НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

Системный дизайн современных приложений

Лекция №6
Глубокое погружение в систему

Организация

Занятия
31.05: Лекция №6 - Deep Dive
07.06: Лекция №7 - Итоги System Design Interview
14.06: Занятие №8 - Практика и Mock Interview

Формула
ДЗ*0.8 + SDD*0.1 + Посещение*0.1

ДЗ (дедлайн всех: 14 июня)
№1: ФТ, НФТ, ограничения, расчет нагрузки
№2: Интеграции
№3: БД
№4: Улучшение системы

Основные аспекты SD

1. Масштабируемость
2. Производительность
3. Надежность
   a. Отказоустойчивость
   b. Доступность
4. Безопасность
5. Адаптивность
6. Управляемость и мониторинг
7. Интеграции

Повышение отзывчивости

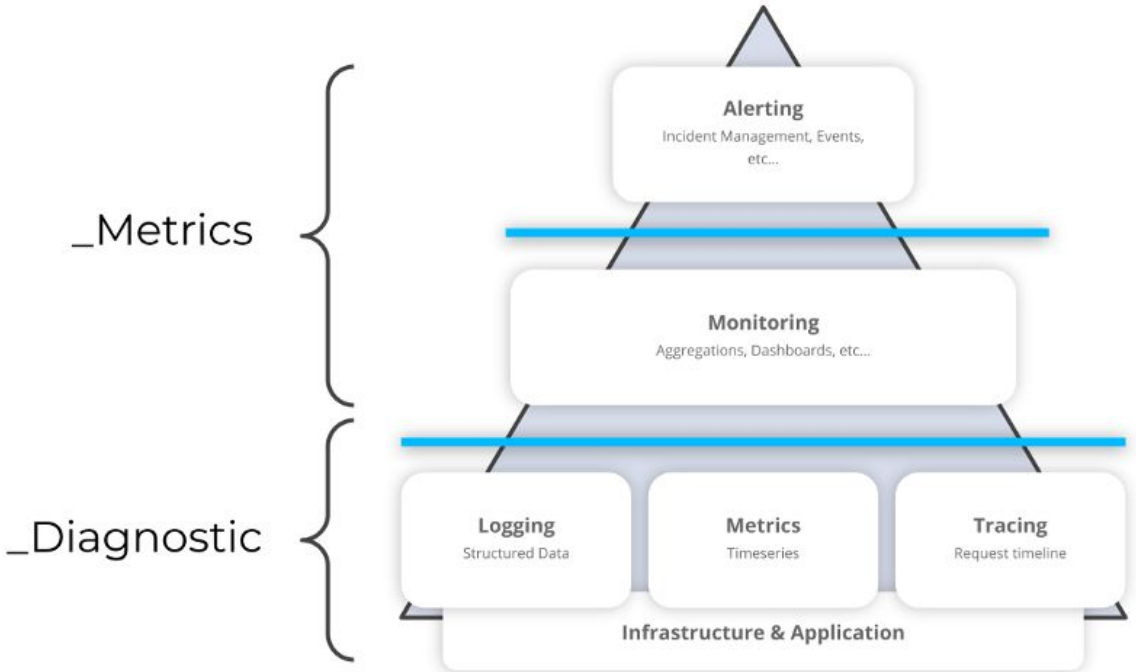
Проблематика
На первых парах все летало, пользователи были довольны
С масштабированием бизнес начал терять деньги от плохой производительности

Что делать
Анализ узких мест (bottleneck)

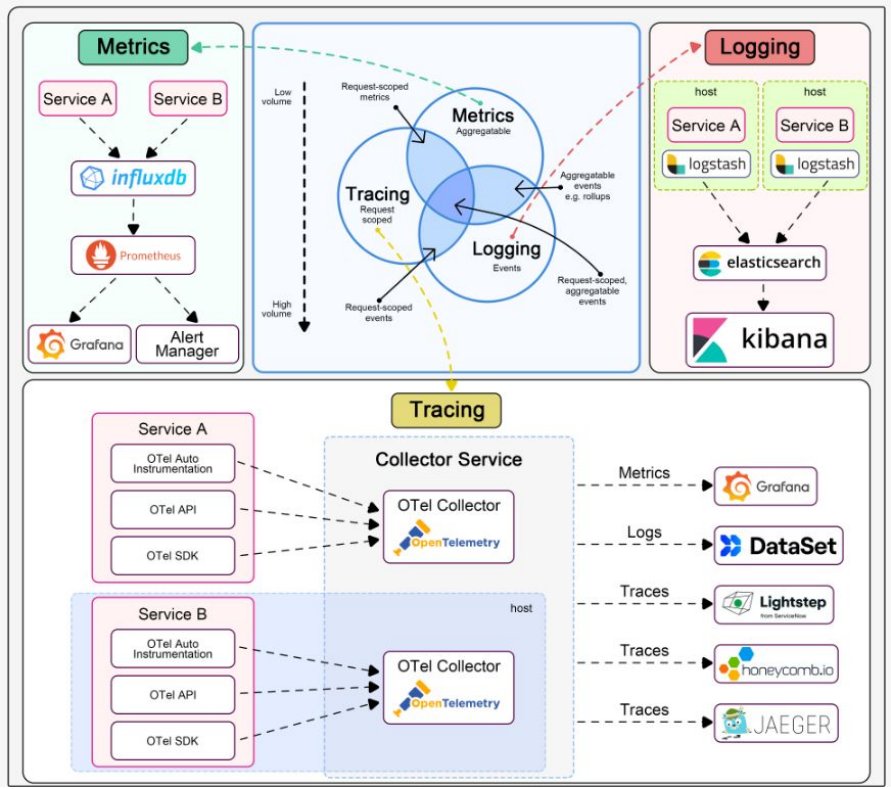
Цель
Снизить задержку (latency), увеличить пропускную способность (throughput)

