

## # Домашнее задание 05: Оптимизация производительности через кеширование и rate limiting

**Цель работы:** Получить практические навыки проектирования систем с учетом производительности, реализации кеширования и rate limiting.

### ## Задание

Для своего варианта задания выполните следующие задачи:

#### 1. Анализ производительности

- Изучите выбранный вариант задания
- Определите операции, которые будут выполняться часто (hot paths)
- Определите операции, которые могут быть медленными (обращения к БД, внешние API)
- Определите требования к производительности (время отклика, пропускная способность)

#### 2. Проектирование стратегии кеширования

- Определите, какие данные можно кешировать:
  - Часто читаемые данные
  - Результаты сложных вычислений
  - Данные, которые редко изменяются
- Выберите стратегию кеширования для каждого типа данных:
  - Cache-Aside (Lazy Loading)
  - Read-Through
  - Write-Through
  - Write-Back (Write-Behind)
- Определите TTL (Time To Live) для кешируемых данных
- Определите стратегию инвалидации кеша

#### 3. Реализация кеширования

- Реализуйте простое кеширование в вашем REST API (можно использовать Redis)
- Примените кеширование минимум для 2 endpoints
- Реализуйте инвалидацию кеша при обновлении данных

#### 4. Проектирование rate limiting

- Определите, какие endpoints требуют rate limiting
- Выберите алгоритм rate limiting для каждого endpoint:
  - Token Bucket
  - Leaking Bucket
  - Fixed Window Counter
  - Sliding Window Log/Counter
- Определите лимиты (например, 100 запросов в минуту для обычных пользователей, 1000 для премиум)

#### 5. Реализация rate limiting

- Реализуйте простой rate limiting для минимум одного endpoint
- Используйте выбранный алгоритм
- Возвращайте правильные HTTP статус-коды (429 Too Many Requests)

- Добавьте заголовки с информацией о лимитах (X-RateLimit-Limit, X-RateLimit-Remaining, X-RateLimit-Reset)

## 6. Анализ производительности

- Опишите, как кеширование и rate limiting улучшают производительность системы
- Определите метрики для мониторинга производительности
- Опишите, как измерить эффективность кеширования (hit rate)

## ## Результат

Результат должен быть оформлен в виде следующих файлов, размещенных в вашем GitHub репозитории:

- ``performance_design.md`` – описание стратегии кеширования и rate limiting
- Исходный код с реализацией кеширования и/или rate limiting
- ``README.md`` – описание проекта и реализованных оптимизаций
- Dockerfile и docker-compose.yaml для запуска приложения

## \*\*Критерии оценки:\*\*

- Обоснованность выбора стратегий кеширования
- Корректность проектирования rate limiting
- Качество реализации (если реализовано)
- Анализ влияния на производительность
- Качество документации

# Варианты

## # 1 Социальная сеть <https://facebook.com/>

Приложение должно содержать следующие данные:

- Пользователь
- Стена
- Сообщения чата (PtP)

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Добавление записи на стену
- Загрузка стены пользователя
- Отправка сообщения пользователю
- Получение списка сообщения для пользователя

## # 2 Магазин <https://ozon.ru/>

Приложение должно содержать следующие данные:

- Пользователь
- Товар
- Корзина

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание товара
- Получение списка товаров
- Добавление товара в корзину
- Получение корзины для пользователя

## # 3 Сайт конференции <https://www.eventboost.com/ru-RU/>

Приложение должно содержать следующие данные:

- Пользователь
- Доклад
- Конференция

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание доклада
- Получение списка всех докладов
- Добавление доклада в конференцию
- Получение списка докладов в конференции

#### # 4 Сайт заказа услуг <https://profi.ru/>

Приложение должно содержать следующие данные:

- Пользователь
- Услуга
- Заказ

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание услуги
- Получение списка услуг
- Добавление услуг в заказ
- Получение заказа для пользователя

#### # 5 Мессенджер <https://slack.com/intl/en-gb/>

Приложение должно содержать следующие данные:

- Пользователь
- Групповой чат
- PtP Чат

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание группового чата
- Добавление пользователя в чат
- Добавление сообщения в групповой чат
- Загрузка сообщений группового чата
- Отправка PtP сообщения пользователю
- Получение PtP списка сообщения для пользователя

#### # 6 Сервис доставки <https://www.cdek.ru/ru/>

Приложение должно содержать следующие данные:

- Пользователь
- Посылка
- Доставка

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание посылки
- Получение посылок пользователя
- Создание доставки от пользователя к пользователю
- Получение информации о доставке по получателю

- Получение информации о доставке по отправителю

#### **# 7 Сервис поиска попутчиков      <https://www.blablacar.ru/>**

Приложение должно содержать следующие данные:

- Пользователь
- Маршрут
- Поездка

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание маршрута
- Получение маршрутов пользователя
- Создание поездки
- Подключение пользователей к поездке
- Получение информации о поездке

#### **# 8 Управление проектами      <https://www.atlassian.com/ru/software/jira>**

Приложение должно содержать следующие данные:

- проект
- задача
- исполнитель

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание проекта
- Поиск проекта по имени
- Поиск всех проектов
- Создание задачи в проекте
- Получение всех задач в проекте
- Получение задачи по коду

#### **# 9 Электронная почта      <https://outlook.live.com/>**

Приложение должно содержать следующие данные:

- почтовая папка
- сообщение
- пользователь

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание новой почтовой папки
- Получение перечня всех папок
- Создание нового письма в папке
- Получение всех писем в папке

- Получение письма по коду

**# 10 Планирование задач <https://www.leadertask.ru/>**

Приложение должно содержать следующие данные:

- цель
- задача
- исполнитель

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание новой цели
- Получение списка всех целей
- Создание новой задачи на пути к цели
- Получение всех задач цели
- Изменение статуса задачи в цели

**# 11 Хранение файлов <https://360.yandex.ru/disk/>**

Приложение должно содержать следующие данные:

- папка
- файл
- пользователь

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание новой папки
- Получение списка всех папок
- Создание файла в папке
- Получение файла по имени
- Удаление файла
- Удаление папки

**# 12 Бюджетирование <https://about.coinkeeper.me/>**

Приложение должно содержать следующие данные:

- планируемый доход
- планируемый расход
- пользователь

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создать планируемый доход
- Получить перечень планируемых доходов
- Создать планируемый расход
- Получить перечень планируемых расходов

- Посчитать динамику бюджета за период

**# 13 Система бронирования отелей <https://www.booking.com/>**

Приложение должно содержать следующие данные:

- Пользователь
- Отель
- Бронирование

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание отеля
- Получение списка отелей
- Поиск отелей по городу
- Создание бронирования
- Получение бронирований пользователя
- Отмена бронирования

**# 14 Фитнес-трекер <https://www.myfitnesspal.com/>**

Приложение должно содержать следующие данные:

- Пользователь
- Тренировка
- Упражнение

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание упражнения
- Получение списка упражнений
- Создание тренировки
- Добавление упражнения в тренировку
- Получение истории тренировок пользователя
- Получение статистики тренировок за период

**# 15 Система управления библиотекой <https://www.librarything.com/>**

Приложение должно содержать следующие данные:

- Пользователь
- Книга
- Выдача

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Добавление книги в библиотеку
- Поиск книги по названию
- Поиск книги по автору
- Создание выдачи книги пользователю
- Получение списка выданных книг пользователя
- Возврат книги

**# 16 Система заказа такси <https://www.uber.com/>**

Приложение должно содержать следующие данные:

- Пользователь
- Водитель
- Поездка

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Регистрация водителя
- Создание заказа поездки
- Получение активных заказов
- Принятие заказа водителем
- Получение истории поездок пользователя
- Завершение поездки

**# 17 Система управления рестораном <https://www.opentable.com/>**

Приложение должно содержать следующие данные:

- Пользователь
- Блюдо
- Заказ

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Добавление блюда в меню
- Получение меню ресторана
- Поиск блюд по категории
- Создание заказа
- Добавление блюда в заказ
- Получение заказов пользователя
- Изменение статуса заказа

**# 18 Система управления складом <https://www.zoho.com/inventory/>**

Приложение должно содержать следующие данные:

- Пользователь
- Товар
- Поступление

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Добавление товара на склад
- Поиск товара по названию
- Получение остатков товаров
- Создание поступления товара
- Получение истории поступлений



- Списание товара со склада

**# 19 Система управления обучением** <https://www.moodle.org/>

Приложение должно содержать следующие данные:

- Пользователь
- Курс
- Урок

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание курса
- Получение списка курсов
- Добавление урока в курс
- Получение уроков курса
- Запись пользователя на курс
- Получение курсов пользователя
- Отметка о прохождении урока

**# 20 Система управления медицинскими записями** <https://www.epic.com/>

Приложение должно содержать следующие данные:

- Пользователь
- Пациент
- Запись

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Регистрация пациента
- Поиск пациента по ФИО
- Создание медицинской записи
- Добавление записи к пациенту
- Получение истории записей пациента
- Получение записи по коду

**# 21 Система управления арендой автомобилей** <https://www.hertz.com/>

Приложение должно содержать следующие данные:

- Пользователь
- Автомобиль
- Аренда

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Добавление автомобиля в парк
- Получение списка доступных автомобилей
- Поиск автомобилей по классу
- Создание аренды
- Получение активных аренд пользователя
- Завершение аренды
- Получение истории аренд

**# 22 Система управления событиями** <https://www.eventbrite.com/>

Приложение должно содержать следующие данные:

- Пользователь
- Событие
- Участник

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание события
- Получение списка событий
- Поиск событий по дате
- Регистрация пользователя на событие
- Получение участников события
- Получение событий пользователя
- Отмена регистрации на событие

**# 23 Система управления рецептами** <https://www.allrecipes.com/>

Приложение должно содержать следующие данные:

- Пользователь
- Рецепт
- Ингредиент

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Создание рецепта
- Получение списка рецептов
- Поиск рецептов по названию
- Добавление ингредиента в рецепт
- Получение ингредиентов рецепта
- Получение рецептов пользователя
- Добавление рецепта в избранное

**# 24 Система управления недвижимостью** <https://www.zillow.com/>

Приложение должно содержать следующие данные:

- Пользователь
- Объект недвижимости
- Просмотр

Реализовать API:

- Создание нового пользователя
- Поиск пользователя по логину
- Поиск пользователя по маске имя и фамилии
- Добавление объекта недвижимости
- Поиск объектов по городу
- Поиск объектов по цене
- Запись на просмотр объекта
- Получение записей на просмотр объекта
- Получение объектов пользователя
- Изменение статуса объекта

