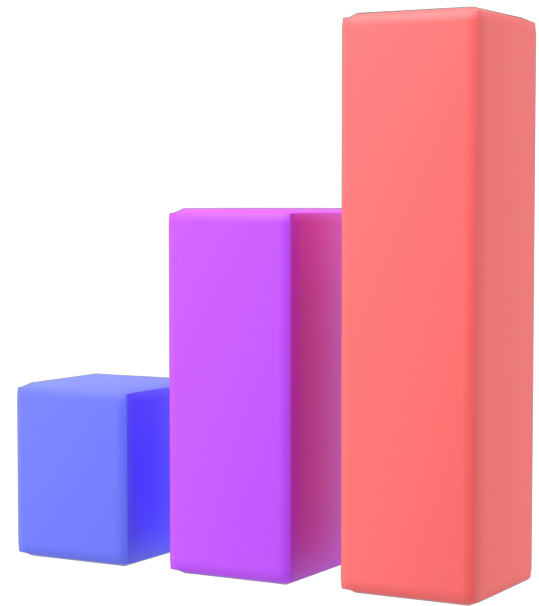


System Design

Основные термины и компоненты

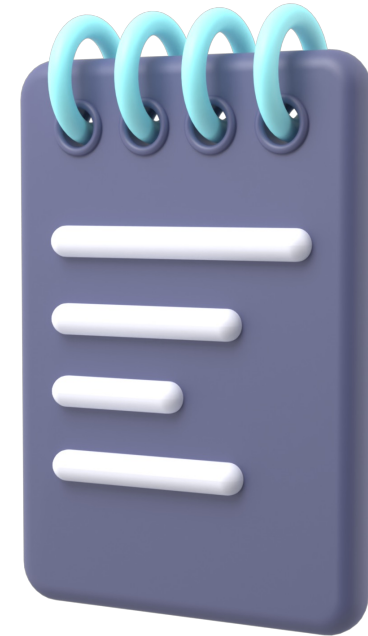


Проверь запись



Маршрут занятия

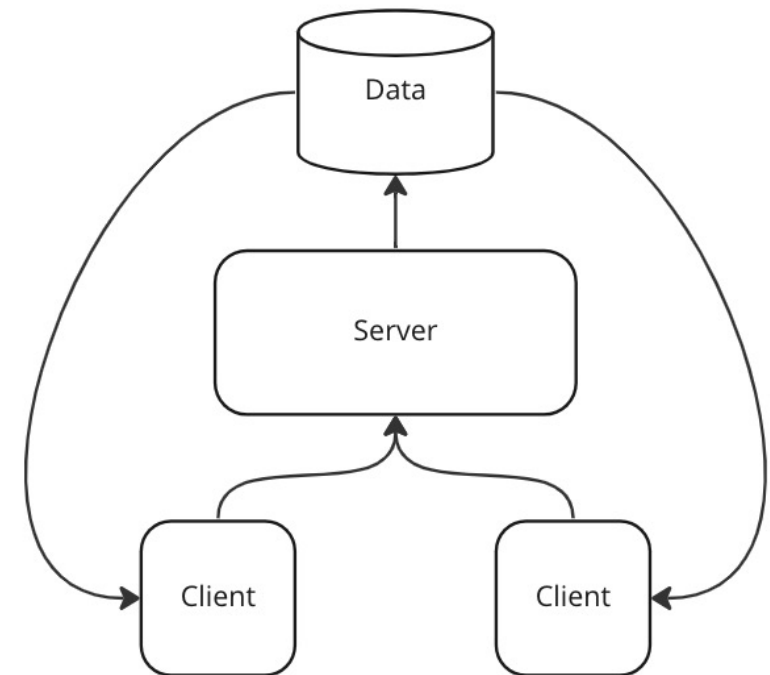
- Архитектуры информационных систем
- Основные критерии информационных систем
- Основные свойства информационных систем
- Архитектура бэкенда
- Балансировка нагрузки и проксирование
- Кэширование
- API
- Observability



Архитектуры ИС

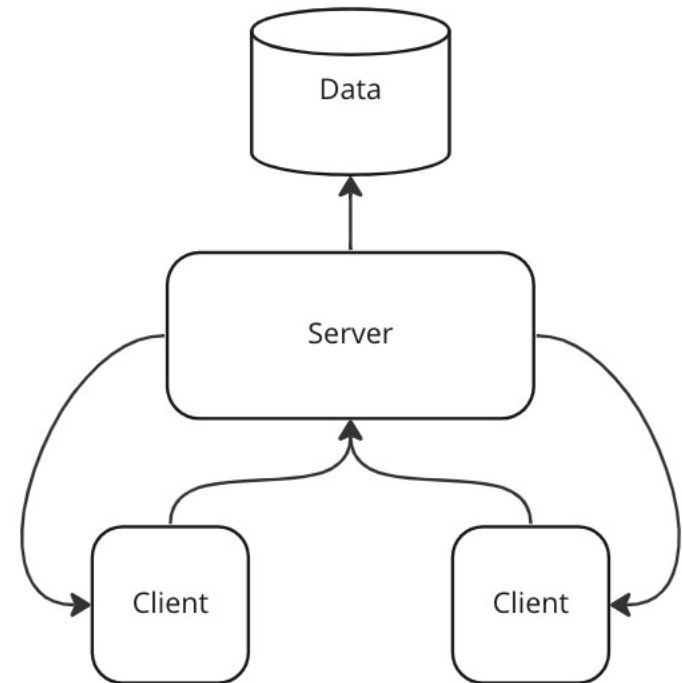
Файл-сервер

Файл-сервер только извлекает данные из файла или базы данных и передает их клиенту для дальнейшей обработки



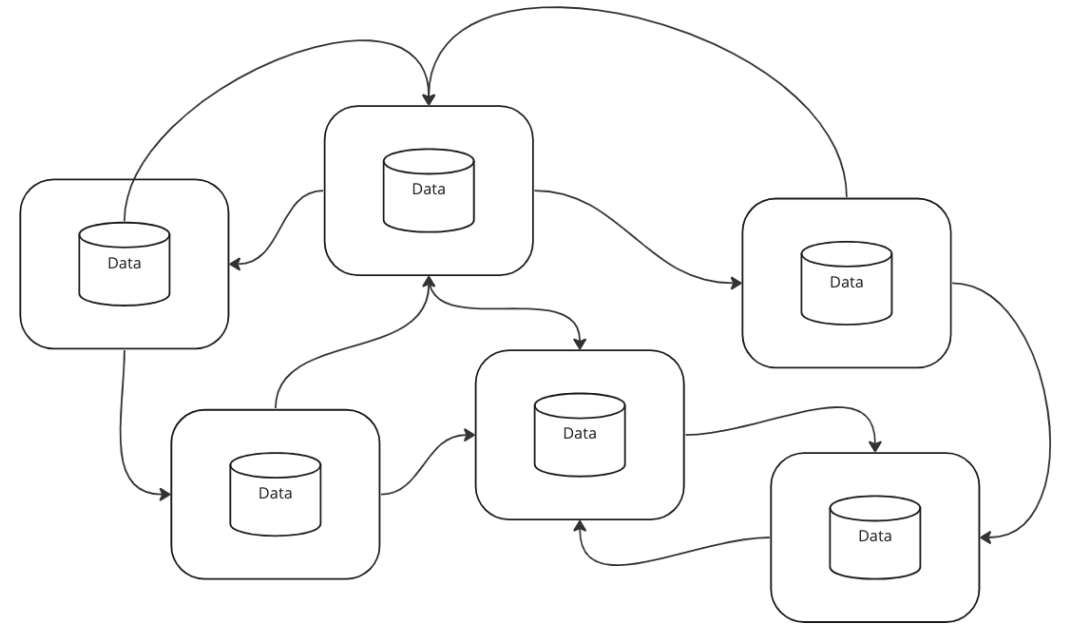
Клиент-сервер

Клиент-сервер извлекает данные из файла или базы данных, обрабатывает и затем передает результат клиенту



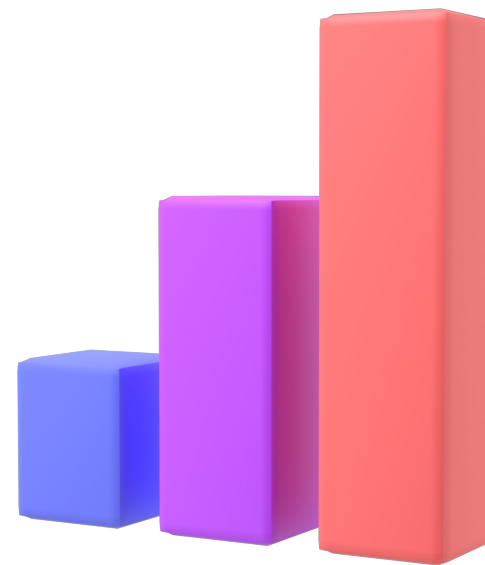
Peer to Peer (P2P)

Все узлы выполняют одинаковые функции – нет централизованного сервера



Резюме

- Файл-сервер умер
- Peer to Peer для блокчейна и торрентов
- Клиент-сервер для всего остального



FAQ:

Архитектуры ИС

- Файл-сервер
- Клиент-сервер
- Peer to Peer (P2P)



Основные критерии систем

Надежность

Система должна продолжать
работать корректно даже при
неблагоприятных обстоятельствах



Availability Level		Average Yearly Downtime
Conventional Server	99%	87 hours, 40 minutes
Public Cloud Service	99.5%	43 hours, 50 minutes
	99.9%	8 hours, 46 minutes
High-Availability Cluster	99.95%	4 hours, 23 minutes
Virtual Fault Tolerance	99.995%	26 minutes, 18 seconds
Continuous Availability	99.999%	5 minutes, 16 seconds
The Stratus Zone	99.9999%	31.6 seconds

SLA / SLO / SLI

- **SLA (Service Level Agreement)** – соглашение, которое вы заключаете со своими клиентами
- **SLO (Service Level Objectives)** - цели, которые команда должна решить, чтобы выполнить соглашение
- **SLI (Service Level Indicators)** – показатели системы