Технические решения
1. Кэширование
2. CDN
3. Индексирование БД
4. Ограничитель запросов

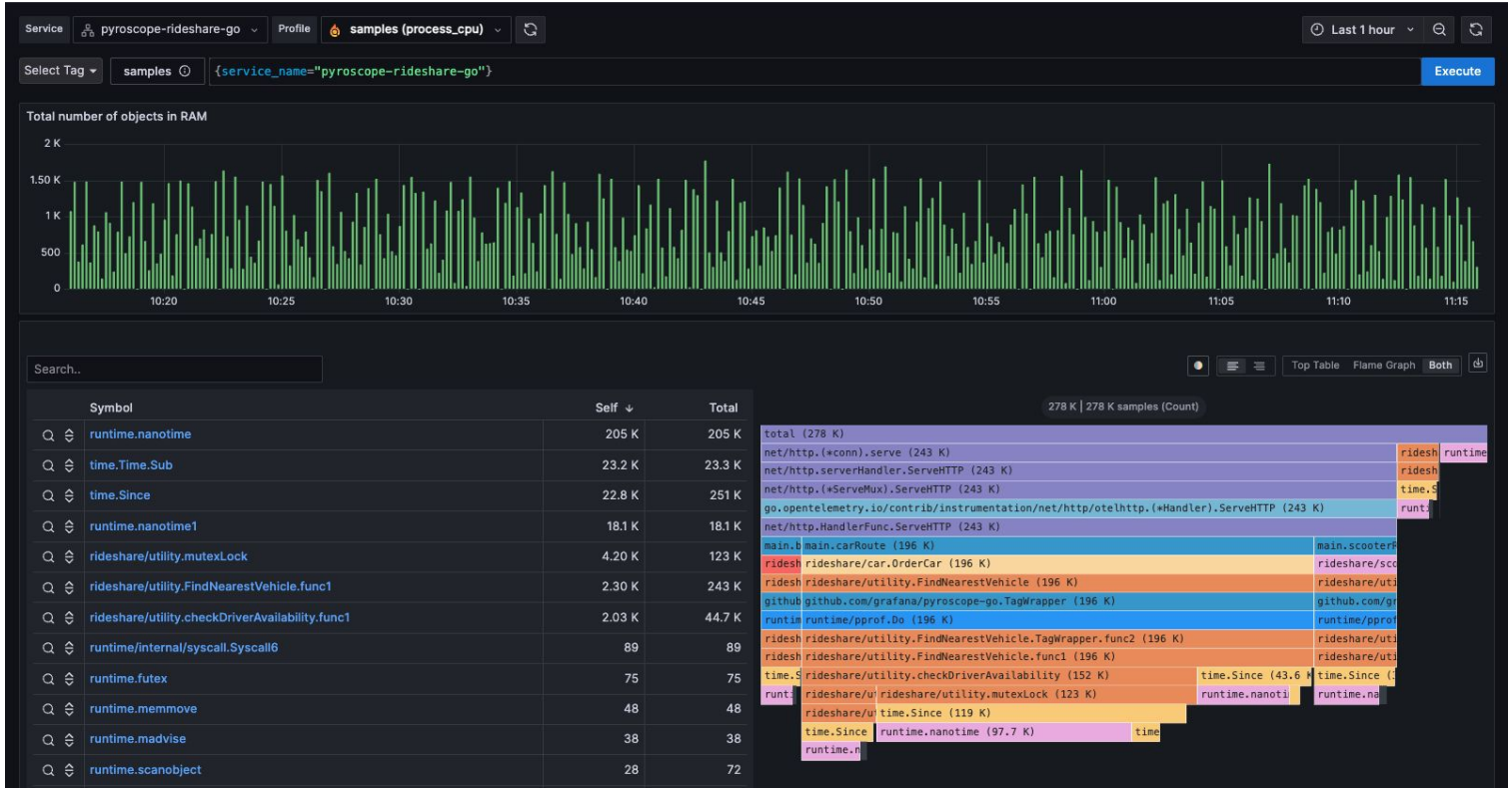
Как выявить узкие места



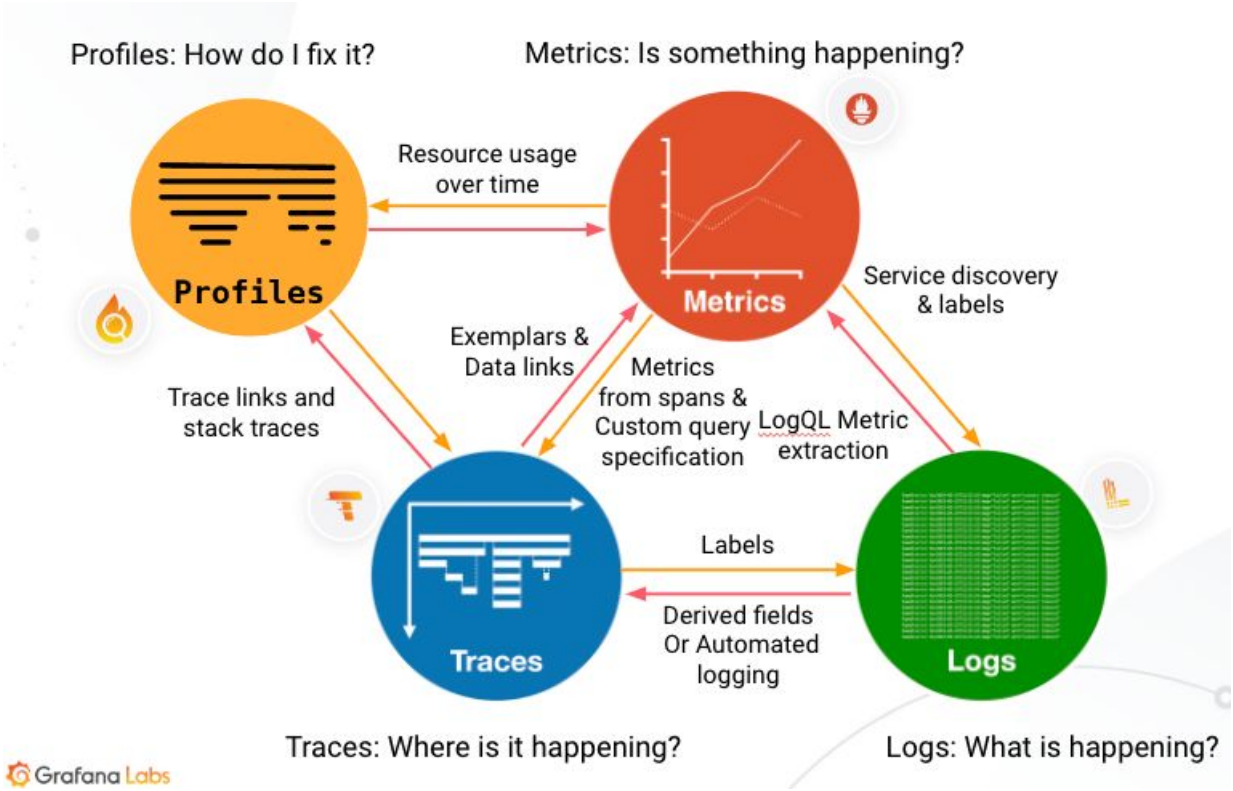
Логи, метрики, трейсы



Continuous Profiling



Комбинация


Profiles: How do I fix it?
Metrics: Is something happening?
Resource usage over time
Profiles
Metrics
Service discovery & labels
Exemplars & Data links
Metrics from spans & Custom query specification
LogQL Metric extraction
Trace links and stack traces
Labels
Derived fields Or Automated logging
Traces
Logs
Traces: Where is it happening?
