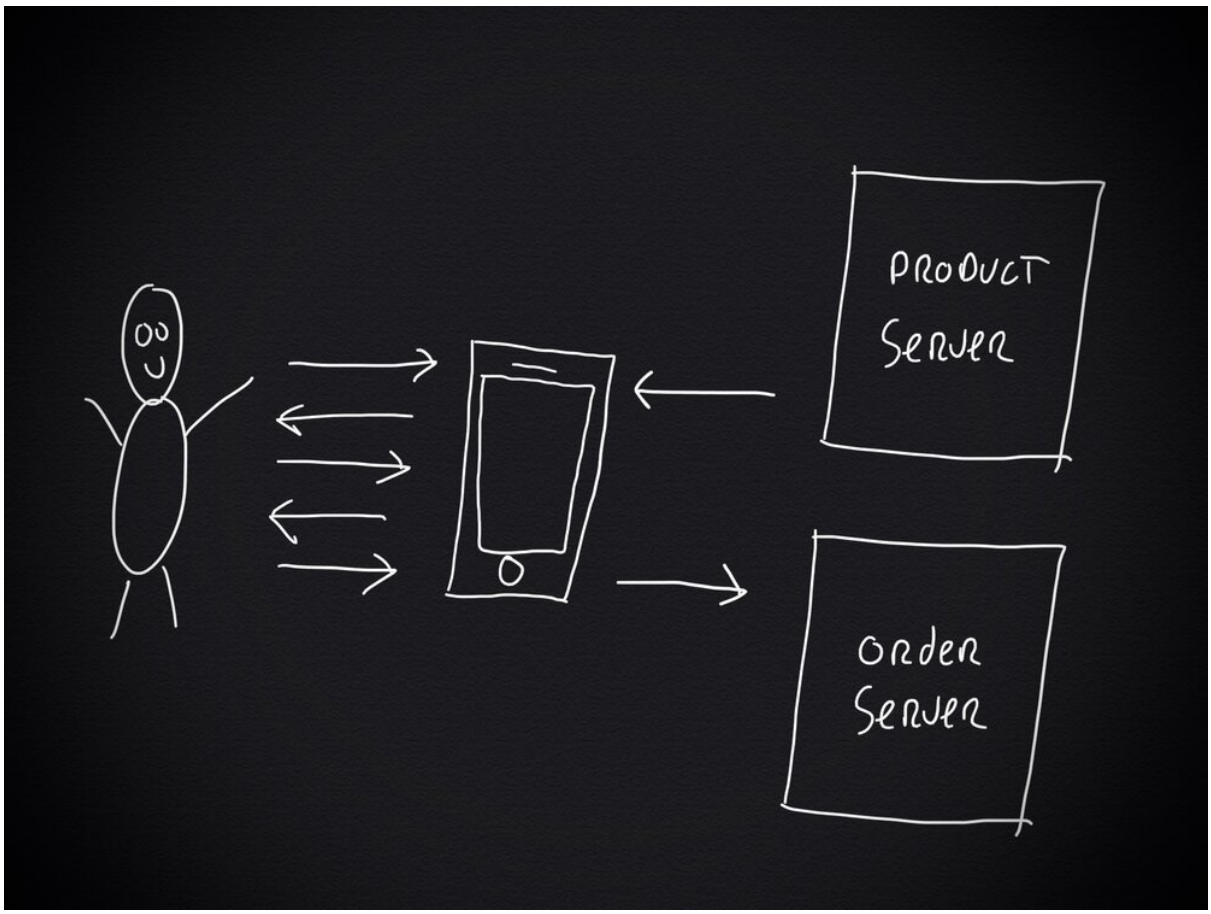
Diseño de aplicaciones escalables sin magia

Aprovechando la potencia de computación del cliente para un alto rendimiento con muchos usuarios

