INSTITUTO DE TECNOLOGIA

Trabajo Autónomo: Herramientas DevOps

-Repositorio de código: Git-

Carrera: Analista de Sistemas
Asignatura: Arquitectura y sistemas operativos
Docente: Gustavo Alessandrini
Curso: 1° 2° A-BE
Integrantes:

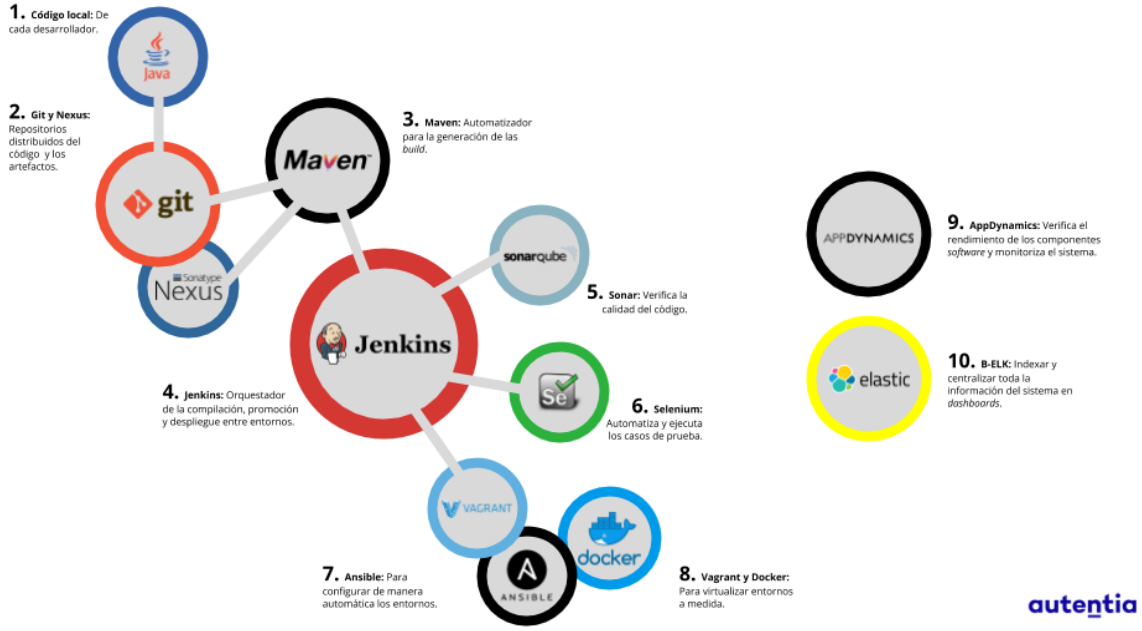
- /////

Fecha de entrega: 11 de junio de 2021
Año calendario: 2021

¿Qué es DevOps?

DevOps es un conjunto de prácticas de desarrollo que combina la integración del desarrollo de software (Dev) y las operaciones de TI (Ops) con la automatización. DevOps rompe las barreras entre los equipos de Dev y Ops para acortar el tiempo que lleva construir, probar y lanzar software de alta calidad. La finalidad de esta metodología es reducir el tiempo y esfuerzo en cada uno de los pasos, consiguiendo entregar código en producción con mayor rapidez y calidad, reduciendo los errores y limitando las tareas manuales que no aporten valor al proceso.

Flujo DevOps:



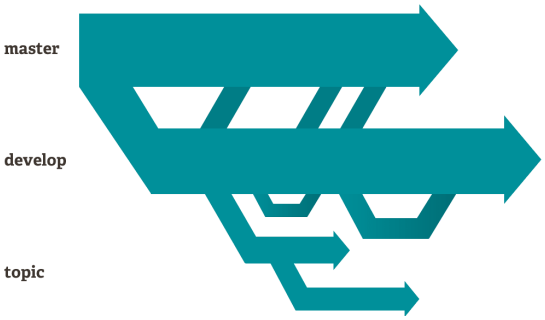
Lo primero con lo que debemos contar es con un repositorio de código distribuido(Git) en el cual se aloje el código de los diferentes desarrolladores de forma centralizada, logrando que el equipo de desarrollo pueda trabajar de forma colaborativa. Este es el primer paso hacia la integración continua. Los beneficios son la posibilidad de versionar código, asegurarse de que los desarrolladores construyen su código sobre la misma versión, y además se simplifica la gestión de cambios en el proyecto, ya que todo está etiquetado y es fácilmente trazable.