Logs: What is happening?
Grafana Labs

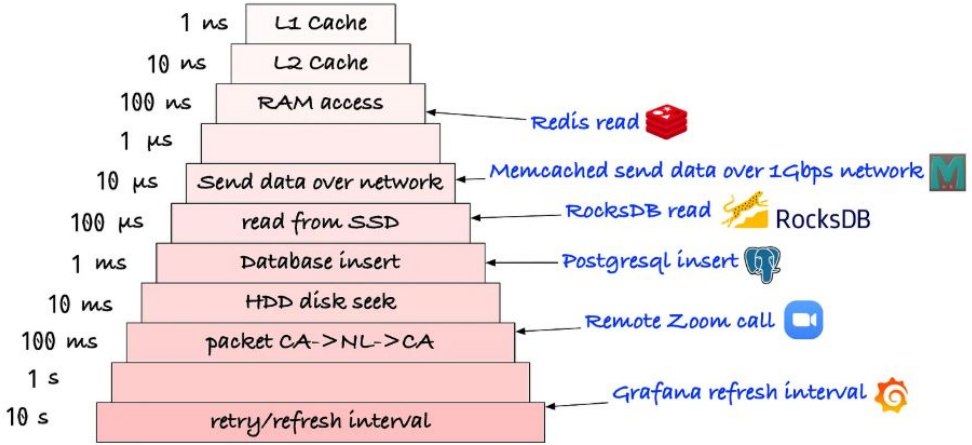
Законы производительности

Большинство из теории массового обслуживания

Закон Литтла (WIP-лимит)
Чем больше элементов находится в работе в определенное время, тем большее количество времени нужно на их завершение

Закон Амдала
Ускорение системы ограничено временем выполнения её самого медленного компонента

Принцип локальности
Данные, используемые вместе, должны храниться близко



Кэширование

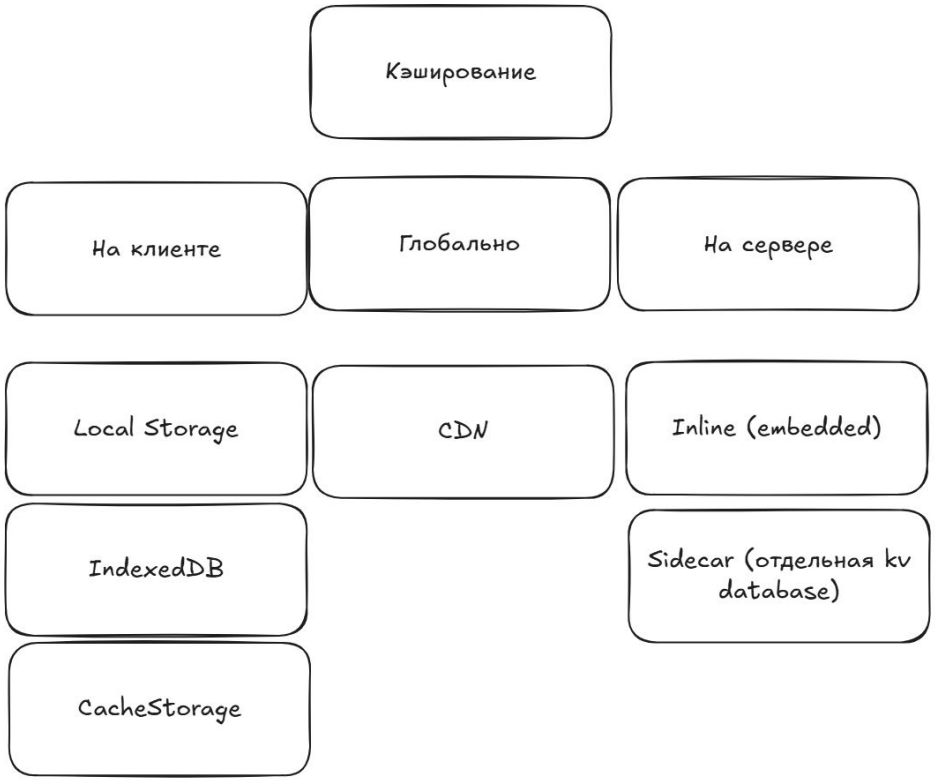
Способ хранения данных как можно ближе к месту их использования