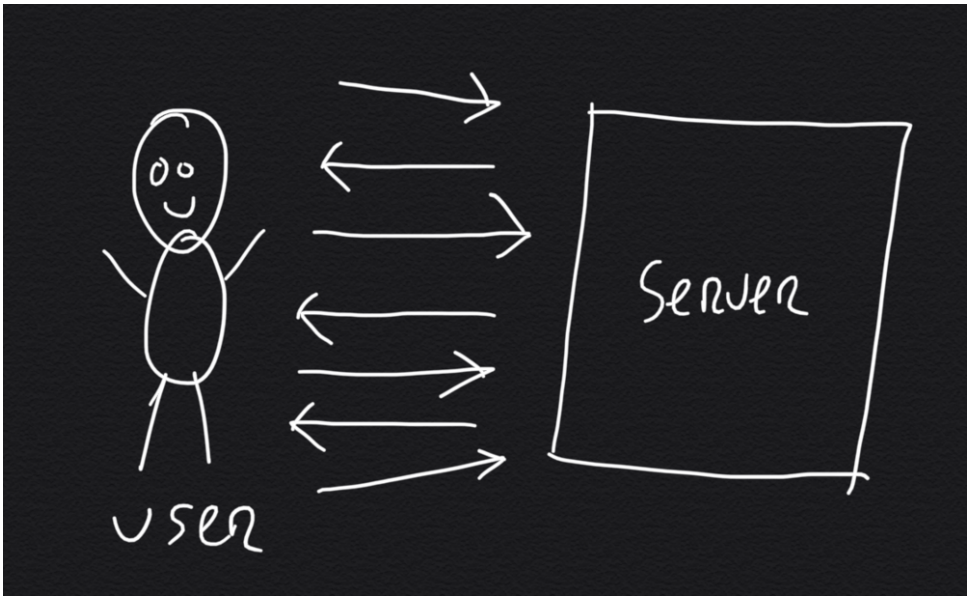
Willem L. Middelkoop
May 11, 2020



Como parte de la aplicación de pedidos de comida online que estoy construyendo, necesitaba diseñar una infraestructura backend escalable que pudiera manejar una gran cantidad de usuarios concurrentes. La escalabilidad se considera un problema difícil de abordar. A menudo se presenta como algo mágico, realizado por compañías millonarias utilizando herramientas secretas. Pero, no existe la magia, ¿o sí?

¿Qué es la escalabilidad?

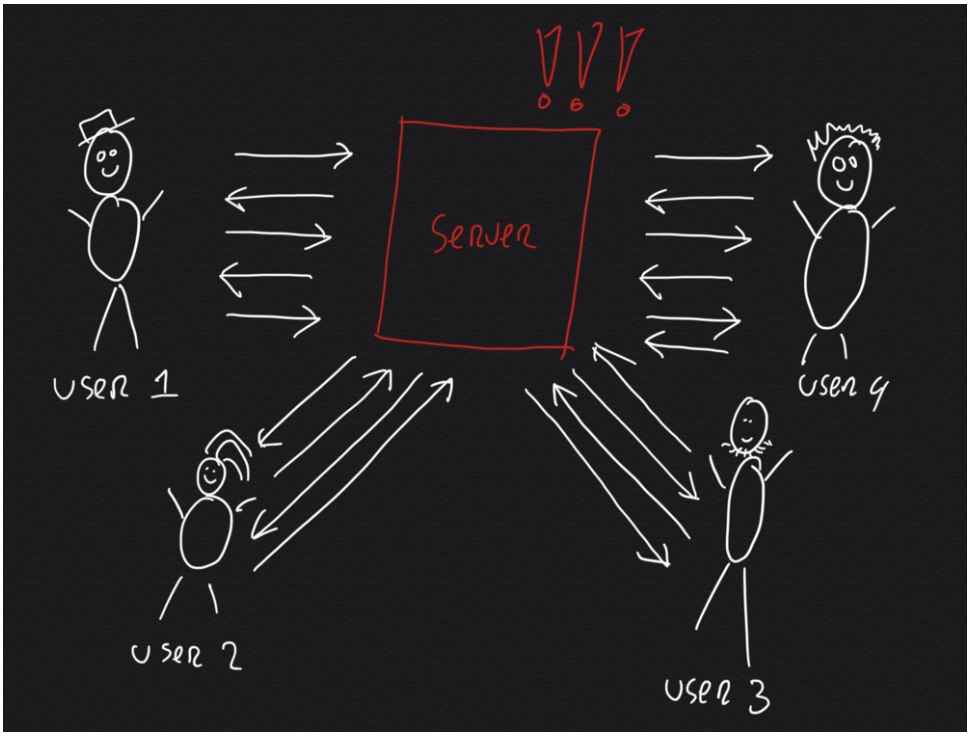
Si estás construyendo una aplicación, probablemente empieces muy pequeño. Solo unos pocos usuarios (o solo tú, el desarrollador) que trabajan con la aplicación. Todo irá de maravilla: la aplicación funciona bien, rápido y no hay problemas. *(¡aprecia esta sensación!)*