Google Drive

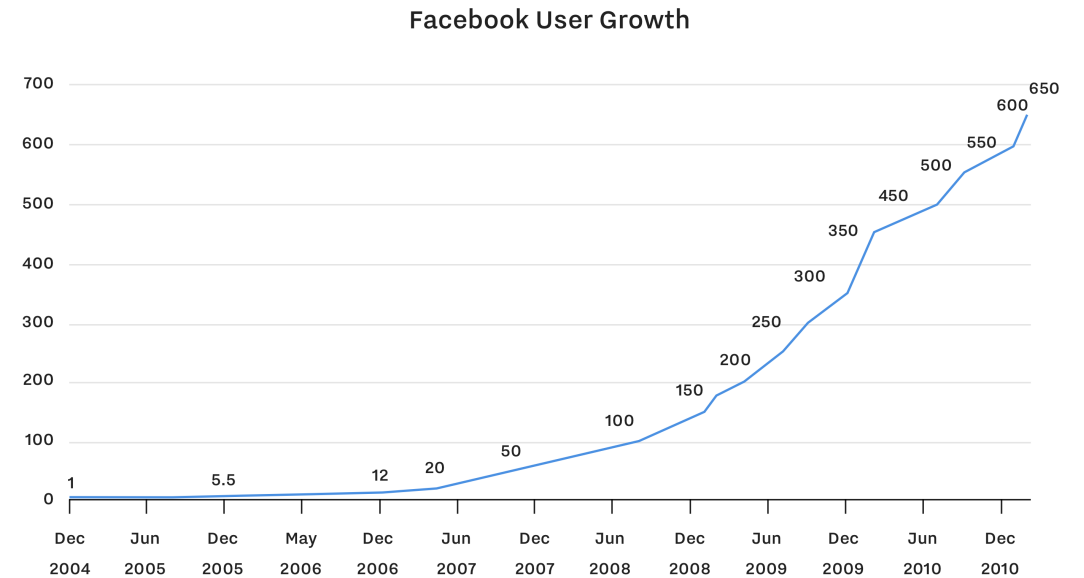
Covered Service	Monthly Uptime Percentage
Standard storage class in a multi-region or dual-region location of Cloud Storage	>= 99.95%
Standard storage class in a regional location of Cloud Storage; Nearline, Coldline, or Archive storage class in a multi-region or dual-region location of Cloud Storage	>= 99.9%
Nearline, Coldline, or Archive storage class in a regional location of Cloud Storage; Durable Reduced Availability storage class in any location of Cloud Storage	>= 99.0%

Google Drive

Monthly Uptime Percentage	Percentage of monthly bill for the Standard storage class in a multi-region or dual-region location of Cloud Storage that does not meet SLO that will be credited to Customer's future monthly bills
99.0% – < 99.95%	10%
95.0% – < 99.0%	25%
< 95.0%	50%

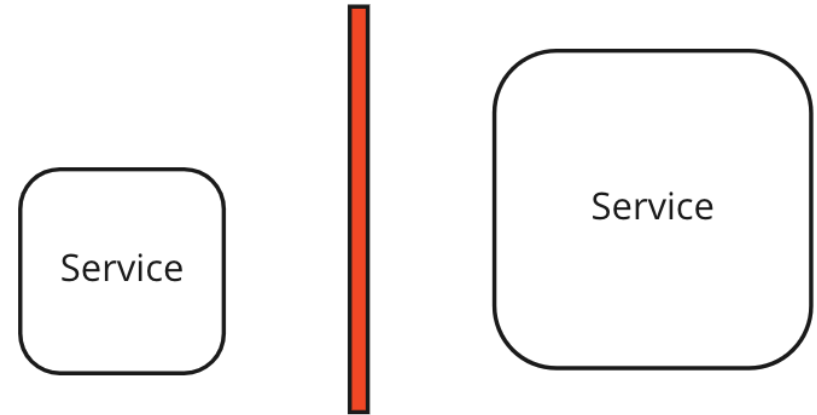
Масштабируемость

Должны быть предусмотрены разумные способы решения возникающих при росте системы проблем



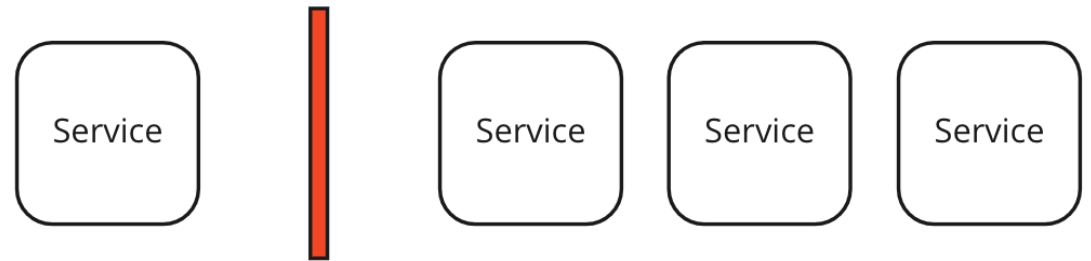
Вертикальное

Стоимость увеличения ресурсов растет не линейно, относительно их увеличения мощности (возможен downtime). Также невозможно бесконечно масштабироваться



Горизонтальное

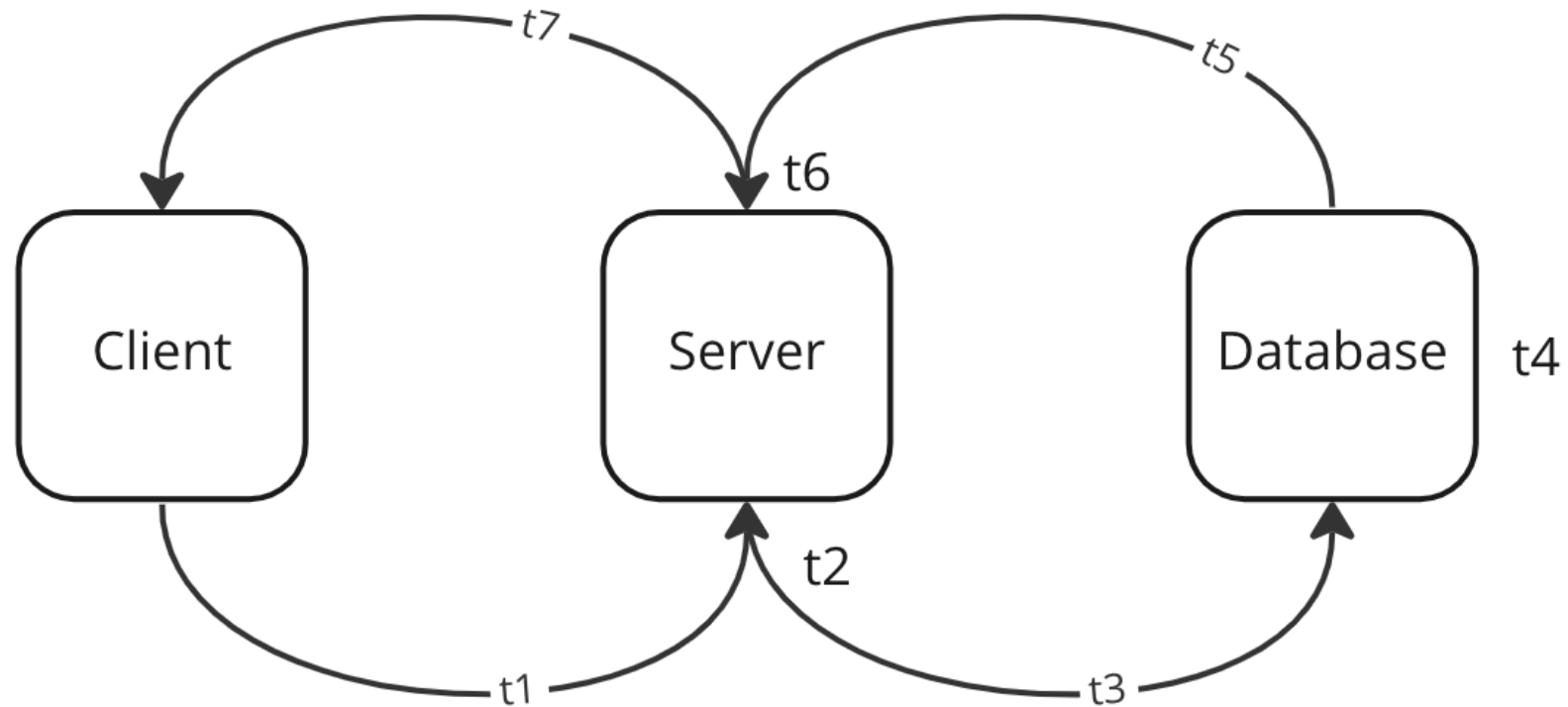
Чаще всего масштабирование
происходит без каких-либо проблем