Особенности

1. Если можно обойтись без кэширования, оно не нужно
2. Как правило, для этого используется быстродействующая память (RAM)
3. Более вероятно, что новые обращения будут происходить к недавним данным
4. Большая часть нагрузки приходится на малую часть запросов (принцип Парето 80 — 20)
5. Кэш представляет из себя базу типа ключ-значение
6. Каждая запись в кэше имеет TTL - “время жизни”, по истечении которого она удаляется

Примечание (помним про косты):
https://azrael.digipen.edu/~mmead/www/Courses/CS180/ram-hd-ssd-prices.html

Типы кэширования



Инвалидация кэша

Стоит всегда держать в голове
Рано или поздно данные в исходной БД изменятся и кэшированные данные придут в негодность

1. **Сквозная запись**: данные сразу записываются и в источник, и в кэш
Плюсы: гарантированная консистентность, минимизация потерь
Минусы: запись становится даже медленнее из-за двух операций

2. **Запись в обход**: данные записываются напрямую в источник
Плюсы: не нагружаем кэш записью невостребованных данных
Минусы: для недавних данных считывать нечего, идем в источник

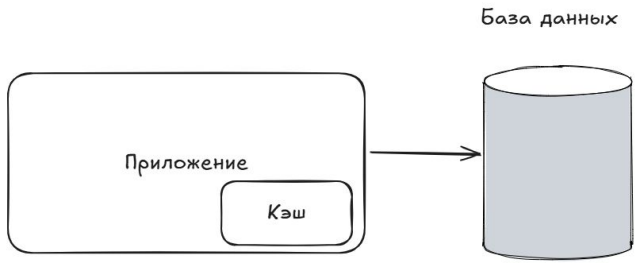
3. **Реверсивная запись**: данные сначала записываются в кэш
Плюсы: низкие задержки и высокая пропускная способность на запись
Минусы: можно потерять недавние данные, не продублированные в источник

Вытеснение данных

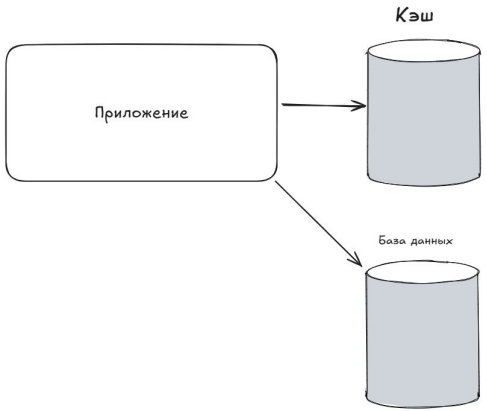
1. First In First Out (FIFO): удаляем самые давние записи несмотря на их популярность
2. Last In First Out (LIFO): удаляем самые свежие записи несмотря на их популярность
3. Least Recently Used (LRU): удаляем записи, не использованные дольше всего
4. Most Recently Used (MRU): удаляем самые недавние из использованных записей
5. Least Frequently Used (LFU): ведем счет обращений, удаляем самые непопулярные
6. Random Replacement (RR): удаляем случайно выбранные записи