Un usuario interactuando con un servidor

En este escenario inicial y pequeño, el único servidor que ejecuta la aplicación puede manejar la interacción del usuario con ella. Cada vez que el usuario toca o hace clic en un botón, el servidor realiza algún trabajo. Si programas bien, estas cargas de trabajo se pueden optimizar en fragmentos de trabajo eficientes. Hacerlo de esta manera es un ejemplo típico de una arquitectura de software monolítica.

Incluso si tu proyecto o aplicación es bastante pequeño, en algún momento puede haber una afluencia de usuarios. Si tu sistema no puede soportar cargas elevadas, tiene altas probabilidades de fallar.



Servidor bajo alta carga manejando múltiples usuarios concurrentes

Debido al diseño de la aplicación monolítica, todos los usuarios interactúan con el mismo servidor. Toda su interacción (toques, clics, entrada) es manejada por este servidor. Con una mayor carga, tiene que trabajar mucho más. En cierto punto, notarás que las cosas se ralentizan porque el servidor no puede seguir el ritmo de la cantidad de trabajo que tiene que hacer. Esto significa que tu aplicación no escala bien, en otras palabras: estás en problemas.

Fallo en el 'momento supremo'

No escalar bien es un gran problema que nadie debería subestimar. La mayoría de las aplicaciones (y sus modelos de negocio) dependen de grandes volúmenes con pocos ingresos por usuario individual. Si quieres que tu aplicación sea un éxito (financiero), debes asegurarte de que funcione bien cuando haya un aumento repentino de tráfico. Si tu aplicación falla cuando de repente recibe la atención de las masas, perderás una oportunidad "única en la vida" para el crecimiento (orgánico).

En cierto modo, diseñar para la escalabilidad es como mantener los cinturones de seguridad abrochados cuando estás sentado en tu cohete construido por ti mismo, con motores experimentales. Nunca se sabe exactamente cuándo se encenderá, pero cuando lo haga, es mejor que te asegures de estar bien sujeto (¡y disfrutar del viaje!).

Formas comunes de lograr la escalabilidad

Se ha escrito mucho sobre la creación de aplicaciones escalables, las formas comunes de mejorar la escalabilidad son:

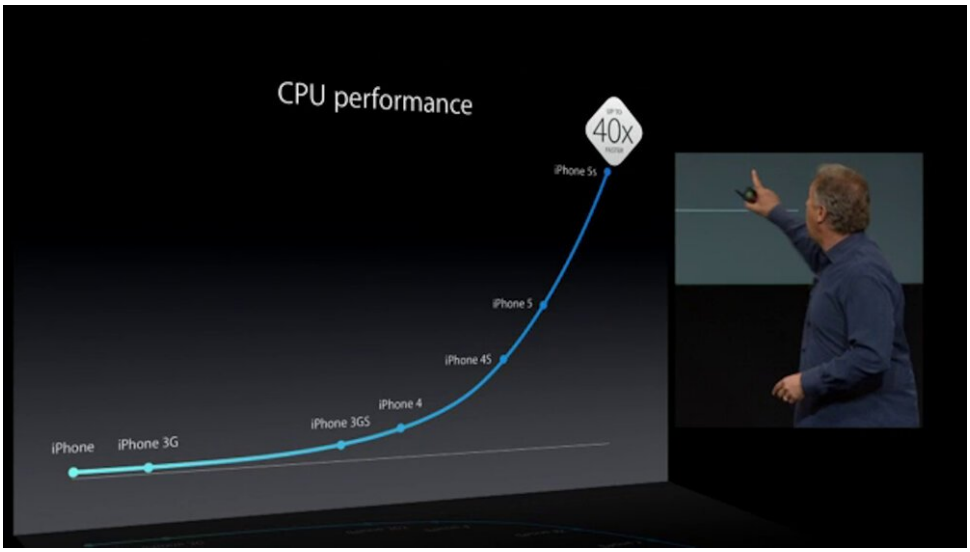
- **Aumentar la capacidad del servidor**: añadir más memoria, potencia de CPU, almacenamiento. Esto funcionará, pero solo te llevará "hasta cierto punto".

