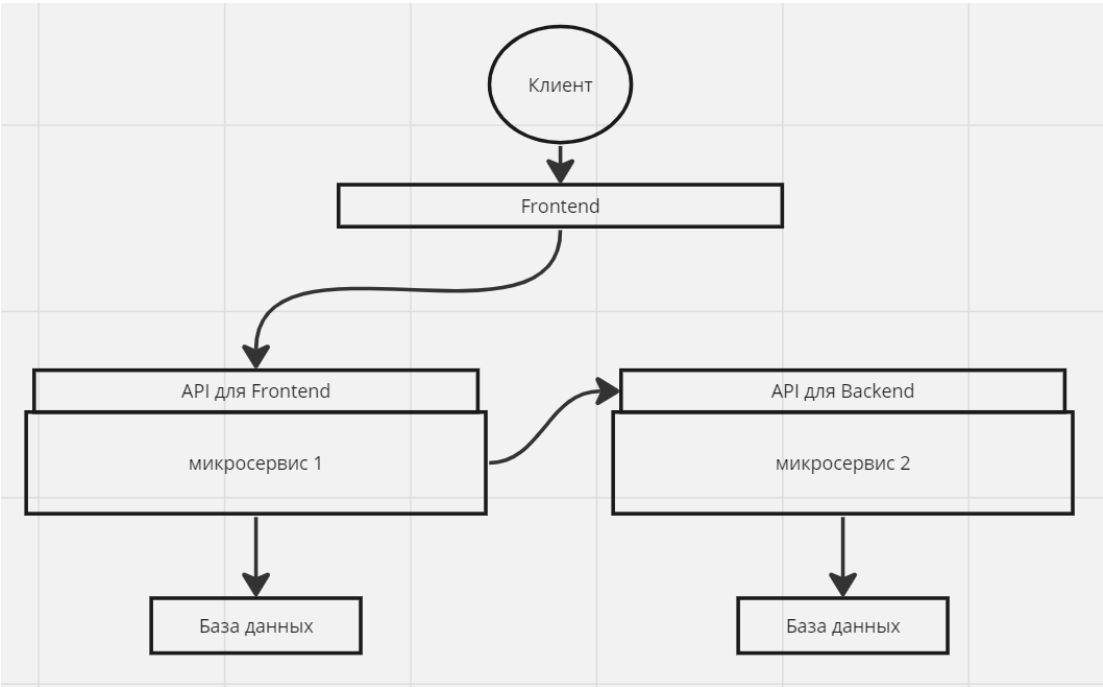


# Что такое микросервис

В бизнес-контексте микросервис — это самостоятельная функциональность или процесс (только одна функциональность или процесс, а не несколько, как в монолите!), которым можно управлять и разворачивать независимо от остальных частей системы. Пример: сервис для создания заказов.

Технически, микросервис — это тоже самое что монолит, только в нём меньше бизнес логики (помним только про одну функцию).

То есть, это независимый компонент программного обеспечения, который выполняет конкретную бизнес-функцию и общается с другими компонентами через API. Он может быть разработан, развернут, масштабирован и обновлен независимо от других микросервисов.



С границами микросервиса определились. А что с распределением микросервисов на команды? Сколько должно быть микросервисов у одной команды разработки?

Когда одна команда отвечает за множество микросервисов, это может привести к ряду проблем:

- 1. **Сложность управления:** Чем больше сервисов у команды, тем труднее им управлять. Это влияет на качество кода, тестирование, развертывание и мониторинг.
- 2. **Размывание фокуса:** Команда может потерять фокус на конкретных бизнес-целях, так как ей приходится заниматься различными аспектами системы.
- 3. **Снижение скорости разработки:** С ростом числа сервисов увеличивается сложность их взаимодействия, что замедляет процесс разработки.

Когда много команд отвечает за один микросервис, это тоже может привести к ряду проблем:

- 1. **Конфликты интересов:** Разные команды могут иметь разные представления о том, как должен функционировать сервис, что может привести к конфликтам.
- 2. **Затруднение координации:** Если множество команд работает над одним сервисом, требуется значительное усилие для координации изменений и развертывания.
- 3. **Риск снижения качества:** Когда один сервис находится под управлением нескольких команд, повышается риск внедрения некачественного кода и ошибок.

К чему можно стремиться - одной команде один или несколько(пара) микросервисов, это даёт:

- 1. **Четкая ответственность:** Каждая команда полностью отвечает за свой микросервис, от разработки до эксплуатации, что упрощает процесс принятия решений.
- 2. **Быстрая итерация:** Команда может быстро и независимо разрабатывать, тестировать и развертывать свой микросервис.
- 3. **Сосредоточение на бизнес-логике:** Команда может полностью сосредоточиться на бизнес-логике и функциональности своего конкретного микросервиса.
- 4. **Упрощение масштабирования:** Когда у одной команды один сервис, масштабирование становится значительно проще, так как команда является экспертом в данной области.

5. **Изоляция от сбоев:** Ошибки или сбои в одном микросервисе не влияют на работу других, что повышает общую устойчивость системы.

Этот подход часто описывается в методологии DevOps и в культуре "you build it, you run it", где команда, создающая сервис, также отвечает за его эксплуатацию.