Inline/Sidecar

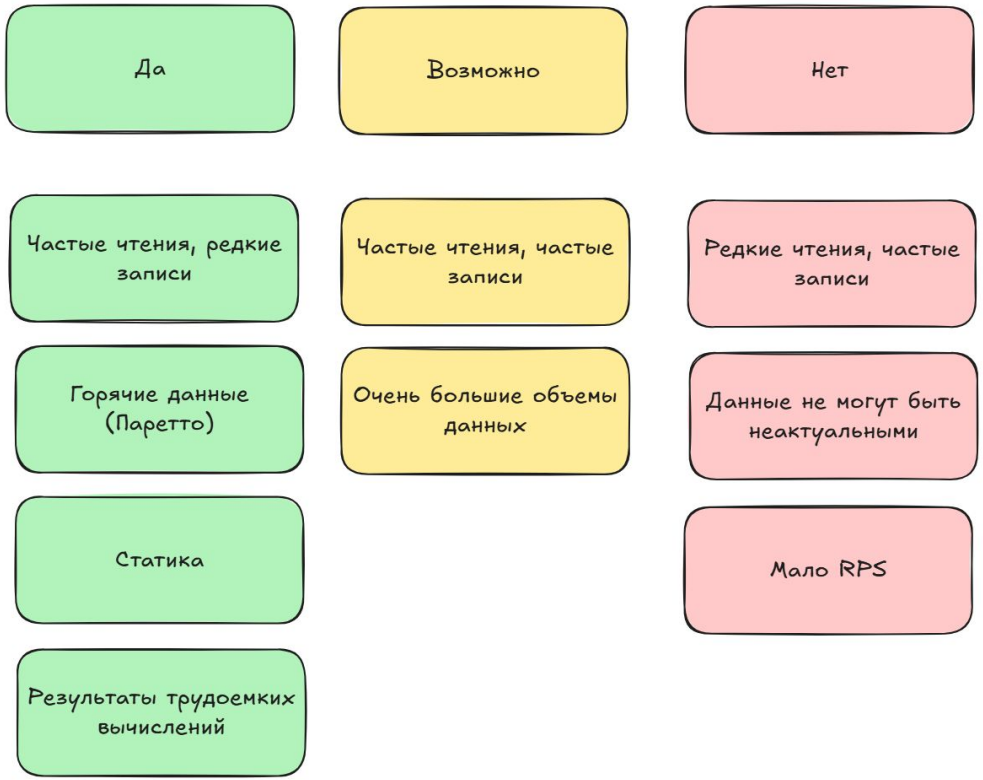
Inline:
В ОЗУ приложения



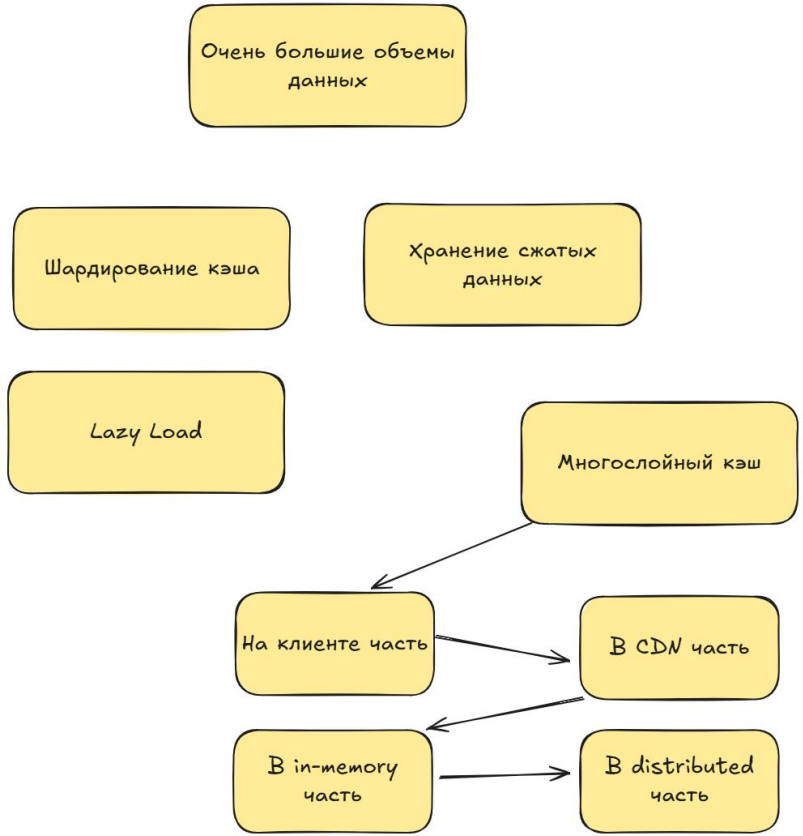
Sidecar:
Отдельная kv-база данных



Кэширование: целесообразность



Кэширование: целесообразность



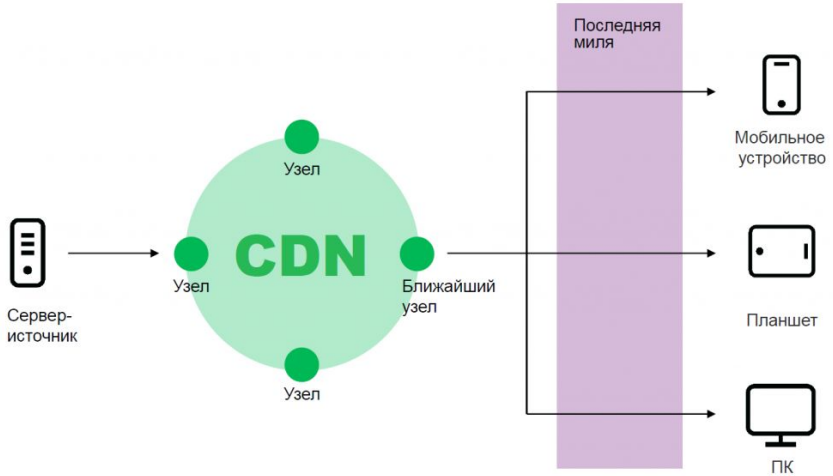
CDN

Глобальная сеть серверов, кэширующих статический контент ближе к пользователям для ускорения загрузки

Как работает?

1. Пользователь запрашивает контент.
2. CDN перенаправляет запрос на ближайший сервер (edge-узел).
3. Если контент есть в кэше — отдаётся мгновенно, если нет — загружается с origin-сервера и кэшируется.