- **Añadir más servidores**: en lugar de un solo servidor que actúe como cuello de botella, puedes añadir más servidores que compartirán la carga de trabajo. Es más fácil decirlo que hacerlo, ya que requerirá equilibrio de carga y uso compartido de datos, lo que puede ser complicado (por ejemplo, si tienes un contador de pedidos y dos servidores están contando, ¿cuál determina el siguiente número?).
- **Optimizar el código de programación**: ¡usa las herramientas adecuadas! Elige un lenguaje de programación orientado al rendimiento y un software de servidor similar. Investiga la programación funcional para permitir que tu código se ejecute en múltiples núcleos de forma asíncrona. Hazlo sin estado. Compáralo. Optimízalo. Cada milisegundo ganado en una sola solicitud se suma rápidamente cuando se manejan grandes volúmenes.
- **Usar la base de datos y separarla del servidor de aplicaciones**: si estás utilizando una base de datos, ¡aprovecha su poder! Aprovecha el almacenamiento en caché de consultas, los índices y las capacidades de búsqueda. La mayoría de las soluciones de bases de datos ofrecen opciones de agrupación a prueba de batalla; no intentes reinventar la rueda que otros ya han perfeccionado.

Lo mejor que puedes hacer es considerar todas estas opciones. Incluso si no estás añadiendo más servidores de inmediato, debes escribir tu código para que lo admita más adelante. Comparar y optimizar tu código debe ser una parte integral de tu trabajo, no solo una idea de último momento.

Aprovechar la potencia de cálculo del cliente

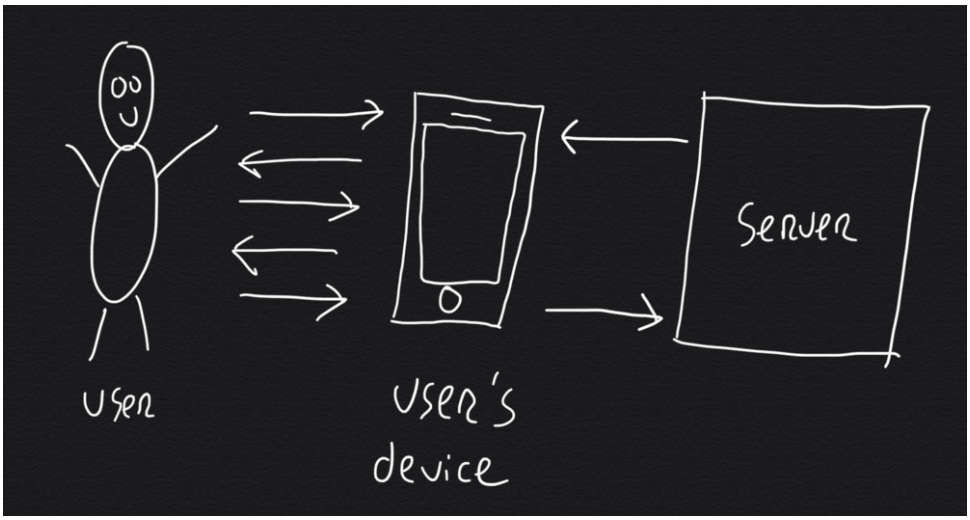
Cuando hayas considerado todas las formas comunes de lograr la escalabilidad, hay "una cosa más".