Stateless and Statefull

Производительность

Latency и Response Time



Latency

Read 1M from RAM = 250 000 нс

Read 1M from SSD = 1 мс

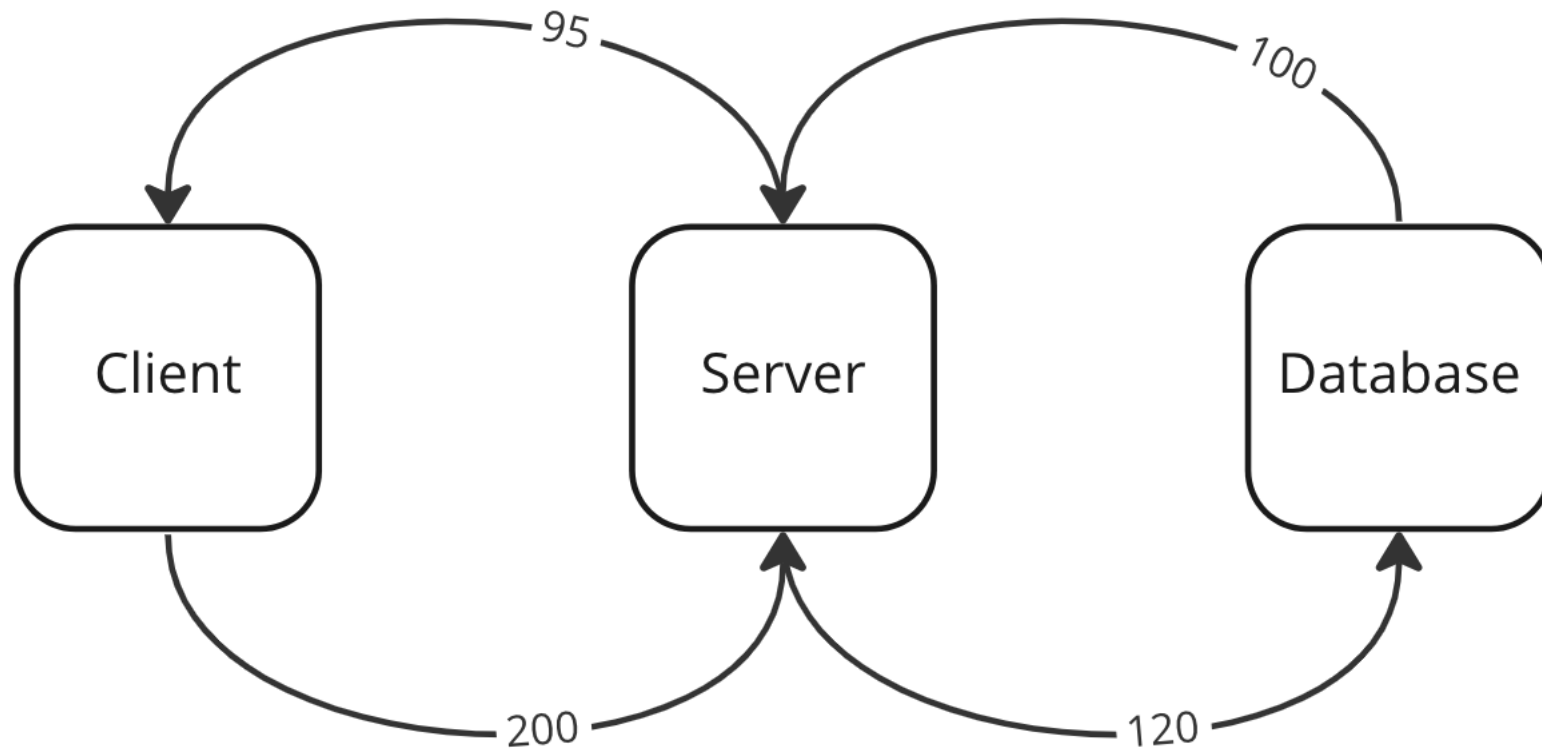
Read 1M from HDD = 20 мс



Low-latency приложения

Такие приложения надо с самого начала писать с оптимизациями, с прямой ориентацией на low-latency. Подход, когда пишут сначала просто работающее приложение, а потом оптимизируют его, здесь не работает

Throughput



Throughput

HDD = 200-300 МБ/с

SSD = 600-700 МБ/с

DDR3 = 17000 МБ/с



High-throughput приложения

Приложения, заточенные под максимально возможную пропускную способность – latency в этом случае часто пренебрегают

«Типичные» приложения

Нужен максимально возможный throughput по какому-то определенному latency

Удобство сопровождения

Необходимо обеспечить возможность
эффективной работы с системой
множеству различных людей



Удобство сопровождения

- Observability
- Улучшение процессов
- Дополнительный инструментарий

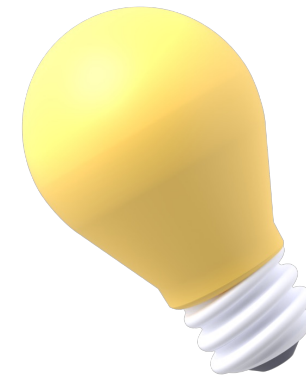
Безопасность

В 2020 году взломали Twitter-аккаунты
Билла Гейтса, Илона Маска, Барака
Обамы, Джеффа Безоса, Канье Уэста и
других известных личностей в США



Безопасность

- Передача данных в открытом виде
- Транспортное шифрование
- Сквозное шифрование



FAQ:

Основные критерии

- Надежность
- Масштабируемость
- Производительность
- Удобство сопровождения
- Безопасность



Основные свойства ИС

Что из этого HighLoad?

20 RPS или 20 000 RPS

Data-intensive

- Нужно сохранять большие данные
- Нужно запоминать результаты ресурсоемких операций
- Нужно предоставлять пользователям возможность искать или фильтровать данные

Compute-intensive

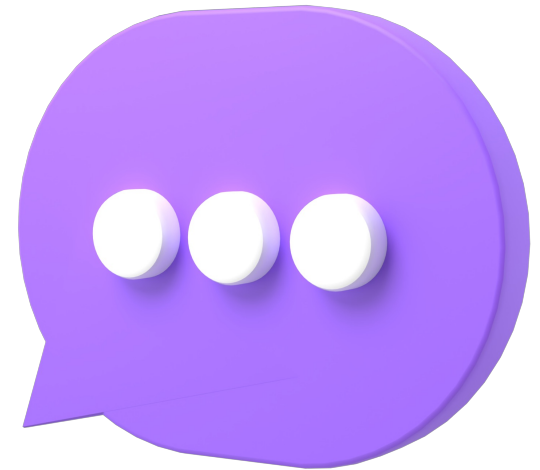
- Нужно делать много вычислительных операций
- Нужно «перемалывать» большие объемы данных

Read / write ratio

FAQ:

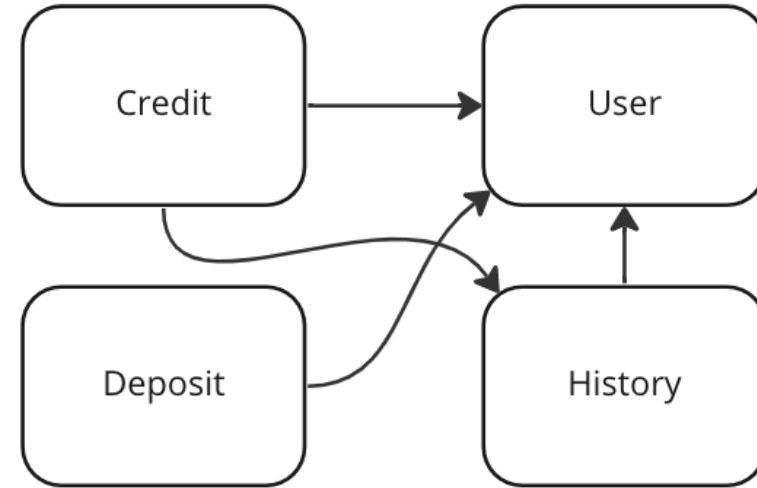
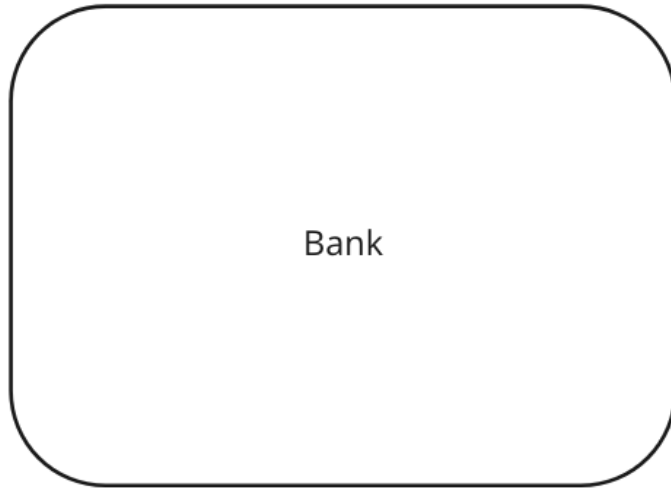
Основные свойства

- Read / write ratio
- Data / compute intensive



Архитектура бэкенда

Монолит VS Микросервисы



Микросервисы

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

Микросервисы

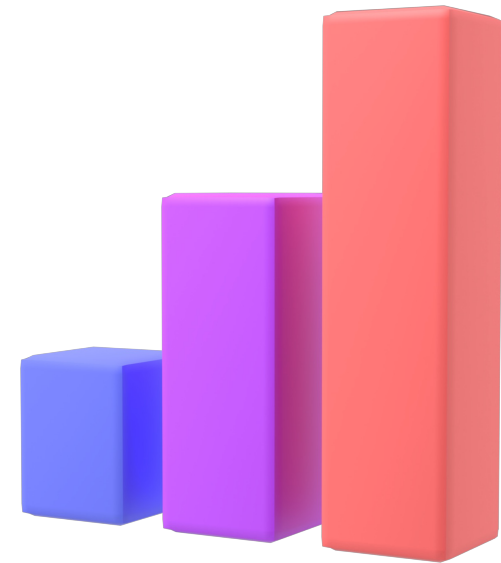
- + Независимые релизы и разработка
- + Независимая масштабируемость
- + Независимая деградация
- + Возможность пробовать новые технологии
- Зоопарк технологий
- Сетевой вызов отвалится вероятнее, чем внутренний
- Распределенность и транзакционность
- Удаленные вызовы дороже локального исполнения
- Понимание всего контекста запроса
- Сложность тестирования всей системы

The Bezos Mandate

1. All teams will henceforth expose their data and functionality through service interfaces.
2. Teams must communicate with each other through these interfaces.
3. There will be no other form of interprocess communication allowed: no direct linking, no direct reads of another team's data store, no shared-memory model, no back-doors whatsoever. The only communication allowed is via service interface calls over the network.
4. It doesn't matter what technology they use. HTTP, Corba, Pubsub, custom protocols — doesn't matter.
5. All service interfaces, without exception, must be designed from the ground up to be externalizable. That is to say, the team must plan and design to be able to expose the interface to developers in the outside world. No exceptions.
6. Anyone who doesn't do this will be fired.
7. Thank you; have a nice day!

Резюме

- Монолиты для стартапов и low-latency
- Микросервисы для всего остального



FAQ:

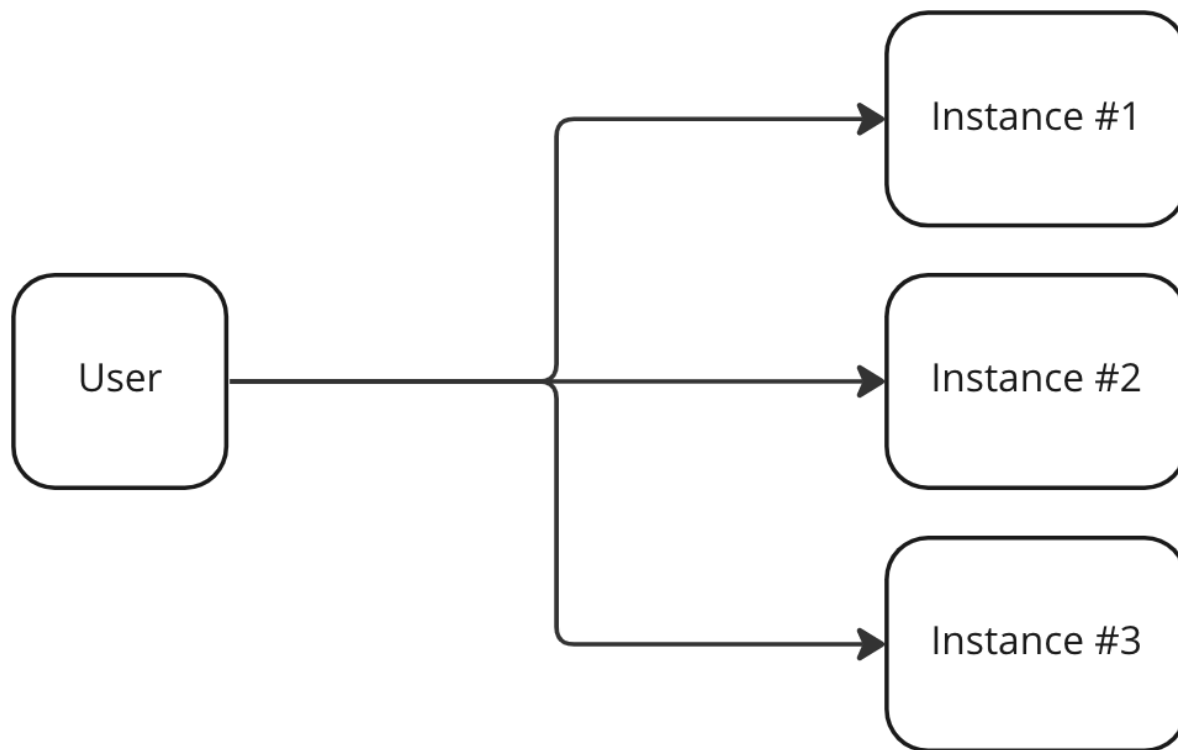
Архитектура бэкенда

- Монолитная
- Микросервисная

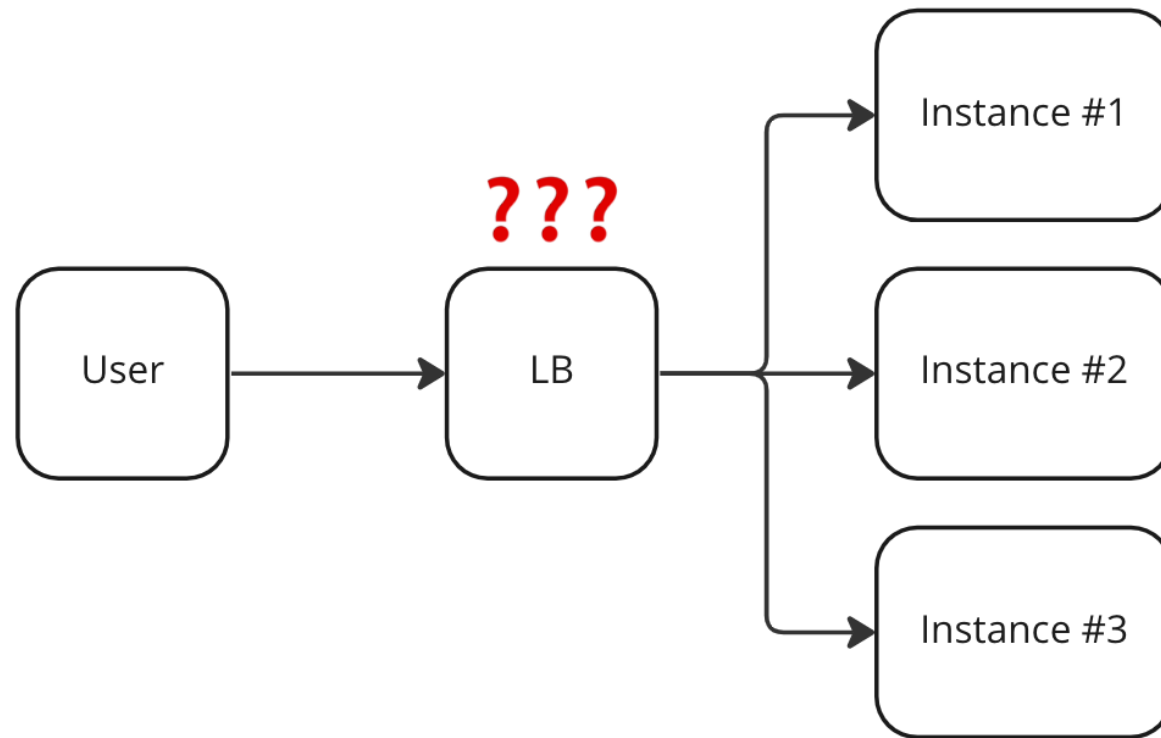


Балансировка нагрузки

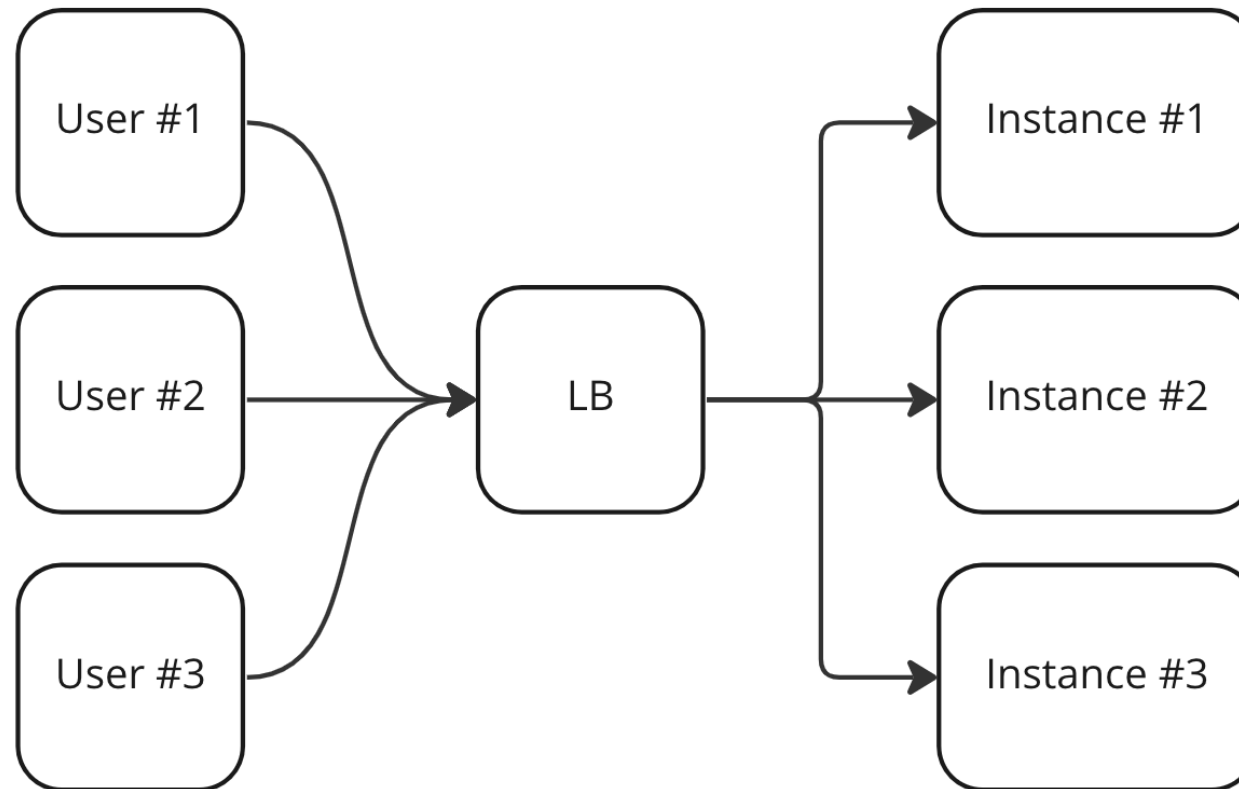
Клиентская балансировка



Серверная балансировка

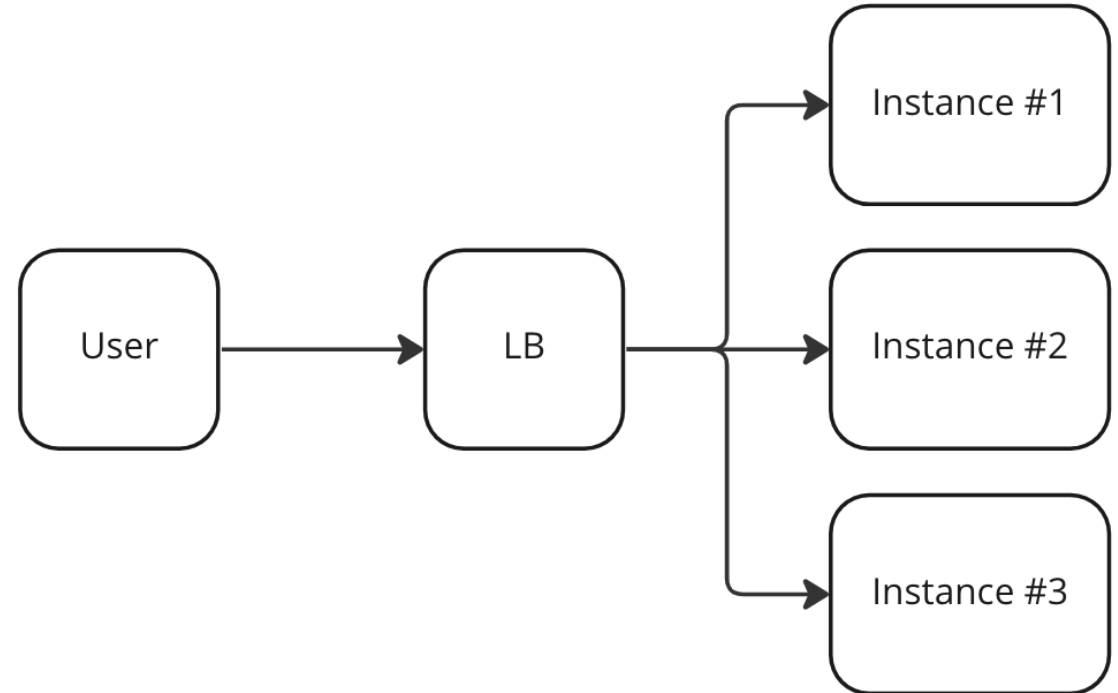


Random



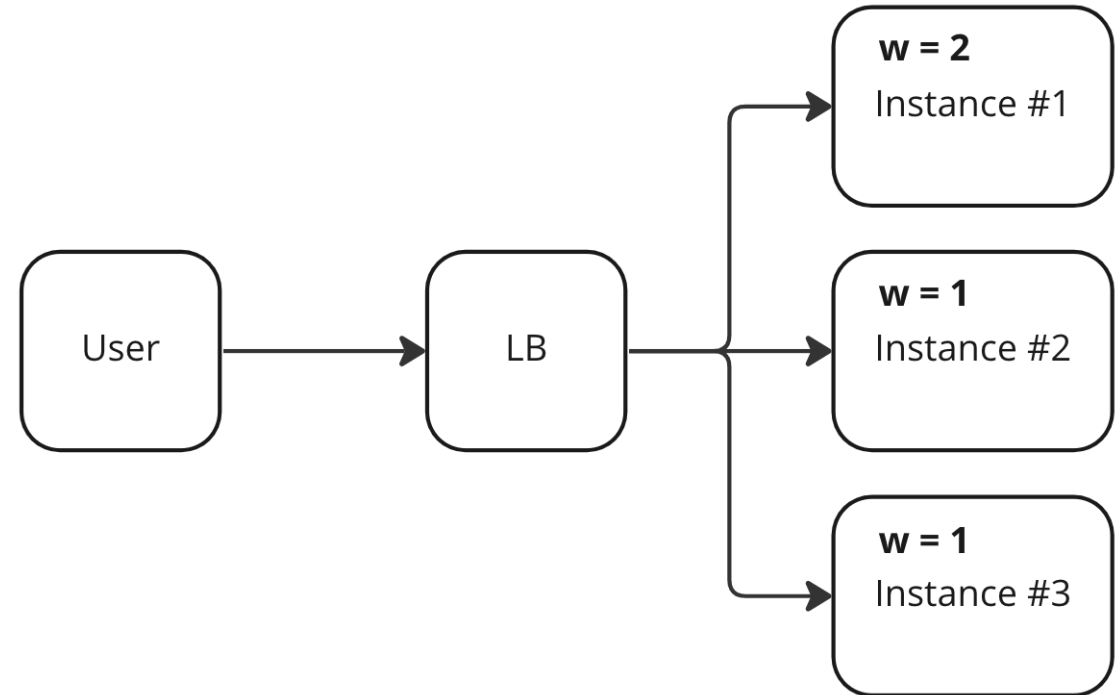
Round Robin

- 1: User -> Instance #1
- 2: User -> Instance #2
- 3: User -> Instance #3
- 4: User -> Instance #1
- 5: User -> Instance #3
- 6: User -> Instance #2



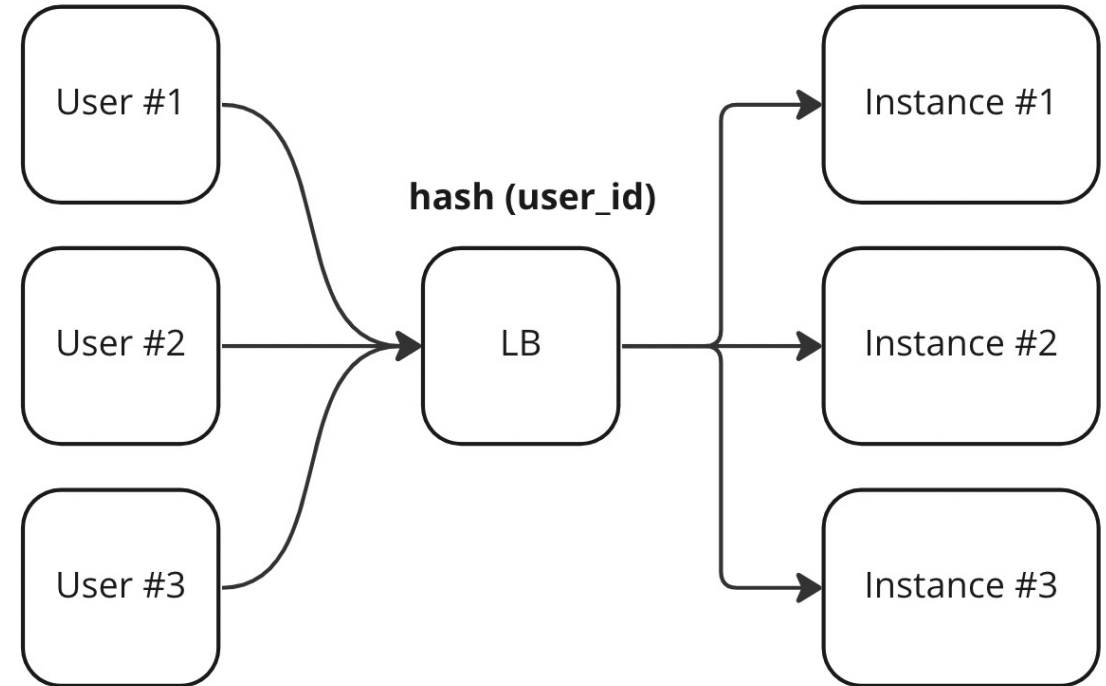
Weighted RR

- 1: User -> Instance #1