Пример: Cloudflare



Индексирование БД

Цель
Ускоряет поиск нужных записей в таблице

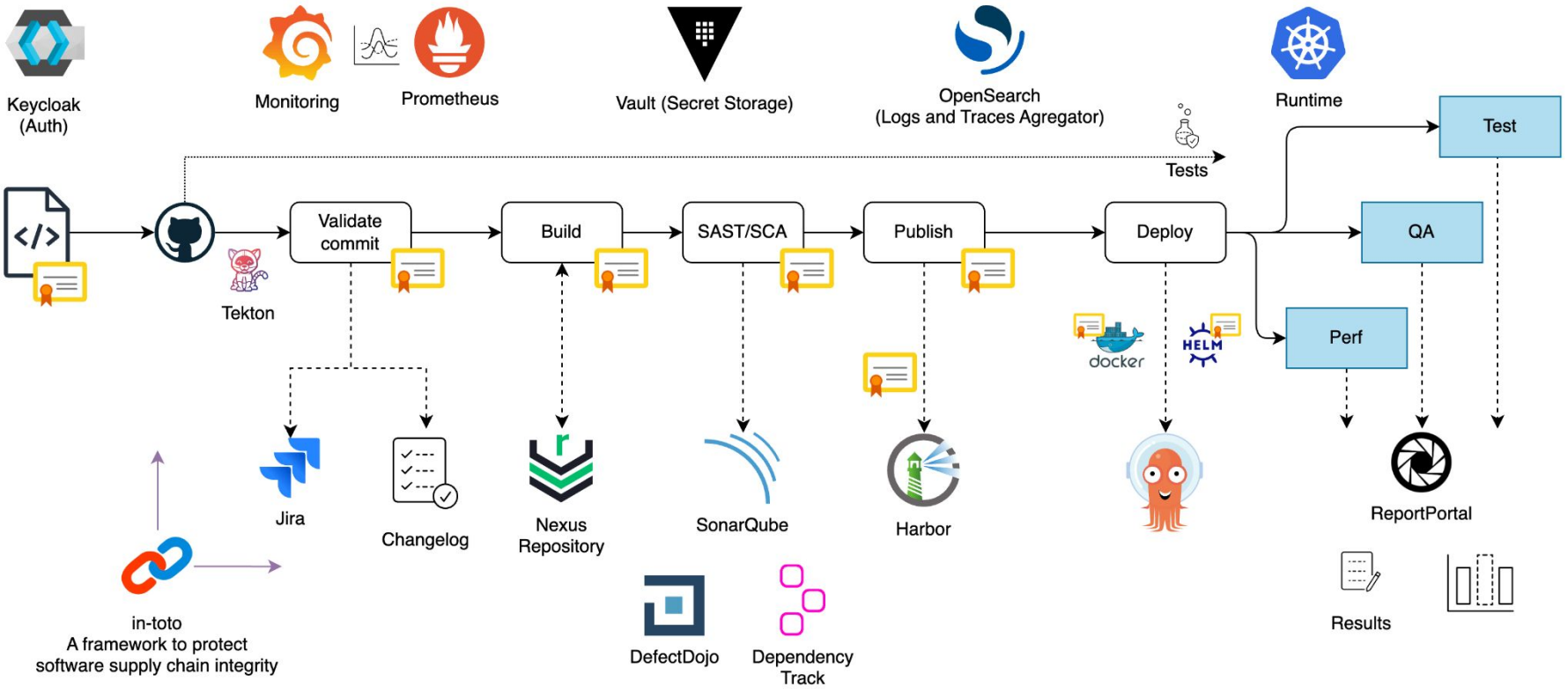
Механизм
Ищет нужные записи на основе значений ключа в индексе

Плюсы
1. Увеличение скорости поиска записей по ключу в индексе (O(N) -> O(logN))
2. Применяем, когда поиск по многочисленным или распределенным записям в БД становится bottleneck

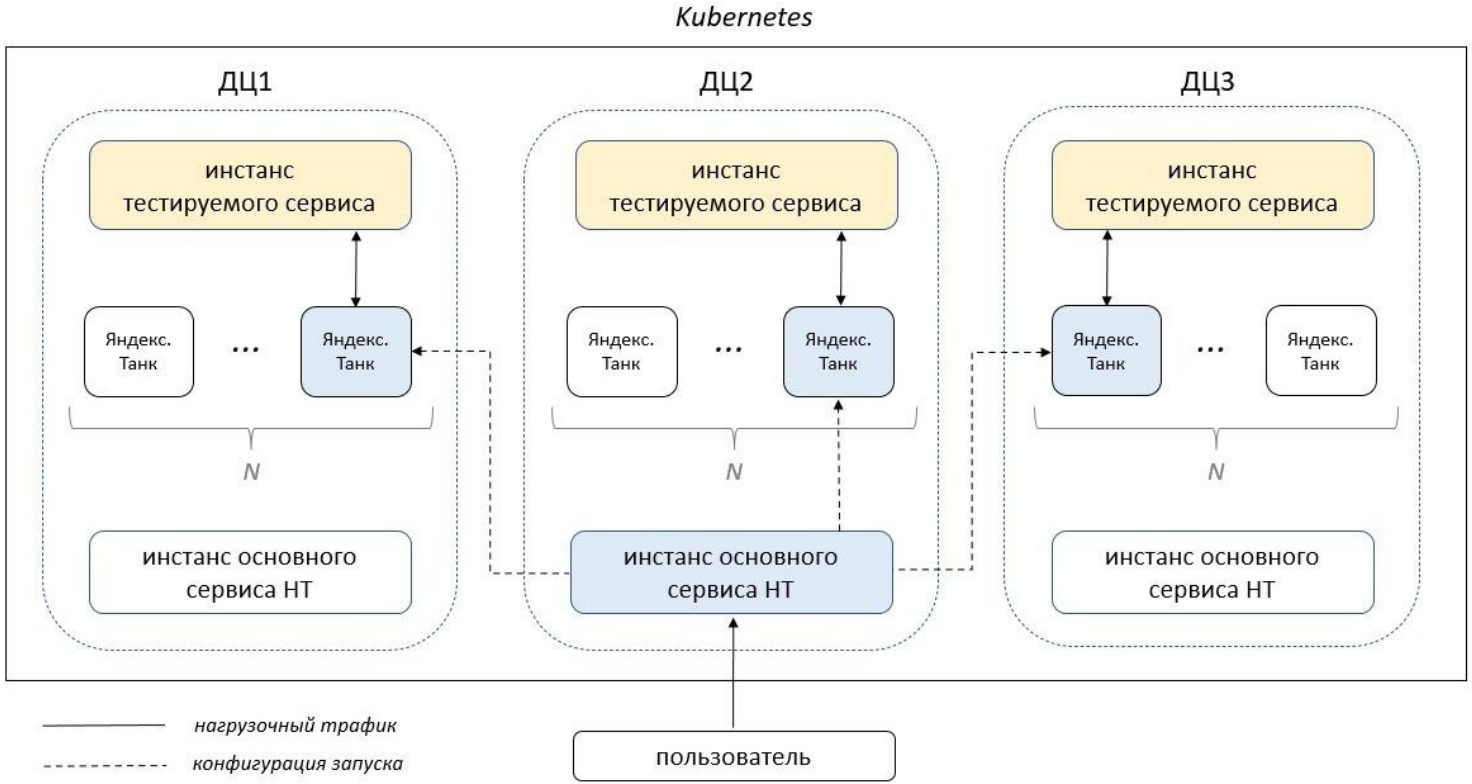
Минусы
1. При обновлении данных нужно обновить не только их, но и записи в индексе.
2. Не применяем, если операций на запись значительно больше операций на чтение

