Planificación del Pipeline de Integración y Entrega Continua (CI/CD)

El propósito del pipeline CI / CD es automatizar la integración de código y las pruebas y despliegue del software para asegurar que cada modificación se valide y despliegue de forma segura y eficiente. En este proyecto, se empleará Jenkins (funcionando en contenedores Docker) junto a GitHub para coordinar todas las fases del proceso, facilitando iteraciones más rápidas y retroalimentación inmediata.

GitHub:
Un lugar centralizado donde se almacena el código fuente y se activan los procesos en Jenkins con cada commit o pull request a través de webhooks.

Jenkins en Contenedor de almacenamiento de datos.
Servidor de integración continua que se encarga de ejecutar el proceso de desarrollo continuo, utilizando contenedores Docker para Jenkins para garantizar la portabilidad y coherencia en diferentes entornos.

Docker:
Vamos a crear las imágenes de la aplicación que se usarán en entornos de prueba y producción para asegurarnos de que la aplicación funcione de la misma forma en todas las situaciones.

Flujo del Pipeline CI/CD

Commit y Notificación
Los programadores hacen envíos en ramas del repositorio de GitHub.
Un webhook avisa a Jenkins sobre las modificaciones en el código.

Etapa de Construcción
Clonación del repositorio por Jenkins y compilación del proyecto en Java.

Etapa de pruebas:
Pruebas de unidad e integración realizadas para validar la funcionalidad del código.
Si alguna prueba no tiene éxito durante la validación del sistema de procesamiento de datos y su ejecución se detiene automáticamente para informar a los desarrolladores involucrados en el proyecto.

Packaging y Contenedorización:
Los objetos se colocan dentro de un contenedor Docker.
La imagen es creada y guardada en un registro de contenedores (por ejemplo, Docker Hub o un repositorio privado).

Despliegue en entorno de pruebas (Staging):
La configuración de Docker se implementa en un entorno de prueba.
Se llevan a cabo pruebas adicionales, ya sea de forma automatizada o manual, para verificar que la aplicación funcione completamente correctamente.

Implementación en Producción:
Una vez que las pruebas en staging han sido aprobadas satisfactoriamente, se procede a implementar la imagen en el entorno de producción.

Notificaciones y Monitoreo
Jenkins envía alertas sobre el progreso del flujo de trabajo (ya sea exitoso o fallido), utilizando correos electrónicos o integraciones en herramientas de mensajería como Slack.
Se establecen avisos y sistemas de vigilancia para controlar el desempeño y la estabilidad después de la implementación.

Beneficios de la canalización
Automatización total : Reduce los errores humanos y agiliza el tiempo de entrega.
Retroalimentación instantánea facilita la detección y solución de problemas en las primeras etapas del desarrollo.
Consistencia en diferentes entornos de aplicación es asegurada por Docker para que la app funcione de igual manera durante desarrollo y pruebas como en producción.
Colaboración Efectiva : La conexión a GitHub simplifica el proceso de revisión y unión de modificaciones, lo que optimiza la coordinación entre los equipos.