Continuando con el ciclo de DevOps se debe testear este código escrito y en caso de que las pruebas sean satisfactorias, se preparía el lanzamiento a productivo del mismo. Aquí es donde el área de Ops toma mayor relevancia ya que se procede a “Deployear” las nuevas funcionalidades, que serán monitoreadas constantemente para poder entregar una devolución al área de desarrollo.

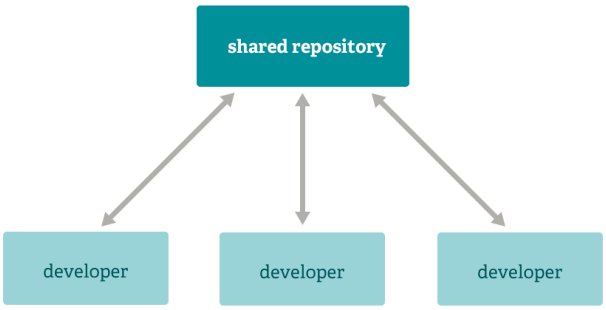
Esta es una de las mayores ventajas de DevOps, es un área interdisciplinaria que permite una gran colaboración entre sus integrantes, esto nos otorga mejoras continuas que se traducen en más y mejores entregas de cara al cliente.

¿Qué es Git?

Git es un sistema de control de versiones de código abierto que permite guardar y mantener un registro de todos los cambios que se realizan en el proyecto de software. Lo que lo diferencia del control de versiones tradicional es que se puede trabajar en diferentes versiones de la rama del software y gestionarlas todas al mismo tiempo.



Con Git, esto ocurre localmente en tu ordenador. Pero también puedes solicitar y dar acceso a otros desarrolladores de forma manual, por ejemplo, a través de una LAN.



Cuando se trabaja con equipos remotos o distribuidos más grandes, la mayoría de las empresas recurren a soluciones basadas en la nube, como lo son GitLab y GitHub.

Estas soluciones ofrecen almacenamiento ilimitado en la nube para los repositorios, y así los desarrolladores pueden colaborar fácilmente en el código.

Pero los repositorios Git basados en la nube, como GitHub y GitLab, también incluyen herramientas de gestión de proyectos, colaboración y despliegue para ayudar a mejorar el flujo de trabajo de desarrollo.

¿Qué es GitLab?

GitLab es una plataforma Git y DevOps basada en la nube que ayuda a los desarrolladores a supervisar, probar y desplegar su código. Ofrece una amplia gama de características DevOps, como la integración continua, la seguridad e incluso herramientas de despliegue de aplicaciones.

También ofrece herramientas esenciales de gestión de proyectos para supervisar y controlar a los miembros del equipo.

¿Qué es GitHub?

GitHub es una de las plataformas Git originales basadas en la nube que permite a los desarrolladores alojar y supervisar sus cambios de código. Ofrece a los desarrolladores la opción de implementar aplicaciones e integraciones libremente a través del mercado de GitHub. Su objetivo es que la misma comunidad contribuya en las funcionalidades que les falta.

Similitudes

- *Control de Versiones y Funcionalidades de Git*

La funcionalidad de Git y los comandos básicos siguen siendo en su mayoría los mismos entre las dos plataformas. Ambas plataformas ofrecen una amplia gama de herramientas de colaboración, revisión de código y gestión de proyectos.

- *Colaboración, Código y Gestión de Proyectos*

Ambas plataformas incluyen herramientas integradas de colaboración y revisión de código. Eso forma parte del kit esencial incluido en ambas versiones gratuitas. Estas herramientas resaltan todos los cambios y facilitan la labor de quien revisa el código.