Что почитать
1. LSTM-деревья
2. B-tree
3. Gist
4. Полнотекстовый индекс

CI/CD


Keycloak
(Auth)
Monitoring
Prometheus
Vault (Secret Storage)
OpenSearch
(Logs and Traces Agregator)
Runtime
Tests
Test
Validate
commit
Build
SAST/SCA
Publish
Deploy
QA
Perf
Tekton
docker
HELM
Jira
Changelog
Nexus
Repository
SonarQube
Harbor
ReportPortal
in-toto
A framework to protect
software supply chain integrity
DefectDojo
Dependency
Track
Results

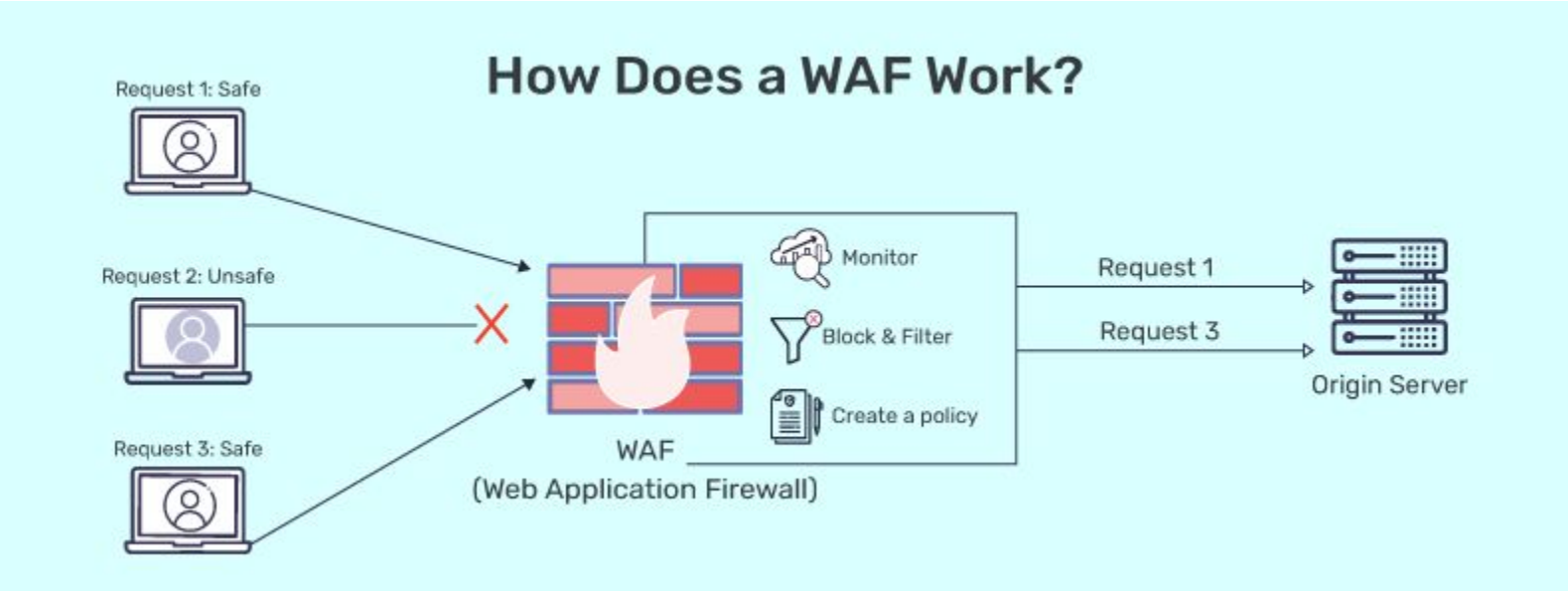
МультиЦОД


Kubernetes
ДЦ1
ДЦ2
ДЦ3
инстанс тестируемого сервиса
инстанс тестируемого сервиса
инстанс тестируемого сервиса
Яндекс. Танк
...
Яндекс. Танк
Яндекс. Танк
...
Яндекс. Танк
Яндекс. Танк
...
Яндекс. Танк
N
N
N
инстанс основного сервиса НТ
инстанс основного сервиса НТ
инстанс основного сервиса НТ
нагрузочный трафик
конфигурация запуска
пользователь

