debe gestionar de una forma **segura,** en vez de utilizar los valores directamente en texto plano en el pipeline, caso contrario se restará puntos.

GITHUB ACTIONS U OTROS

1. Utilizar *Github Actions* u otra herramienta como, por ejemplo, las propuestas por *AWS*, G*itLab*, etc.

2. Construir el pipeline según los mismos requisitos (punto 2 y 3) del pipeline de Jenkins.

3. Configurar la lógica para la ejecución del pipeline, por ejemplo, *push* a *main* u otro tipo de *trigger*.

GITHUB

1. Publicar todos los archivos creados en su repositorio personal.

Es importante que se trabaje el *readme* del repositorio (incluso pueden agregar un *readme* por carpeta con más información para probar cada parte de este).

El objetivo es que una persona pueda visualizar su repositorio y testear de punta a punta (es decir, probar todo en conjunto y también cada una de las partes por separado).

MODALIDAD DE TRABAJO

En el archivo, documentar lo siguiente:

- Archivos utilizados en cada una de las herramientas.
- Comandos.
- Capturas de pantalla que responde la documentación.
- Problemas que se presentaron, pasos a seguir para encontrar la solución, etc.

Además, el documento debe tener una portada, datos personales y titulo del desafío.

ENTREGABLE

Los documentos son almacenados en la carpeta compartida que tienen en *Google Drive* con el formato:

<carpeta con su nombre>/<Fase>/<desafío>/archivo.

Por ejemplo, el instructivo se debe almacenar en la carpeta compartida con el nombre del alumno, en una carpeta llamada Fase 3, dentro debe tener otra carpeta llamada Desafío 13 y, por último, almacenar dentro de ella todos los archivos relevantes a este desafío.

Se esperan los siguientes archivos:

- Instructivo.
- Enlace del repositorio de código donde publican los archivos creados (dentro del instructivo).
- Enlace al repositorio de la imagen (dentro del instructivo).
- Archivos adicionales.

Recuerden seguir las instrucciones al pie de la letra para los entregables.

CONSEJOS

- A la hora de lanzar el contenedor, si definen un nombre y puerto donde se ejecute el contenedor, y si ya se encuentra uno corriendo con el mismo nombre o utilizando el mismo contenedor, este fallara. Es importante agregar una lógica al pipeline que, si ya hay un contenedor corriendo con ese nombre, lo detenga y lo elimine antes de crear el nuevo.

- Pueden agregar una segunda máquina virtual usando *Vagrant,* que actúe como *agent* para no correr los *pipelines* en el *Jenkins Máster*.