Ingeniería del Software 2

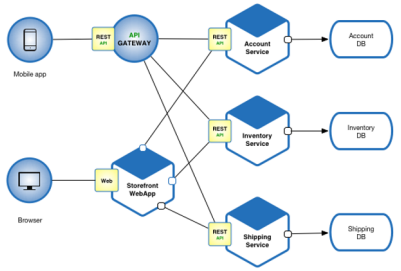
Arquitectura de Microservicios

Agenda

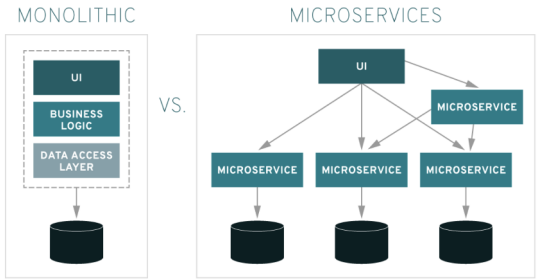
- Qué es la Arquitectura de Microservicios?
- Cuáles son las principales ventajas?
- Spring Boot y Spring Cloud
- Algunos Conceptos
- Un ejemplo de implementación

Microservicios

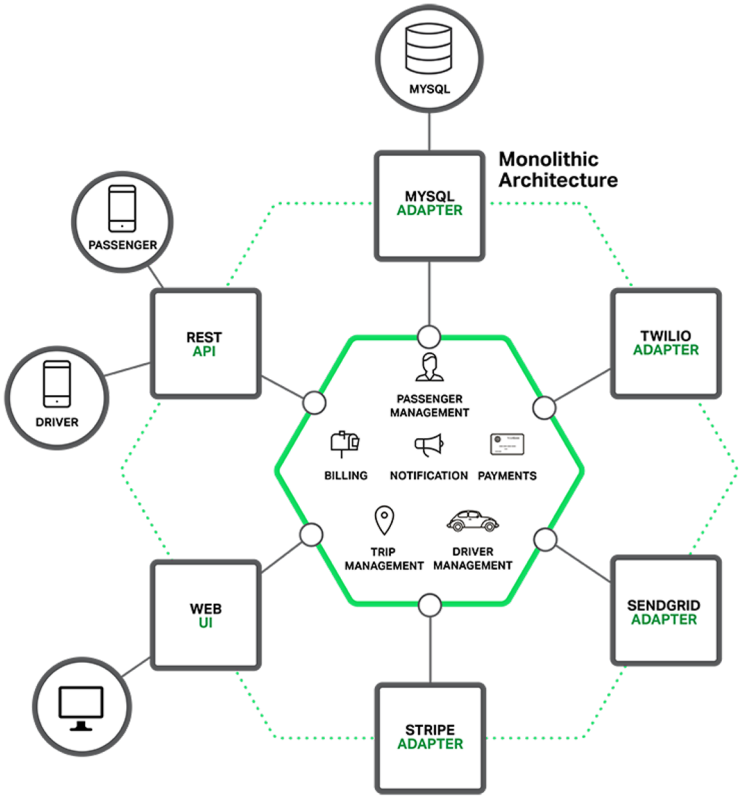
- Es un estilo arquitectónico que concibe a una aplicación como un conjunto de Servicios que se trabajan entre sí y son:
  - Altamente disponibles y testeables.
  - Débilmente acoplados.
  - Deployados independientemente.

