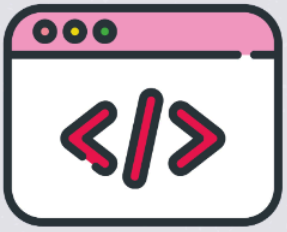
DigitalHouse>

Certified Tech Developer
The Ultimate Degree

Front End III

Los ejes de Infraestructura II

Infraestructura como código



La **infraestructura como código** es la gestión de la infraestructura en un modelo descriptivo, utilizando las mismas herramientas de versionado que un equipo utiliza para su código fuente. Así como el mismo código fuente genera el mismo código binario, un modelo de infraestructura como código debe generar el mismo entorno cada vez que se aplica. La infraestructura como código, en conjunto con los pipelines de despliegue continuo, permite automatizar los despliegues de infraestructura, haciéndolos más rápidos y menos propensos a errores, además nos evita depender de un equipo de infraestructura.

Pipelines de CI/CD

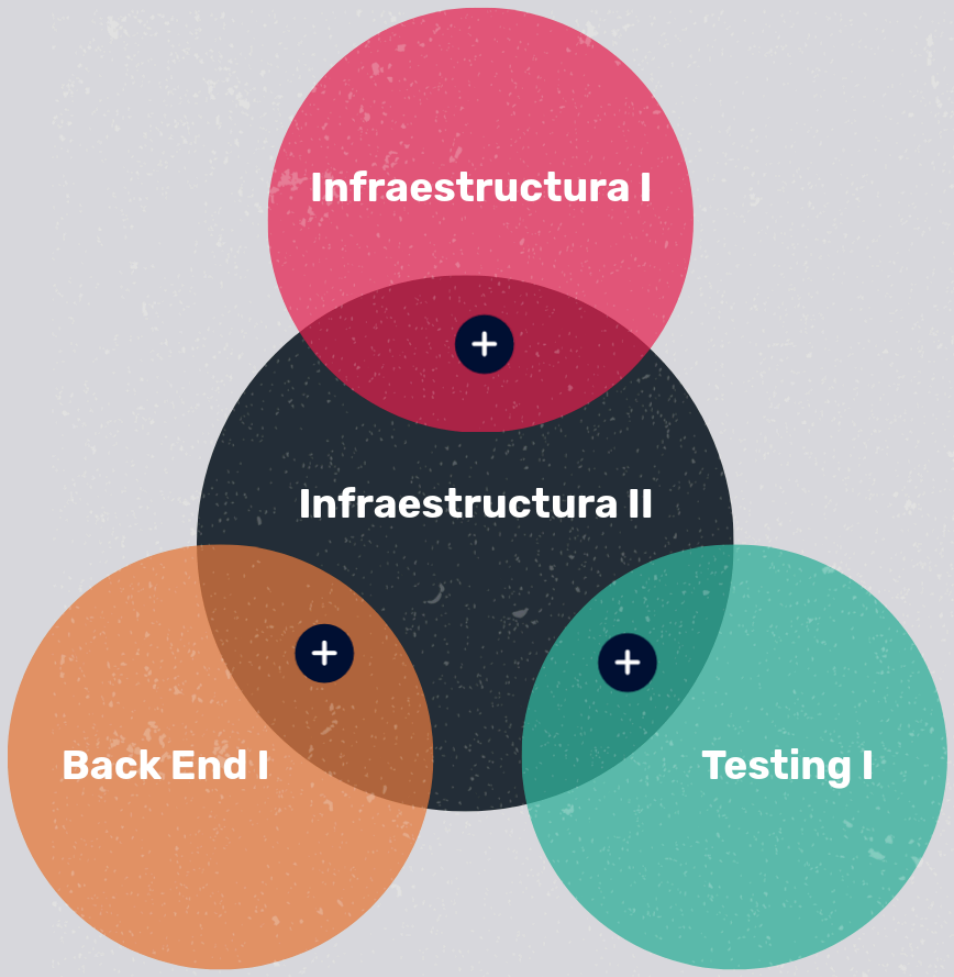
La **integración continua (CI)** es una práctica de desarrollo que consiste en integrar el código a un repositorio compartido de manera frecuente, idealmente varias veces al día. Cada integración es verificada por un proceso automatizado, permitiendo a los equipos detectar problemas rápidamente.
El **despliegue continuo (CD)** es la capacidad de poner en producción, en manos de los usuarios, cambios de cualquier tipo (nuevas funcionalidades, cambios de configuración, soluciones de errores y experimentos) de manera segura y sostenible. Esto se logra al asegurarnos que el código se encuentra en un estado desplegable, incluso al hacer cambios constantes.
Estas dos prácticas se llevan a cabo mediante **pipelines de CI o CD** respectivamente, que son procesos automatizados por los que pasa el código (código fuente o binarios) hasta llegar a su destino final, que puede ser un entorno de pruebas o el entorno de producción.

Monitoreo

El **monitoreo** se divide en dos grandes ramas:

- El **monitoreo de aplicaciones** es el proceso de medir la performance, disponibilidad y experiencia de usuario de una aplicación. Estas métricas se utilizan para identificar y resolver problemas en la aplicación antes que impacten a los usuarios.
- El **monitoreo de servidores** es el proceso de ganar visibilidad respecto a la actividad de nuestros servidores, sean físicos o virtuales. Se puede enfocar en distintas métricas de los servidores, pero las principales son la disponibilidad y la carga.

Conexión con otras materias



Infraestructura I

En Infraestructura I, vimos las bases, las diferentes tecnologías que vamos a poder utilizar y nos van a habilitar a desplegar, configurar, y monitorear nuestras aplicaciones.

Testing I

- ***API Testing:*** *Las pruebas de APIs pueden realizarse de forma automatizada como pasos dentro de un pipeline de CI/CD, facilitando de esa manera su ejecución reiterada (por ejemplo, en cada build).*

- ***Automatización de pruebas:*** *En el mundo moderno de infraestructura la automatización de pruebas acelera el proceso de compilación, distribución y despliegue de las aplicaciones, eliminando el factor del error humano y ahorrando el costo producido por la repetibilidad de tareas.*

.

Back End I

- ***Maven**: Esta herramienta de Back End I se puede utilizar manualmente y embebida dentro de nuestros procesos automatizados de build y release, también conocidos como pipelines.*

- ***Testing y JUnit:** Automatizar el despliegue de una aplicación no se trata solo de instalarla, sino que hay un conjunto de validaciones a ejecutar tanto durante el proceso de compilación como durante el proceso de liberación para verificar el correcto funcionamiento de la misma. Los procesos modernos de infraestructura son la amalgama de todas estas actividades.*

- ***REST APIs:** Las APIs son contratos entre nuestros sistemas, formas estandarizadas de intercambiar información, ejecutar acciones y tomar decisiones utilizando mensajes protocolizados. Cuando interactuamos con un proveedor de nube, ya sea por medio de herramientas de scripting o por herramientas de infraestructura como código, lo que está sucediendo por debajo es que estamos consumiendo una o un conjunto de APIs.*

- ***Docker:** Es nuestro amigo de Introducción a la Informática e Infraestructura I. Esta herramienta nos va a permitir construir nuestras aplicaciones en un formato trasladable y que no dependa de recursos externos al contenedor en sí mismo. Es muy común ver en los procesos de build y release la dockerizacion de la aplicación en cuestión.*

- ***Microservicios:** Arquitectura en la que nuestra aplicación se distribuye entre varios componentes más pequeños, especializados, que resuelven problemas específicos. Para poder construir de manera dinámica y ágil estos componentes podemos hacer uso de los procesos de build y release.*

.