Los teléfonos inteligentes (y tabletas, PC) se han vuelto más potentes a lo largo de los años (imagen: Apple)

Pocos desarrolladores consideran esto: ¡el problema de escalabilidad contiene la solución! Cuando tienes 1000 usuarios, tienes 1000 computadoras con potentes CPU y mucha memoria. Las computadoras y los teléfonos inteligentes modernos se han vuelto cada vez más capaces.

Simplemente tienes que encontrar una manera de aprovechar la potencia de cálculo del cliente. Si bien puede sonar como algo malvado (usar la potencia de cálculo del usuario), les proporcionará una experiencia mucho mejor y más rápida. Lo mejor es que *su* potencia de cálculo se gastará en *su* fantástica experiencia, resolviendo tu problema de escalabilidad como un feliz efecto secundario.



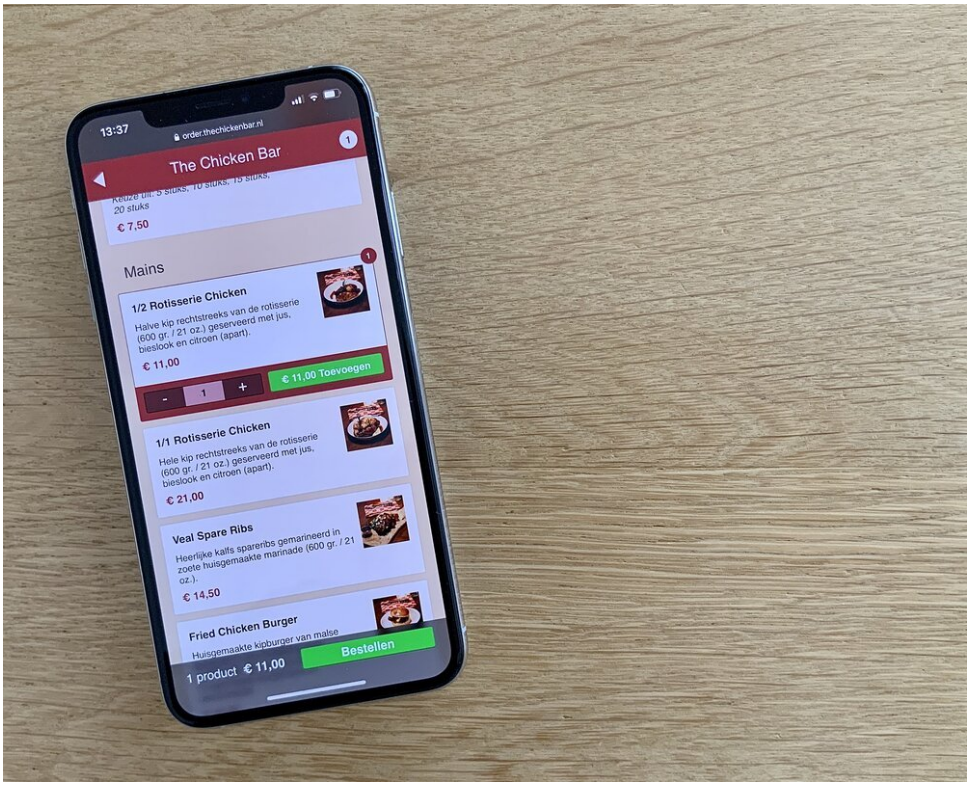
Aprovechando el dispositivo de un usuario: el teléfono del usuario hace la mayor parte del trabajo que originalmente hacía el servidor

Al aprovechar la potencia de cálculo del usuario, reduces la cantidad de trabajo que tiene que hacer el servidor. En lugar de que el servidor haga un poco de trabajo *cada* vez que el usuario toca o hace clic en un botón, este trabajo ahora lo realiza principalmente el dispositivo del usuario. Esto es más eficiente que comunicarse a través de una conexión Wifi/4G con un servidor (ahorrando batería y tiempo).

Requiere un poco de replanteamiento, pero este principio también se puede aplicar a las aplicaciones web. En lugar de manejar todo en el servidor, puedes realizar mucho trabajo en el cliente. Cosas como buscar, navegar, filtrar, navegar e incluso agregar cosas a un carrito de compras se pueden hacer usando JavaScript del lado del cliente. Tú, querido desarrollador, tienes que buscar formas de hacer que esto funcione, pero si tienes éxito, tendrás una escalabilidad casi infinita a tu alcance.

