

## Выводы по микросервисам:

Цель использования микросервисной архитектуры - ускорение процесса разработки за счет возможности параллелизма работ и независимости компонентов системы (это главное), повышение гибкости, масштабируемости и надежности системы. Но это очень дорого и сложно.

- **Независимость разработки и развертывания:** Микросервисы позволяют разрабатывать и разворачивать различные компоненты системы независимо друг от друга. Это ускоряет процесс разработки и упрощает масштабирование.
- **Упрощение обслуживания:** В случае сбоя, команда может быстро определить, какой сервис вызвал проблему, что упрощает отладку и устранение неполадок.
- **Гибкость технологического стека:** Каждый микросервис может использовать технологии, которые наиболее подходят для выполнения своих задач, независимо от технологий, используемых другими сервисами.
- **Масштабируемость:** Микросервисы можно масштабировать независимо друг от друга, что позволяет более эффективно использовать ресурсы.
- **Снижение риска:** Если один микросервис выйдет из строя, он не повлияет на работу остальных сервисов.

**Выбор архитектуры приложения зависит от множества факторов**, включая размер команды разработчиков, стадию жизненного цикла продукта и специфику бизнеса. Нет "лучшей" архитектуры в абсолютном смысле, есть лишь наиболее подходящий выбор в конкретной ситуации.

**Монолит:** Это частый выбор на начальной стадии идеи или стартапа, когда быстрота разработки и простота изменений приоритетнее, чем масштабируемость и сложность управления кодовой базой. Все части приложения тесно связаны и работают в рамках одного процесса.

**Распределённый монолит:** Это вариант монолитной архитектуры, но с разделением на фронтенд приложение и бэкенд. Это удобно для продуктов, над которыми работают до ~100 разработчиков, поскольку позволяет разделить работу по клиентской и серверной частям приложения.

SOA: Это стиль архитектуры, где приложение состоит из отдельных сервисов, каждый из которых выполняет конкретные функции. SOA часто встречается в больших корпорациях с "legasy" (устаревшими) системами, где нежелательно или невозможно полностью переписать существующую систему.

Микросервисы: Это подход, где приложение состоит из множества независимых сервисов, каждый из которых может разрабатываться, развертываться и масштабироваться независимо. Микросервисы обычно выбираются для больших продуктов, над которыми работает больше 100 разработчиков, поскольку они обеспечивают высокую степень гибкости и масштабируемости.

Важно понимать, что выбор архитектуры должен основываться на реальных потребностях бизнеса и команды, а не на текущих тенденциях или личных предпочтениях. Каждый стиль архитектуры имеет свои преимущества и недостатки, и их нужно учитывать при принятии решения.