- 2: User -> Instance #1
- 3: User -> Instance #2
- 4: User -> Instance #3
- 5: User -> Instance #1
- 6: User -> Instance #3

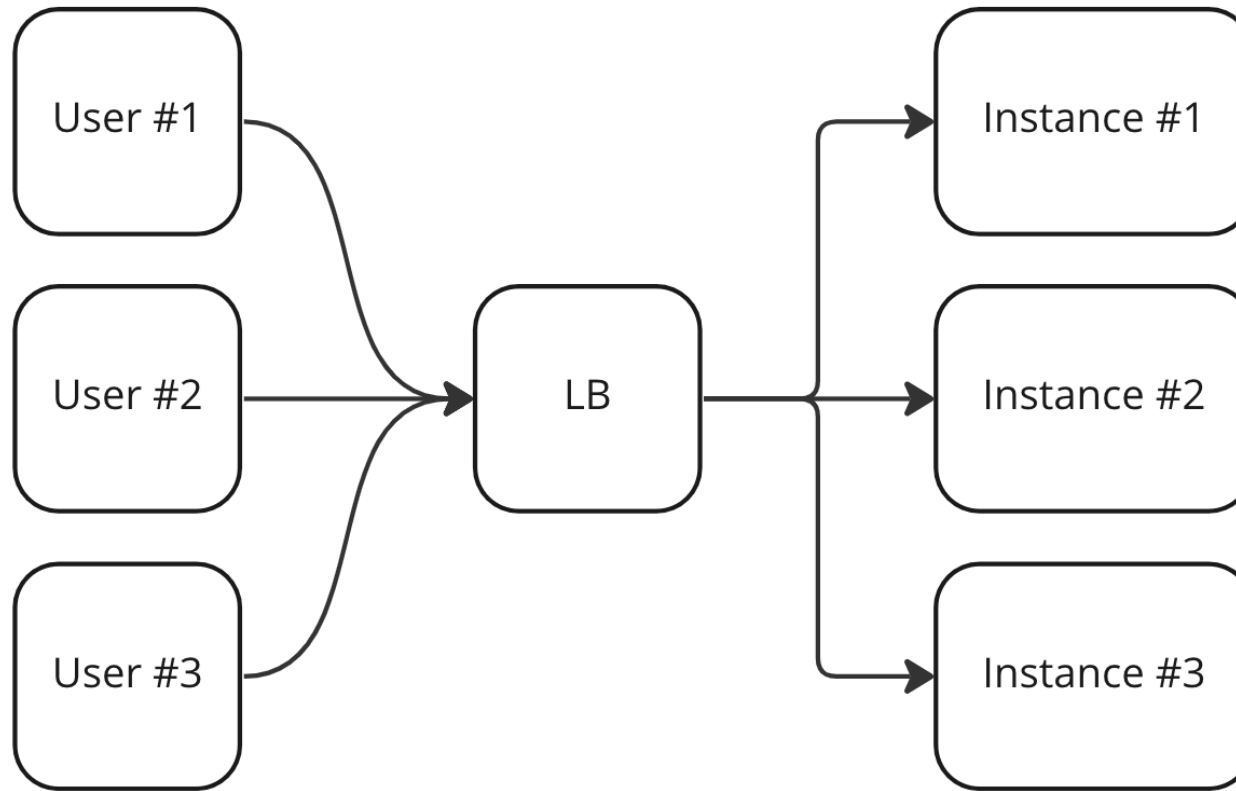


Sticky Sessions

- 1: User #1 -> Instance #2
- 2: User #2 -> Instance #1
- 3: User #3 -> Instance #3
- 4: User #2 -> Instance #1
- 5: User #3 -> Instance #3
- 6: User #1 -> Instance #2



Least Connections / RT / Bandwidth



Power of two choices

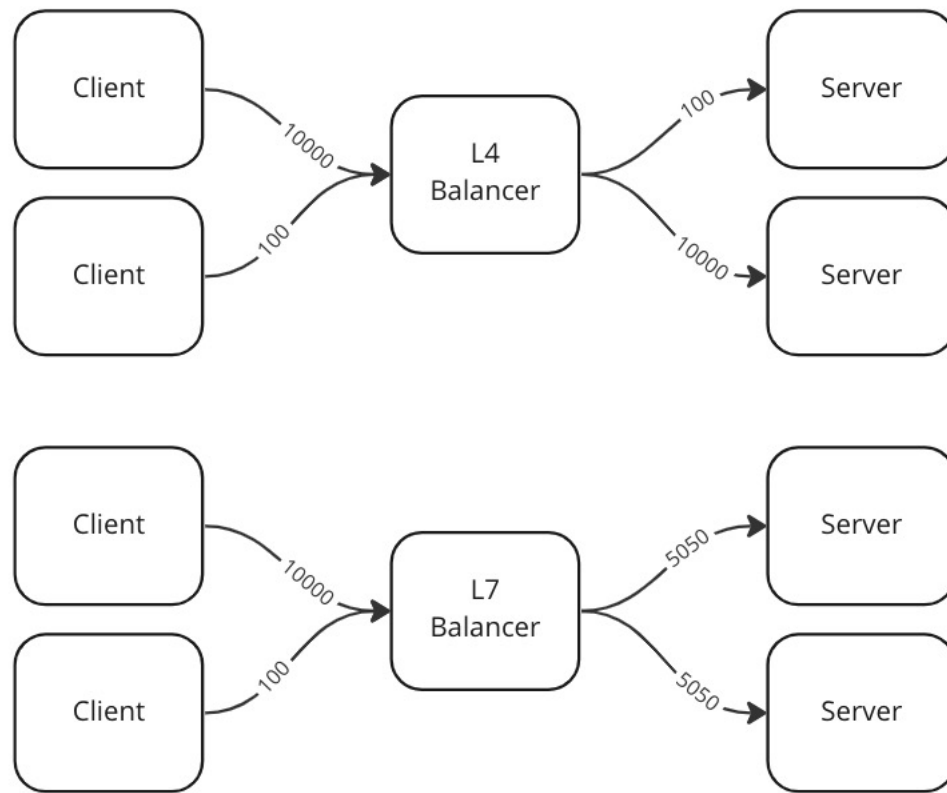
Случайным образом выбираем два бэкенда из списка – из этих двух выбираем лучший по целевой метрике (least connections / sessions / ...)



