

## Способ уменьшения задержки - оптимизация размера данных

Оптимизация размера данных — одна из ключевых тем в уменьшении задержки и увеличении производительности веб-приложений. Давайте разберём это на разных уровнях.

### Уровень сети

#### Сжатие данных

1. Применение алгоритмов сжатия данных на сетевом уровне, например, при использовании протоколов, которые поддерживают сжатие.
2. **Плюсы и минусы:**
  - Уменьшение объема передаваемых данных.
  - Возможный негативный эффект на производительность из-за необходимости распаковки.

#### Оптимизация протоколов

1. Использование более эффективных сетевых протоколов, таких как HTTP/2 (или gRPC между микросервисами), которые позволяют уменьшить задержку.
2. **Плюсы и минусы:**
  - Ускорение загрузки страниц.
  - Сложность миграции и настройки.

### Уровень фронтенда

#### Минимизация и сжатие ресурсов

1. Минимизация CSS, JS и других файлов для уменьшения их размера.

## 2. Плюсы и минусы:

- Быстрое время загрузки.
- Потенциальная потеря читаемости и отладочной информации.

## Использование кэша и lazy-loading

1. Кэширование статических ресурсов и отложенная загрузка (lazy-loading) изображений и другого контента.

## 2. Плюсы и минусы:

- Уменьшение объема загружаемых данных.
- Сложность управления кэшем.

## Уровень бэкенда

### Пагинация и лимит запросов

1. Ограничение количества данных, возвращаемых в одном запросе.

## 2. Плюсы и минусы:

- Уменьшение задержки ответа.
- Необходимость дополнительных запросов для получения всей информации.

### Оптимизация запросов к базе данных

1. Выборка только необходимых полей, использование индексов.

## 2. Плюсы и минусы:

- Увеличение скорости выполнения запросов.

- Сложность оптимизации и поддержки.

## **Уровень базы данных**

### **Сжатие данных**

1. Использование механизмов сжатия, поддерживаемых СУБД.

#### **2. Плюсы и минусы:**

- Уменьшение занимаемого дискового пространства.
- Потенциальное увеличение времени выполнения запросов из-за необходимости распаковки.

### **Денормализация данных**

1. Структурирование данных так, чтобы уменьшить количество необходимых запросов для получения полной информации.

#### **2. Плюсы и минусы:**

- Увеличение скорости чтения.
- Увеличение сложности поддержки и рисков ошибок.

## **Уровень инфраструктуры**

### **Уменьшение размера образов контейнеров**

1. Использование минималистичных базовых образов и удаление ненужного софта.

#### **2. Плюсы и минусы:**

- Ускорение развертывания и миграции.

- Ограничения в доступных инструментах и библиотеках.

Оптимизация размера данных является непрерывным процессом, требующим мониторинга и постоянного адаптирования под изменяющиеся условия. Успешная оптимизация может значительно улучшить производительность и уменьшить задержки, делая взаимодействие с системой более комфортным для конечного пользователя. Старайтесь мыслить оптимизацией на всех уровнях и не допускайте такого, чтобы оптимизация в одном месте ухудшало другое.