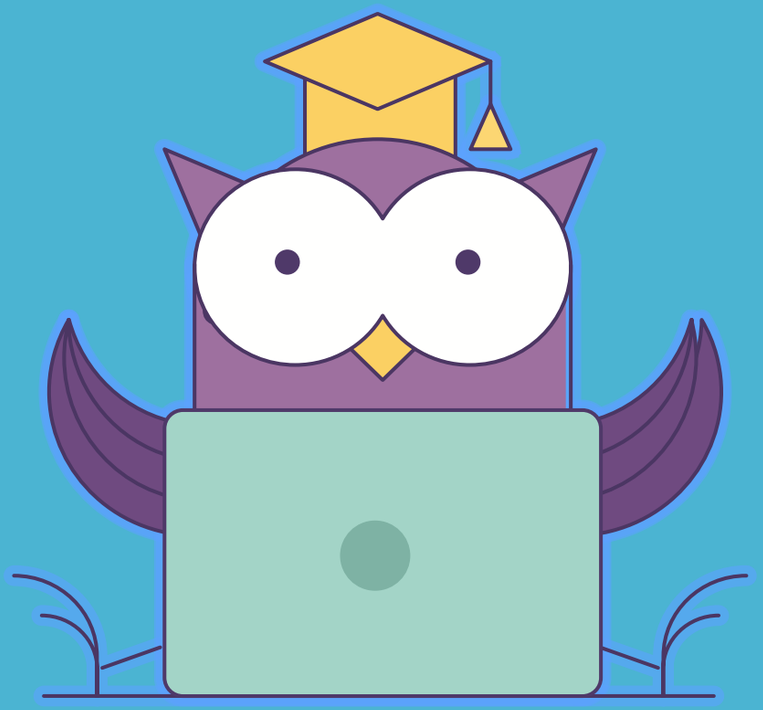
ОНЛАЙН-ОБРАЗОВАНИЕ

OTUS

Как меня слышно и видно?



> Напишите в чат

+ если все хорошо

- если есть проблемы со звуком или с видео

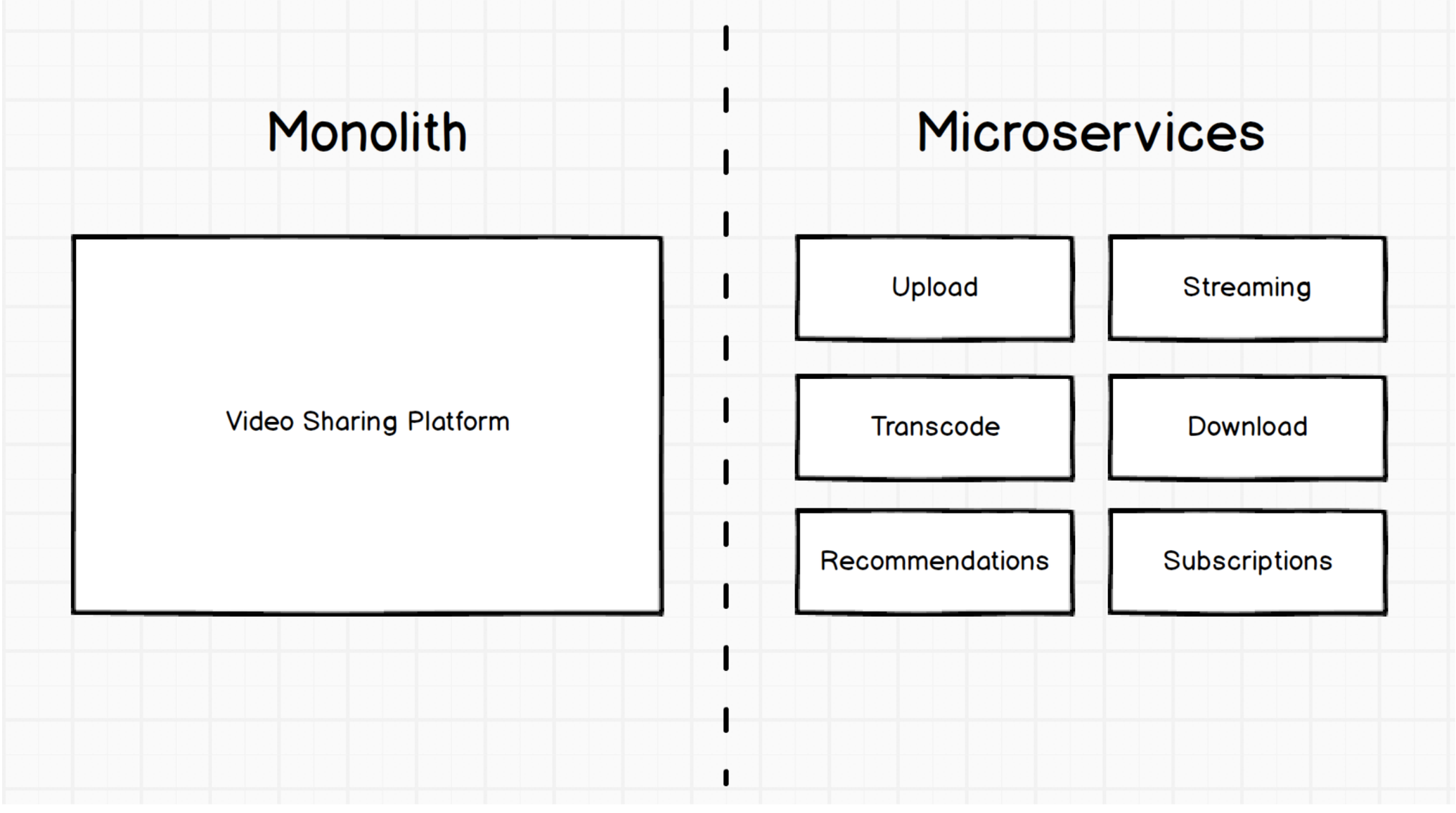
!проверить запись!