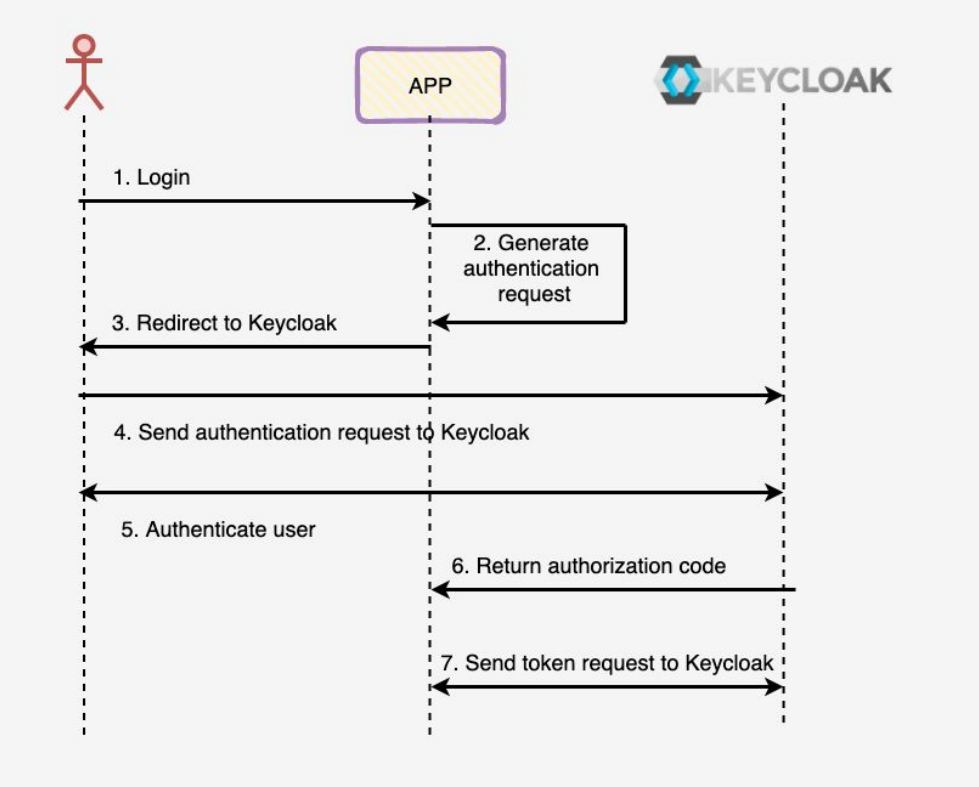
Безопасность

1. WAF
2. IdP
3. Firewall
4. Anti-DDoS
5. Quality Gates
    a. SAST
    b. DAST
6. Хранилище секретов

WAF


How Does a WAF Work?
Request 1: Safe
Request 2: Unsafe
Request 3: Safe
Monitor
Block & Filter
Create a policy
WAF
(Web Application Firewall)
Request 1
Request 3
Origin Server

Identity Provider



Система X-cursion

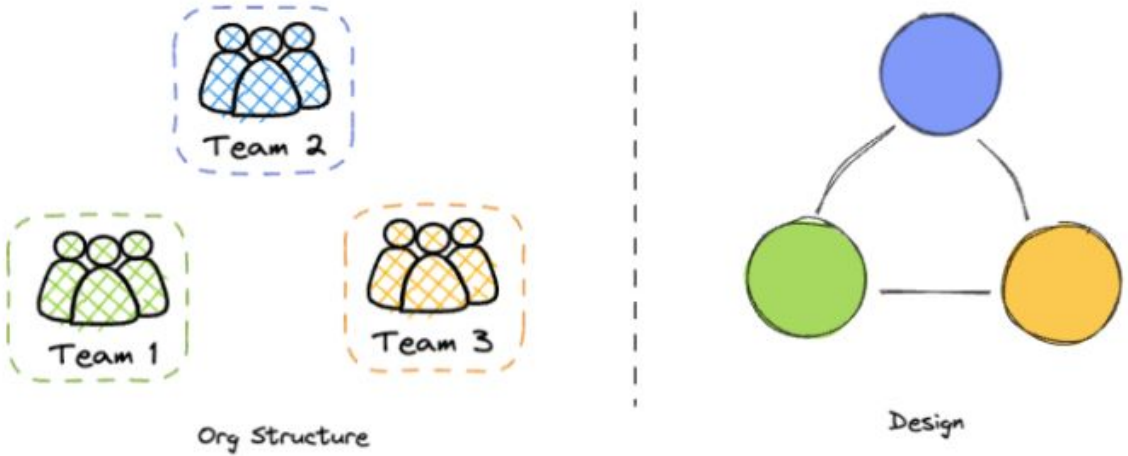
MVP


B2C
[Пользак]
Слушает аудио-экскурсии
ЯкП хочу слушать аудио с экскурсией
ЯкП хочу иметь аккаунт для сохранения экскурсий
ЯкП хочу искать экскурсии на карте и через поиск
ЯкП хочу удобно платить за экскурсии
B2C
[Автор]
Создает аудио-экскурсии
ЯкА хочу создать карточку и загрузить аудио-файл с экскурсией
ЯкП хочу иметь аккаунт для сохранения записей экскурсии
ЯкП хочу видеть аналитику по экскурсиям
ЯкП хочу удобно монетизировать экскурсии
B2B
[Локальный бизнес]
Покупает рекламу
ЯкЛБ хочу настроить таргетированную рекламу
ЯкЛБ хочу управлять рекламными компаниями

Закон Конвея

Формулировка

Организации проектируют системы, которые **копируют** структуру коммуникаций в этой организации



Домашка №4

Дополнительные компоненты

Доделать High Level Design, добавив в систему дополнительные компоненты и обосновав их целесообразность

System Design Document

SDD (System Design Document)

1. Вышлю шаблон
2. Сделать документ
3. Выложить на Github

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