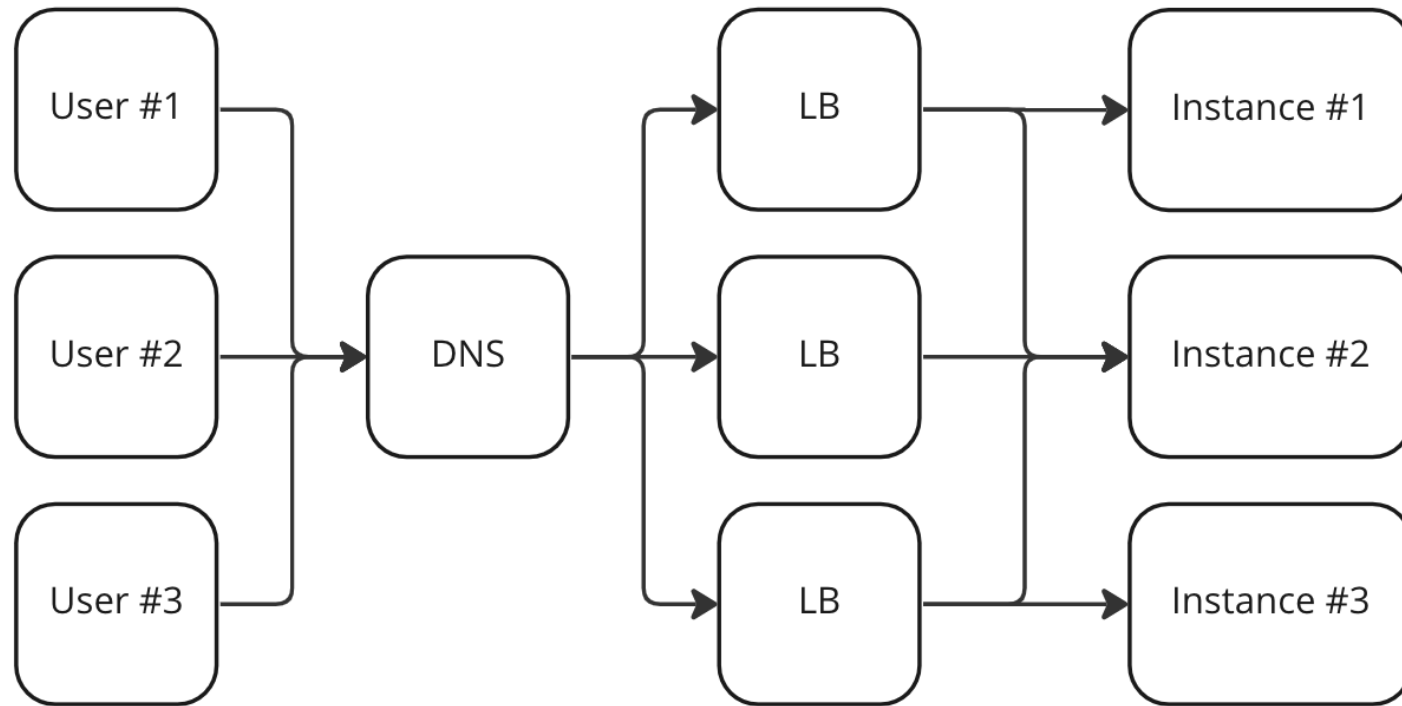
Nginx

```
1 upstream backend {  
2     server backend1.example.com:8080 weight=5;  
3     server backend2.example.com:8080;  
4 }  
5  
6 server {  
7     location / {  
8         proxy_pass http://backend;  
9     }  
10 }
```

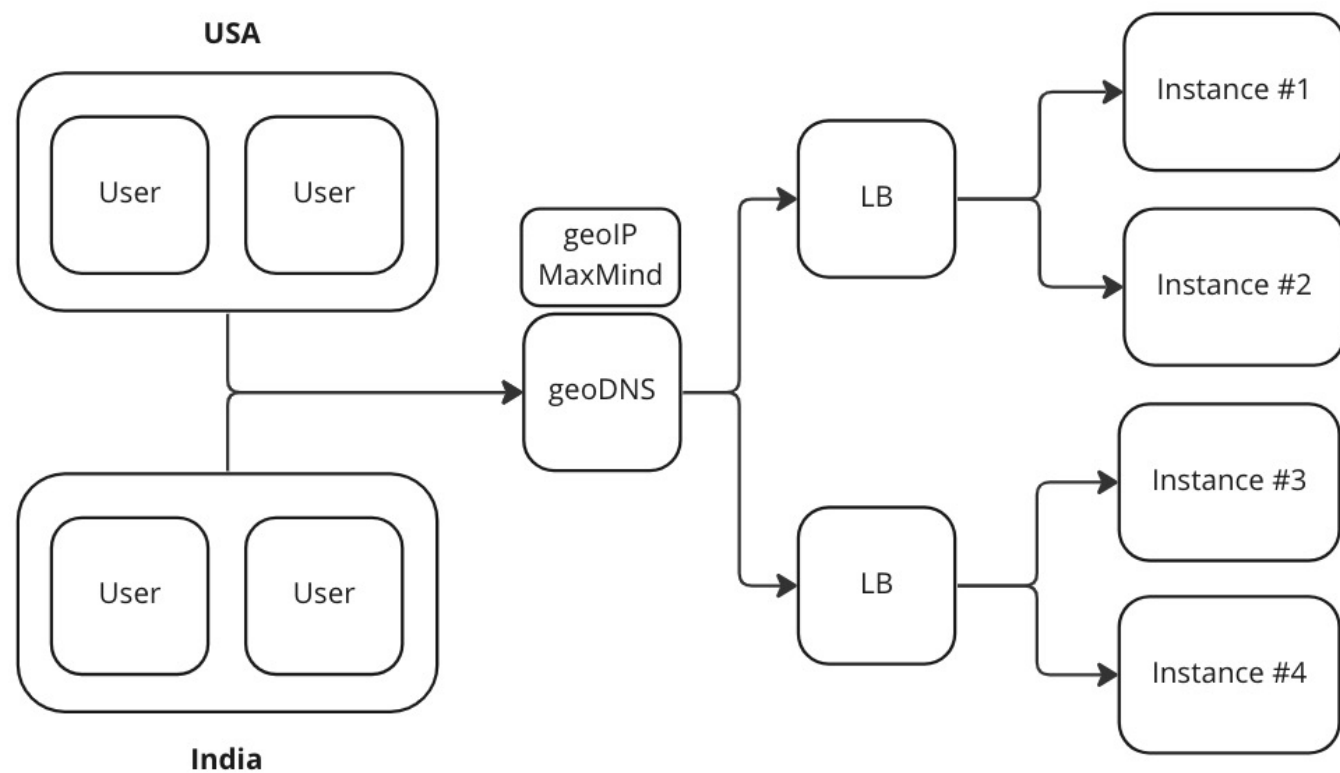
L4 / L7 Балансировка



DNS балансировка



geoDNS балансировка



FAQ:

Балансировка нагрузки

- Round Robin
- Weighted RR
- Sticky Sessions
- Least Connections
- L4 / L7 балансировка