Монолит и микросервисы

Желтак Артем

План занятия

- Монолит и микросервисы
- Плюсы и минусы микросервисной архитектуры
- 12 Factor application
- Serverless


Monolith
Video Sharing Platform
Microservices
Upload
Streaming
Transcode
Download
Recommendations
Subscriptions

Микросервисы: отличия

- Микро не обязательно про размер, но про зону ответственности
- Самодостаточны, идеальны для горизонтального масштабирования
- Разные технологии для разных задач
- Распределенная кодовая база

Микросервисы: плюсы

- Независимая разработка
- Независимое развёртывание
- Независимая масштабируемость
- Повторное использование
- Проще юнит тестирование

Микросервисы: минусы

- Возросшая сложность для тестирования
- Возросшая сложность для разработчиков
- Возросшая сложность для эксплуатации/Devops
- Выше требования к компетенции

Микросервисы: отличия (минусы часть 2)

- Удаленные вызовы дороже локального исполнения
- Реальные системы обычно не имеют чётко определённых границ
- Сложности stateful
- Версионность и работа с ней
- Сложности взаимодействия между сервисами
- Распределенные транзакции
- Попытка замаскировать монолит

1. Кодовая база
Одна кодовая база, отслеживаемая в системе контроля версий, – множество развёртываний
2. Работа с зависимостями
Явно объявляйте и изолируйте зависимости
3. Конфигурация
Сохраняйте конфигурацию в среде выполнения
4. Сторонние службы (Backing Services)
Считайте сторонние службы (backing services) подключаемыми ресурсами

5. Сборка, релиз, выполнение
Строго разделяйте стадии сборки и выполнения
6. Процессы
Запускайте приложение как один или несколько процессов не сохраняющих внутреннее состояние (stateless)
7. Привязка портов
Экспортируйте сервисы через привязку портов
8. Параллелизм
Масштабируйте приложение с помощью процессов

9. Утилизируемость
Максимизируйте надёжность с помощью быстрого запуска и корректного завершения работы
10. Паритет разработки/работы приложения
Держите окружения разработки, промежуточного развёртывания (staging) и рабочего развёртывания (production) максимально похожими
11. Журналирование
Рассматривайте журнал как поток событий
12. Задачи администрирования
Выполняйте задачи администрирования/управления с помощью разовых процессов

- Сервер все таки есть
- Function as Service
- Долгий запуск, короткое время жизни
- Нет зависимости от инфраструктуры, но есть vendor lock

- Сократить простой ресурсов
- Ускорить разработку (писать нужно только бизнес логику)
- Масштабирование никогда не было таким простым
- Не подходит для realtime систем

- https://12factor.net/ru/
- https://habr.com/ru/company/flant/blog/347518/
- https://habr.com/ru/company/selectel/blog/452266/

Опрос

Заполните пожалуйста опрос

https://otus.ru/polls/4749/

Спасибо за внимание!