

# Микросервисная архитектура для работы с данными

ruslansmol19@gmail.com [Сменить аккаунт](#) Не



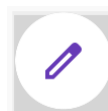
будет видно получателю

**Какой у вас уровень опыта в разработке микросервисных архитектур?**

- ☐ Начинающий (менее 1 года)
- ☐ Опытный (1-3 года)
- ☐ Эксперт (более 3 лет)

**Какие преимущества микросервисной архитектуры вы считаете наиболее важными?**

- ☐ Масштабируемость Гибкость
- ☐ разработки Упрощенное
- ☐ тестирование
- ☐ Независимое развертывание
- ☐ Другое:



**Какие инструменты или фреймворки вы используете для создания микросервисов?**

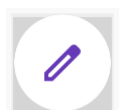
- ☐ Spring Boot
- ☐ Node.js
- ☐ Flask/FastAPI
- ☐ .NET Core
- ☐ Другое:

**Какую базу данных вы предпочитаете использовать в микросервисах?**

- ☐ Реляционные (PostgreSQL, MySQL)
- ☐ NoSQL (MongoDB, Cassandra)
- ☐ Гибридный подход
- ☐ Зависит от задачи

**Насколько важно, чтобы микросервисы были автономными?**

	1	2	3	4	5	
Совсем не важно	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Критически важно



**Как вы решаете проблему согласованности данных между микросервисами?**

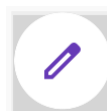
- ☐ Использую распределенные транзакции с помощью паттерна Saga для координации действий между сервисами.
- ☐ Применяю event sourcing для отслеживания изменений в данных и обеспечения их согласованности
- ☐ Синхронизирую данные через общее хранилище (например, Kafka) и обрабатываю события асинхронно.
- ☐ Использую базу данных для каждого сервиса и периодически выполняю синхронизацию через API.
- ☐ Пока не сталкивался с этой проблемой, но планирую изучить подходы, такие как CQRS

**Используете ли вы event-driven архитектуру в своих микросервисах?**

- ☐ Да
- ☐ Нет
- ☐ Планирую внедрить

**Какие инструменты для оркестрации микросервисов вы используете?**

- ☐ Kubernetes
- ☐ Docker Swarm
- ☐ AWS ECS
- ☐ Не использую
- ☐ Другое:



**Как часто вы сталкиваетесь с проблемами производительности в микросервисах?**

	1	2	3	4	5	
Очень редко	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Постоянно

**Как вы мониторите и логируете работу микросервисов?**

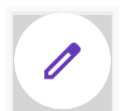
- ☐ Prometheus + Grafana
- ☐ ELK Stack (Elasticsearch, Logstash, Kibana)
- ☐ Splunk
- ☐ Другое:

**Какие методы защиты данных вы применяете в микросервисах?**

- ☐ Шифрование данных
- ☐ Аутентификация/Авторизация (OAuth, JWT)
- ☐ Сегментация сети
- ☐ Другое:

**Как вы оцениваете сложность перехода с монолитной архитектуры на микросервисы?**

	1	2	3	4	5	
Очень просто	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Очень сложно

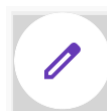


**Какие основные вызовы вы видите при работе с микросервисами?**

- ☐ Очень сложно поддерживать согласованность данных между сервисами, особенно в случае отказов
- ☐ Рост сложности мониторинга и логирования, так как система становится более распределенной.
- ☐ Необходимость постоянного обучения команды новым технологиям и подходам.
- ☐ Увеличение времени развертывания новых версий из-за большого количества зависимостей
- ☐ Сложность тестирования системы в целом, особенно при интеграции нескольких сервисов.
- ☐

**Какие дополнительные технологии или практики вы хотели бы изучить для работы с микросервисами?**

- ☐ Хочу изучить Kubernetes для оркестрации контейнеров и автоматизации развертывания.
- ☐ Планирую углубиться в event-driven архитектуру и Kafka Streams для обработки событий.
- ☐ Интересуюсь Service Mesh (например, Istio) для управления взаимодействием между сервисами.
- ☐ Хотел бы научиться эффективно использовать Prometheus и Grafana для мониторинга.
- ☐ Изучаю подходы к проектированию отказоустойчивых систем, такие как Circuit Breaker.
- ☐



### Что вы считаете ключевым фактором успеха микросервисной архитектуры?

- ☐ Правильное разделение сервисов на автономные компоненты с четко определенными границами.
- ☐ Надежная система мониторинга и логирования для быстрого выявления проблем.
- ☐ Использование стандартных протоколов и форматов данных (например, REST, gRPC, JSON).
- ☐ Культура DevOps и автоматизация процессов развертывания и тестирования. Гибкость в
- ☐ выборе технологий для каждого сервиса в зависимости от задач.
- ☐

Отправить

Очистить форму

Никогда не используйте формы Google для передачи паролей.

Компания Google не имеет никакого отношения к этому контенту. - Условия использования - Политика конфиденциальности

Does this form look suspicious? [Отчет](#)

Google Формы