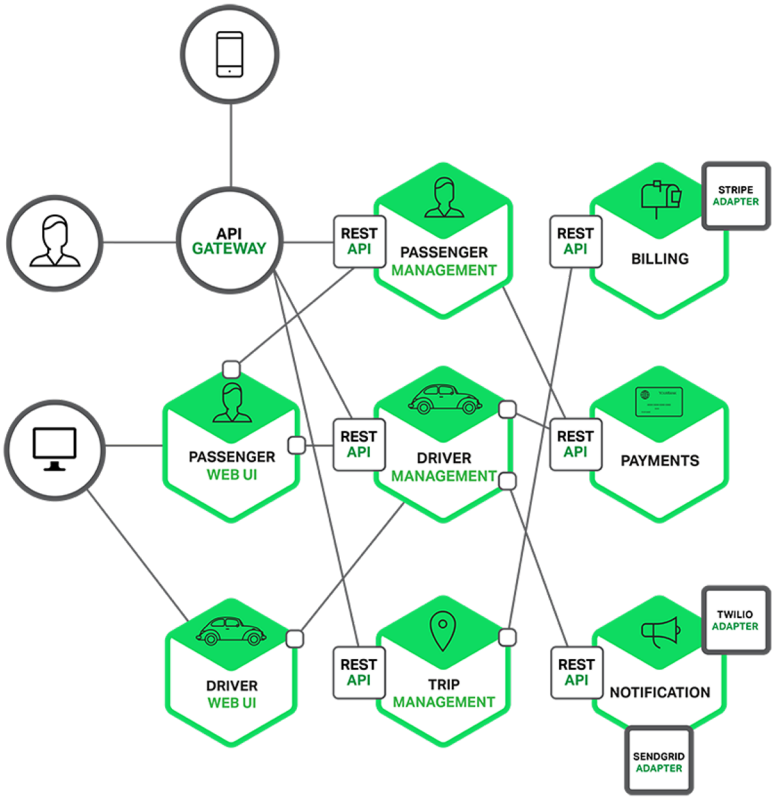


- Surgen como una alternativa a la arquitectura monolítica, la cual plantea a una aplicación como un sólo bloque de código.



Microservicios





Principales Ventajas

- Fomenta la división y modularidad del código.
- Cada microservicio es independiente del resto.
- Soportan mucho mejor la escalabilidad.
- Permite el intercambio de distintas tecnologías.

Principales Desventajas

- Implica trabajar en una aplicación distribuida (comunicación, mensajería).
- Resulta más complejo implementar cambios en funcionalidades que afecten varios servicios.
- Requiere un mucho mayor uso de devops.
- Inicialmente puede requerir de una infraestructura mayor que una arquitectura monolítica.

Spring Cloud

- Spring Cloud es un proyecto de Spring y brinda soporte de microservicios para:
  - Ruteo.
  - Registro y Descubrimiento de Microservicios.
  - Balanceo de Carga.
  - Circuit Breaking.
  - Configuración centralizada.
  - Mensajería Distribuida.
  - Y mucho más.

- Se basa en la idea de crear proyectos con Spring Boot (microservicios) y agregarles funcionalidad.

Algunos Conceptos - API Gateway

- Si bien cada servicio puede exponerse públicamente, es recomendable centralizar los accesos en un único punto.
- Esto brinda ventajas de poder implementar features cómo autorización, autenticación, cifrado de forma transparente.
- Muchas veces se centraliza ésto es un microservicio → ***API GATEWAY***.
- Spring Cloud nos trae como solución a: ***Spring Boot Zuul Proxy***

Algunos Conceptos - Service Registry

- Cuando una aplicación escala y se agregan nuevos servicios se hace necesario conocer la ubicación y estado de cada uno de ellos.

- Ante esto existen dos enfoques:
  - Microservice Discovery → se emiten mensajes de multicast y cada servicio contesta registrándose.
  - Microservice Registry → cuando cada servicio se inicia, se registra.