En la práctica: aplicación de pedidos de comida

Para la aplicación de pedidos de comida busqué formas de aprovechar la potencia de cálculo del cliente para lograr un alto rendimiento. El principal desafío con los pedidos de entrega y para llevar es que alcanzan su punto máximo a la hora de la cena. Mucha gente usa la aplicación al mismo tiempo, esta es una receta para problemas de escalabilidad.

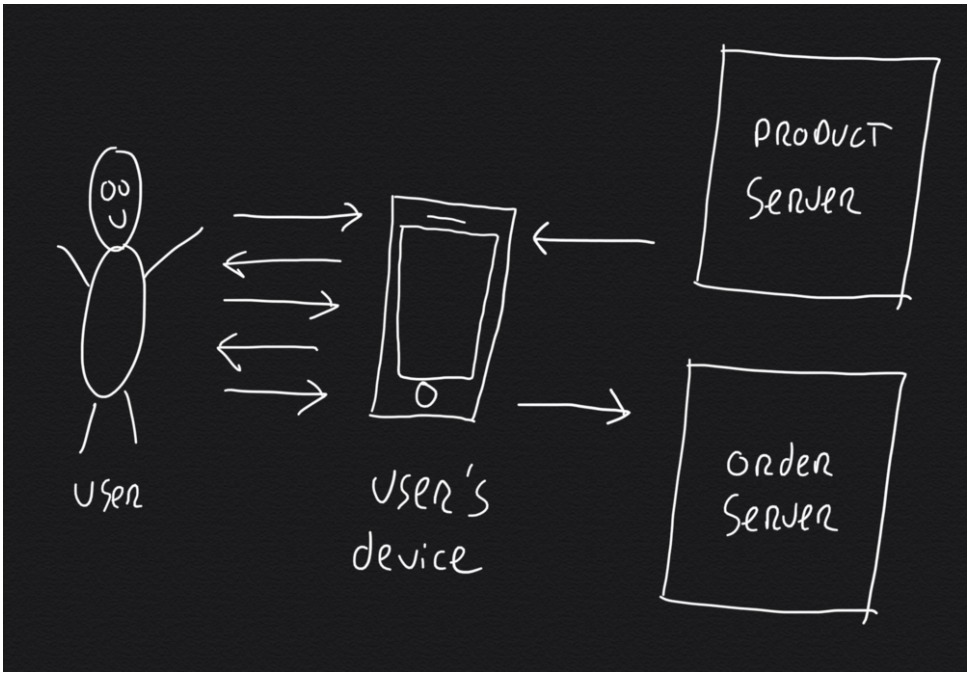


Pedir comida en línea

Si piensas en pedir comida en línea, puedes dividir todo el proceso en partes más pequeñas:

- **1) Abrir la página**: cargar la aplicación/página
- **2) Navegar por las diferentes opciones**: listar los productos, ver sus descripciones, precios, fotos, etc.
- **3) Buscar algo específico**: buscar por categoría, nombre del producto, etc.
- **4) Seleccionar un producto**: agregarlo a tu pedido
- **5) Personalizar un producto**: seleccionar una salsa, guarnición, aderezo, etc.
- **6) Ingresar tu información**: tu nombre, número de teléfono, detalles de entrega, etc.
- **7) Pagar tu pedido**: seleccionar un método de pago, conectarse a tu banco o tarjeta de crédito
- **8) Cerrar la aplicación**