- *Páginas de GitHub vs. Páginas de GitLab*

Tanto GitHub como GitLab ofrecen páginas web estáticas gratuitas con información sobre tu proyecto de software y tu repositorio. Ni GitHub ni GitLab ofrecen un procesamiento dinámico del lado del servidor, como el que necesitarías con archivos PHP o ASP. Solo puedes publicar contenido estático del sitio web como HTML y CSS.

- *Plan Gratuito con Repositorios Ilimitados*

Ambos servicios ofrecen planes gratuitos con repositorios ilimitados (públicos y privados). También limitan el acceso a las funciones avanzadas de gestión, seguridad y cumplimiento en sus opciones gratuitas.

Diferencias

- *Integración Continua*

Tal vez la diferencia más significativa en la experiencia del usuario sea la dedicación de GitLab a la integración continua y al flujo de trabajo DevOps.

Las herramientas de IC de GitLab te permiten construir, preparar y desplegar el código automáticamente sin tener que depender de las actualizaciones manuales o de las engorrosas integraciones personalizadas.

- *GitHub Flow vs GitLab Flow*

Aunque la tecnología y las características subyacentes de Git son idénticas, el flujo de trabajo recomendado no es el mismo. GitHub hace hincapié en la velocidad, mientras que GitLab en la fiabilidad. Esa es la esencia de la diferencia entre las dos plataformas de repositorios en la nube.

GitHub aboga por un enfoque de desarrollo rápido y centrado en las características para fusionar (incluir) nuevas ramas con la rama maestra. Este flujo de trabajo es perfecto para equipos y proyectos ágiles pequeños. La rama maestra siempre está lista para ser

desplegada, lo que garantiza que puedas restablecer rápidamente el status *quo* si algo va mal. Puedes volver a la versión anterior en cuestión de segundos.

En el flujo de trabajo de GitLab, se crean múltiples ramas estables más allá de la maestra, normalmente al menos producción y preproducción. Eso significa un proceso de pruebas de múltiples pasos en el que una sola revisión del código tras la solicitud de fusión no es suficiente.

- *Plataforma Completa vs Mercado*

GitLab adoptó el enfoque de una plataforma completa y empaquetada, en lugar de dar a cada uno la opción de construir su plataforma con diferentes aplicaciones. No obstante, GitLab admite integraciones con más 30 aplicaciones y plataformas.

En cambio GitHub tiene la posibilidad de realizar integraciones con más de 300 aplicaciones, que posee en su marketplace. Y cabe recordar que es a través de estas integraciones de terceros que GitHub ofrece *algunas* de las características que vienen por defecto en GitLab.

- *Instalación de Auto Alojamiento para un Servidor Privado*

GitLab comenzó como una plataforma de código abierto auto alojada. Dicho esto, todavía tienes la opción de alojar una versión privada de GitLab en tus máquinas virtuales.
Y puedes hacerlo con un plan gratuito de GitLab, mientras que con GitHub solo está disponible para los planes empresariales de GitHub.

- *Código Abierto*

GitLab es en sí mismo un software de código abierto, y la versión auto alojada es de uso gratuito para cualquiera.

Esto no quiere decir que GitHub no sea una buena opción para el código abierto, ya que ofrece acceso instantáneo a la mayoría de estos desarrolladores. Los equipos de desarrolladores de GitHub también contribuyen a proyectos de código abierto, concretamente a Git LFS y otros relacionados con Git.

Conclusiones

Hoy en día la gran mayoría de equipos, por no decir todos, utilizan Git, ya sea con GitHub, Gitlab, etc. Y si bien, no solo se utiliza en la práctica de DevOps, mantiene uno de sus grandes estandartes, la colaboración para una mejora contínua. Es una tecnología que tuvo mucho éxito, ya que desde su lanzamiento en 2007 hasta 2018, solo en GitHub, tiene 27 millones de usuarios y más de 80 millones de repositorios.

Bibliografía:

https://azure.microsoft.com/es-es/overview/what-is-devops/#practices
https://www.autentia.com/2018/08/17/entendiendo-devops-en-5-minutos/
https://kinsta.com/es/blog/gitlab-vs-github/
https://tanzu.vmware.com/devops