- Spring Cloud nos trae como solución a: ***Spring Boot Eureka***

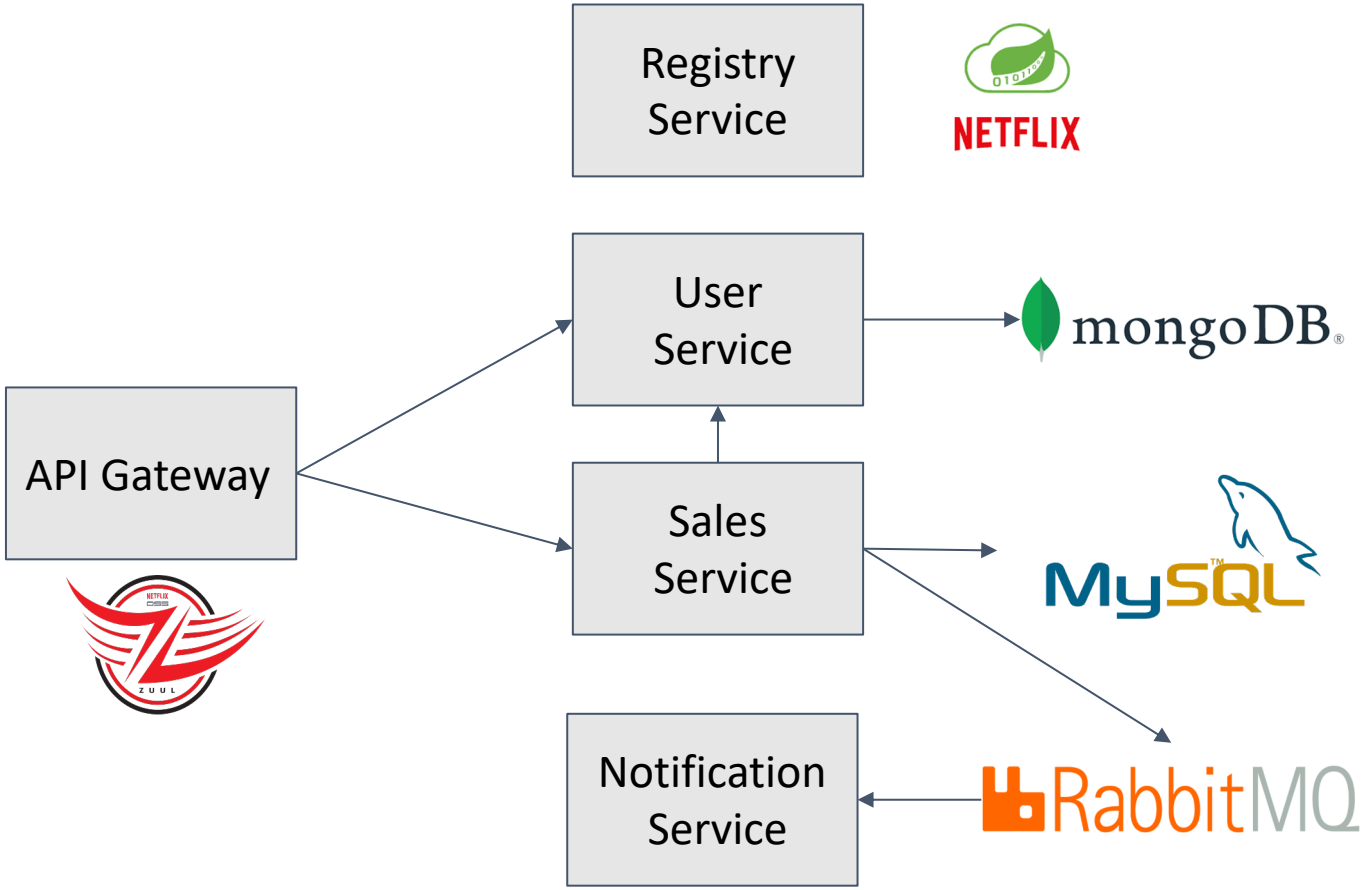
Comunicando Microservicios

- Generalmente los servicios pueden comunicarse entre sí para obtener información o para realizar acciones.

- Esta comunicación puede darse mayormente de dos maneras:
  - ***Comunicación Síncrona:*** un servicio invoca a otro y espera su finalización para continuar → llamadas HTTP.

  - ***Comunicación Asíncrona:*** un servicio puede notificar a otro de un determinado evento y continuar su ejecución → colas de mensajería.

Un ejemplo práctico



¿Dudas? ¿Consultas?