Проксирование

Проксирование

Взлом / защита

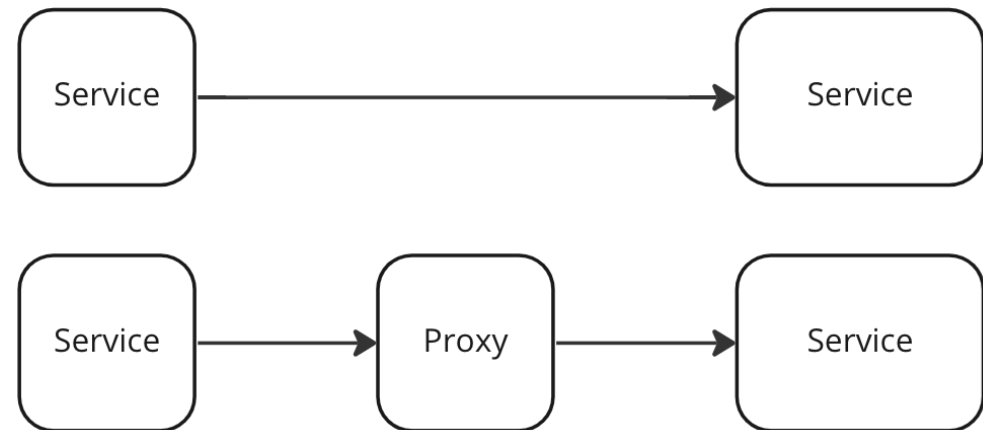
Кэширование данных

Ограничения трафика

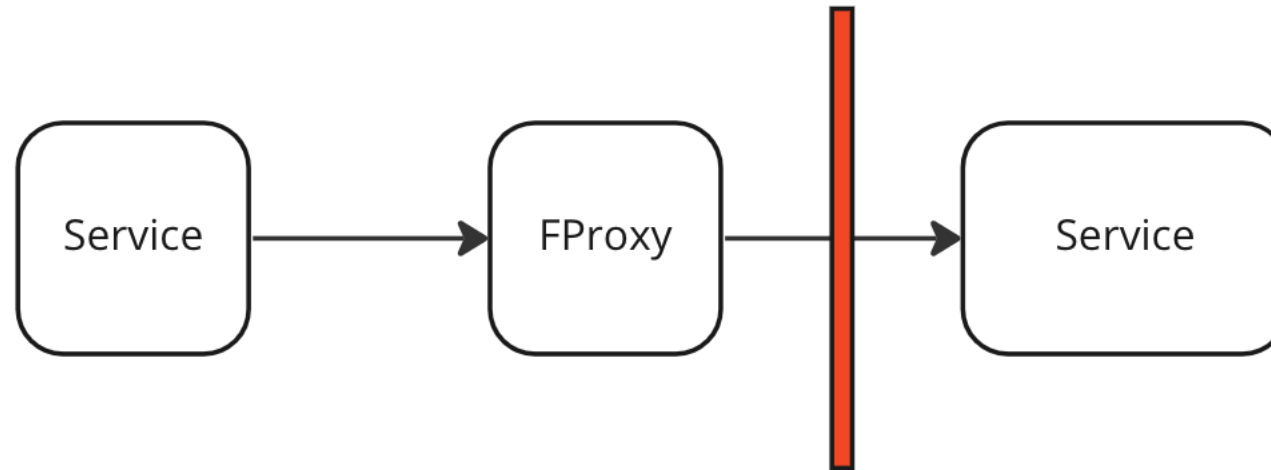
Обход ограничений доступа

Анонимность пользователей

Сжатие и модификация данных



Forward Proxy



14 марта 2022, 09:56 / Медиа

Роскомнадзор заблокировал Instagram в России

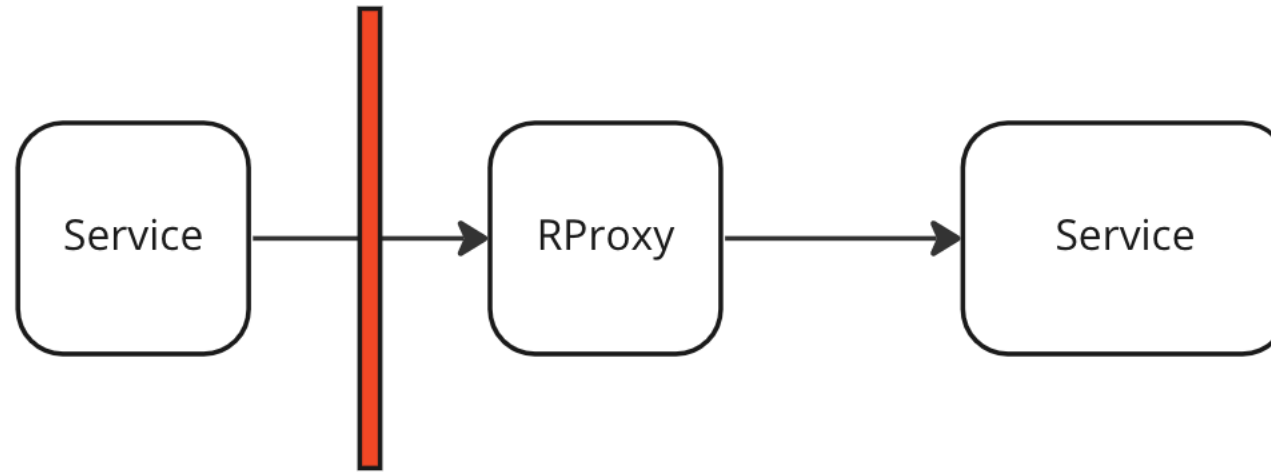
Ведомости



Прочту позже

Роскомнадзор внес социальную сеть Instagram в реестр запрещенных сайтов. Согласно [сервису](#) ведомства по проверке блокировок страниц и сайтов, доступ к Instagram.com ограничен по требованию Генпрокуратуры от 11 марта. По [данным](#) сервиса по мониторингу блокировок Globalcheck, доступность соцсети в России составляет 0%.

Reverse Proxy





Поиск в Google

Мне повезёт!

```
1 vladimirbalun@Vladimirs-MacBook-Pro ~ % nslookup google.com
2 Server:      192.168.1.254
3 Address:     192.168.1.254#53
4
5 Non-authoritative answer:
6 Name:   google.com
7 Address: 173.194.222.100
8 Name:   google.com
9 Address: 173.194.222.101
10 Name:  google.com
11 Address: 173.194.222.113
12 Name:  google.com
13 Address: 173.194.222.139
14 Name:  google.com
15 Address: 173.194.222.138
16 Name:  google.com
17 Address: 173.194.222.102
```

FAQ:

Проксирование

- Reverse
- Forward



Кэширование

Кэширование

- Сокращение response time сервисов
- Снижение лишней нагрузки на сторонние сервисы
- Переиспользование ранее полученных или вычисленных данных
- Стабилизация работы при кратковременных отказах систем

Основные термины

Cache miss - промах кэша, запрошенный ключ не был найден в кэше

Cache Hit - попадание в кэш, запрошенный ключ найден в кэше

Hit ratio - процент попаданий запросов в кэш, характеризует
эффективность кэширования

Горячий ключ - ключ, на который приходится большая часть запросов

Прогрев кэша - процесс наполнения кэша данными

Инвалидация - удаление кэшированных данных

Какие данные кэшировать

Меняются часто (секунды) – кэшировать чаще всего бессмысленно, но иногда может пригодиться

Меняются нечасто (минуты и часы) – чаще всего здесь задаемся вопросом – «стоит ли мне кэшировать»

Меняются редко (дни, недели, месяцы) – в данном случае можно спокойно кэшировать эти данные

Кэширование ошибок

Кэшируем ошибки и тогда последующие запросы не будут обращаться к источнику информации, а также не будет **cache miss attack**

По хорошему, нужно уметь держать нагрузку без кэша.

Задача кэша – ускорить ответ, а не держать нагрузку.

Всегда ли кэширование полезно?

ЭФФЕКТИВНОСТЬ

$AverageTime = DBAccessTime * CacheMissRate + CacheAccessTime$

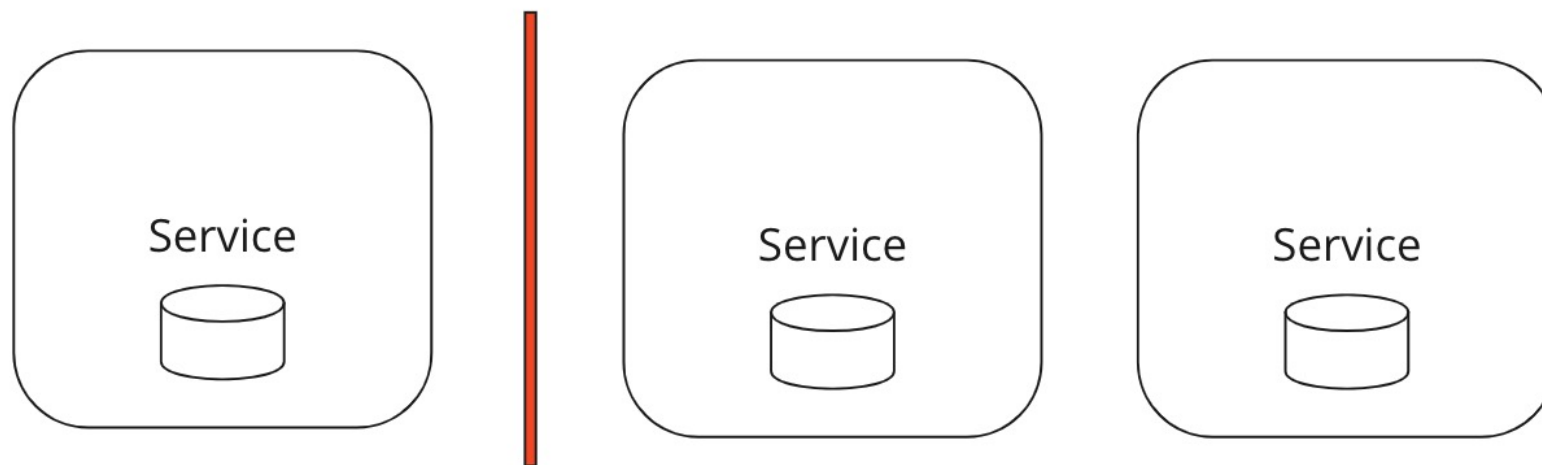
Пусть:

- $DBAccessTime = 100ms$
- $CacheAccessTime = 20ms$

Тогда при $CacheMissRate > 0.8$ – кэш вреден!

Виды кэширования

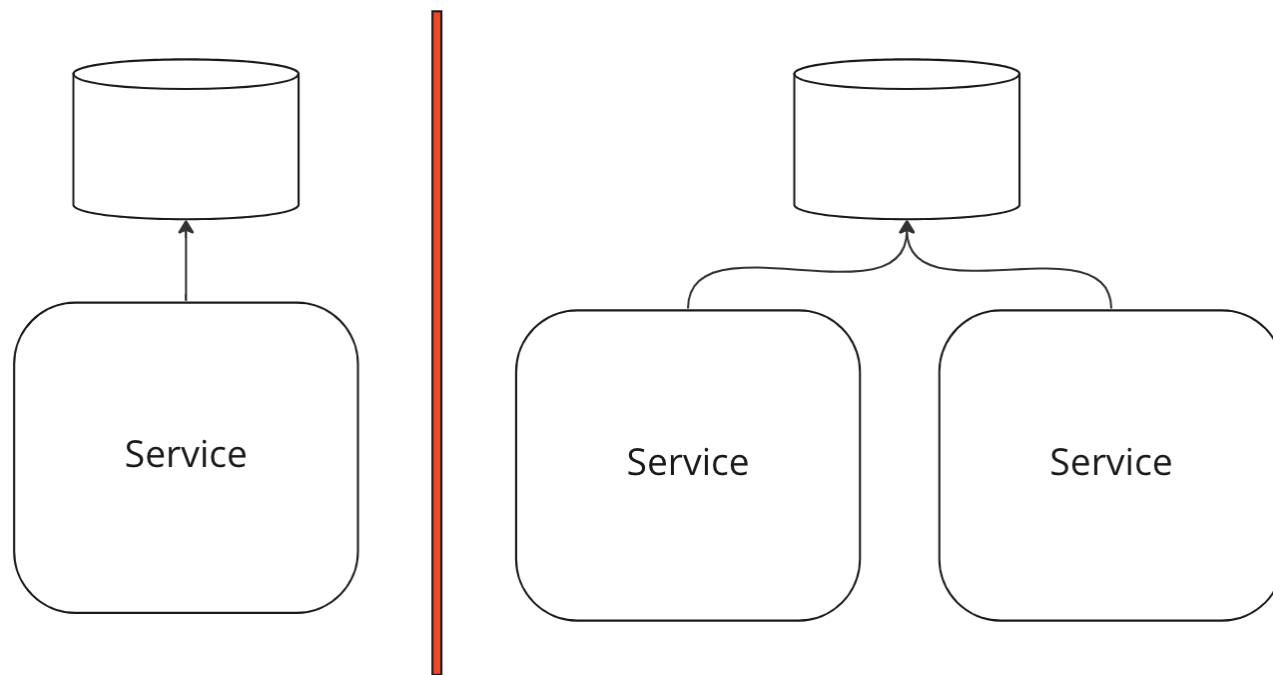
Внутреннее кэширование



Внутреннее кэширование

- + Высокая скорость
- + Отсутствие сетевых запросов
- + Нет расходов на Marshaling/Unmarshalling данных
- Горизонтальное масштабирование
- Прогрев кэша после падения сервиса

Внешнее кэширование



Внешнее кэширование

- + Хранение большого объема данных
- + Простое горизонтальное масштабирование
- + После падения сервиса данные кэша не теряются
- + Простой прогрев кэша и простая логика инвалидации
- Скорость работы