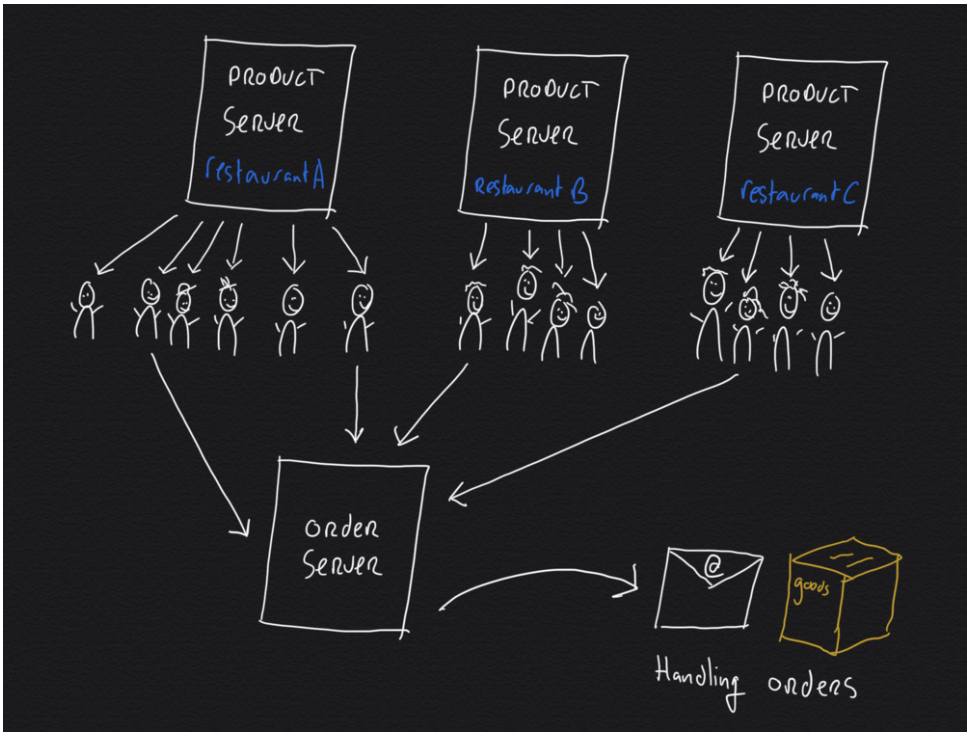
La mayoría de estos pasos se pueden programar de tal manera que el servidor *no sea necesario*. Solo el paso 1 (**cargar**) y el paso 7 (**pagar**) requieren contacto con la infraestructura de backend. Es importante darse cuenta de que los pasos del 2 al 5 se repiten varias veces, ya que es común que los usuarios agreguen varios productos a su pedido.



Separando la carga de trabajo restante del servidor: servir productos y gestionar pedidos

Al separar la carga de trabajo restante por tarea, el software se puede optimizar aún más. El **"servidor de productos"** está optimizado para servir activos estáticos (textos, precios, imágenes) que permiten al cliente crear la experiencia de navegación del producto. Puedes optimizar mucho este tipo de servidor, aprovechando el almacenamiento en caché, HTTP/2, la compresión, etc.

Solo aquellos usuarios que completen el proceso de pedido serán reenviados al **"servidor de pedidos"** que maneja el pago. Se conectará al proveedor de pagos. Una vez que se completa un pago, el proveedor de pagos se comunicará con el servidor de pedidos para informarle sobre el estado del pago. Esto es algo sobre lo que deseas tener control total (del lado del servidor), como lo hice cuando diseñé un sistema de pago anteriormente. La carga de trabajo restante en el "servidor de pedidos" es mucho menor, ya que no todos realizarán un pedido, y gran parte de su trabajo de procesamiento de pedidos se puede realizar en segundo plano.



Manejando mucho tráfico a través de la computación distribuida

Para maximizar la escalabilidad, puedes agregar fácilmente varios servidores (o usar una red de entrega de contenido) para servir los activos estáticos. Para la aplicación de pedidos de comida, utilizo un "servidor de productos" separado *por restaurante*. Esto permite un alto rendimiento y reduce los tiempos de carga. La gente nota que es rápido, algunos no pueden creer que sea tecnología web, es *casi* mágico.

Conclusión

Diseñar una aplicación escalable se puede hacer sin magia ni presupuestos millonarios. Solo tienes que pensarlo bien y buscar oportunidades que se presenten junto con los desafíos.

El problema contiene la solución. Lidiar con muchos usuarios concurrentes no es solo un problema: traen su propia potencia de cálculo, ¡es la solución! Solo tienes que usarla.