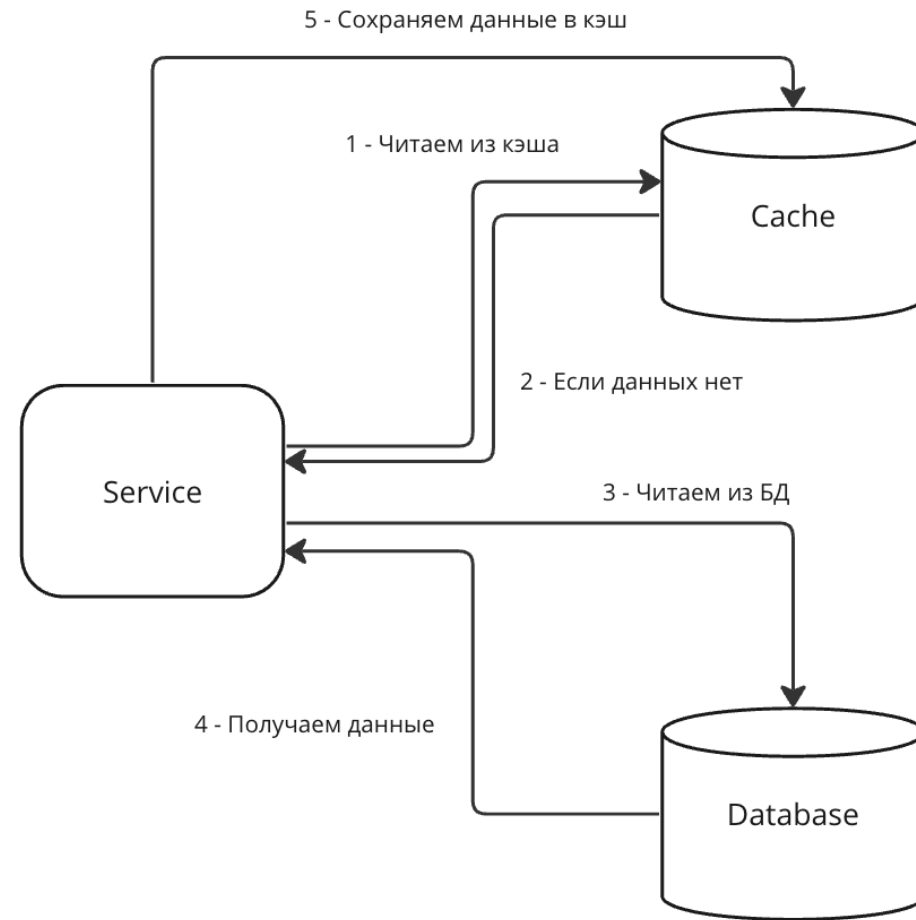
Способы взаимодействия с кэшем

Cache Aside

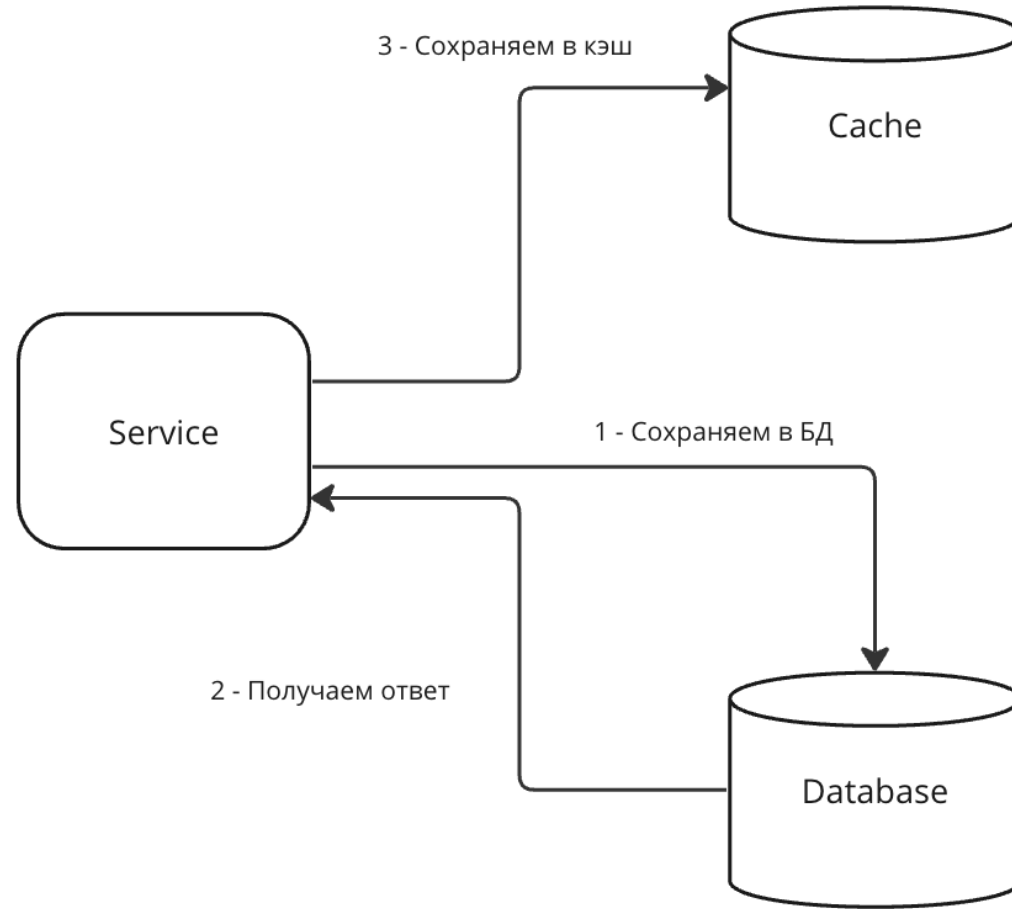
(Кэширование на стороне)

В этой стратегии, приложение координирует запросы в кэш и БД и само решает, куда и в какой момент нужно обращаться

Read Aside



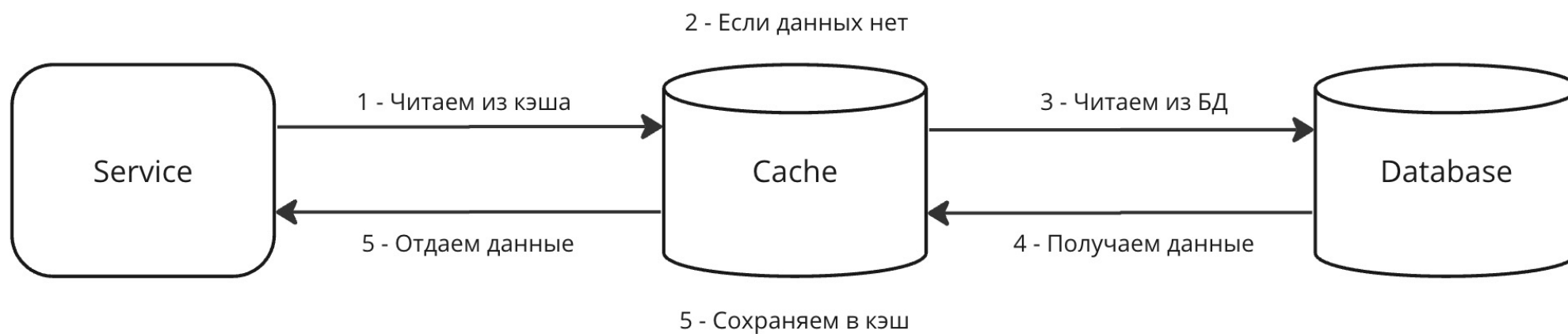
Write Aside



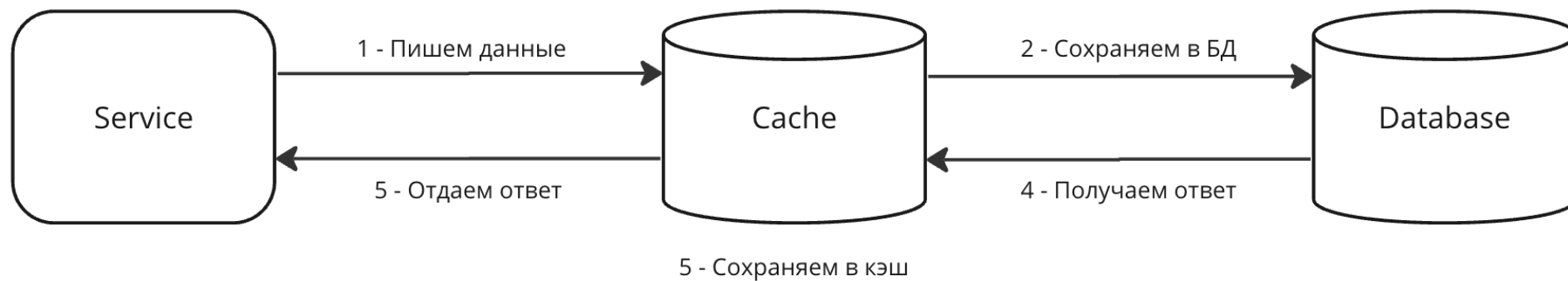
Cache Through **(Сквозное кэширование)**

В рамках этой этой стратегии все запросы от приложения
проходят через кэш

Read Through



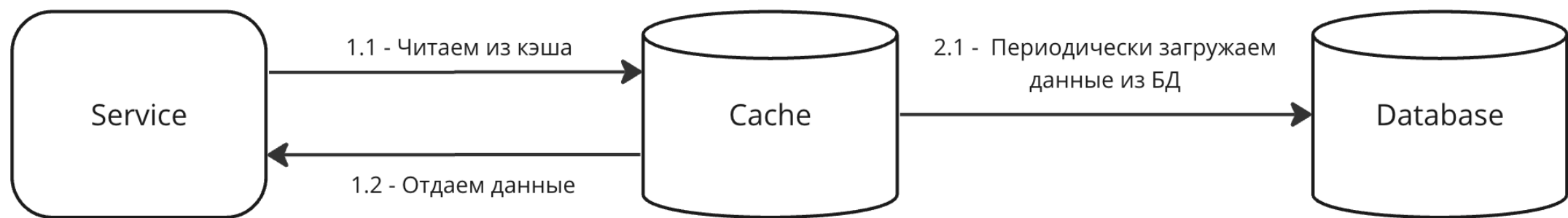
Write Through



Cache Ahead

(Опережающее кэширование)

Запросы на чтение всегда идут только в кэш, никогда
не попадая в БД напрямую

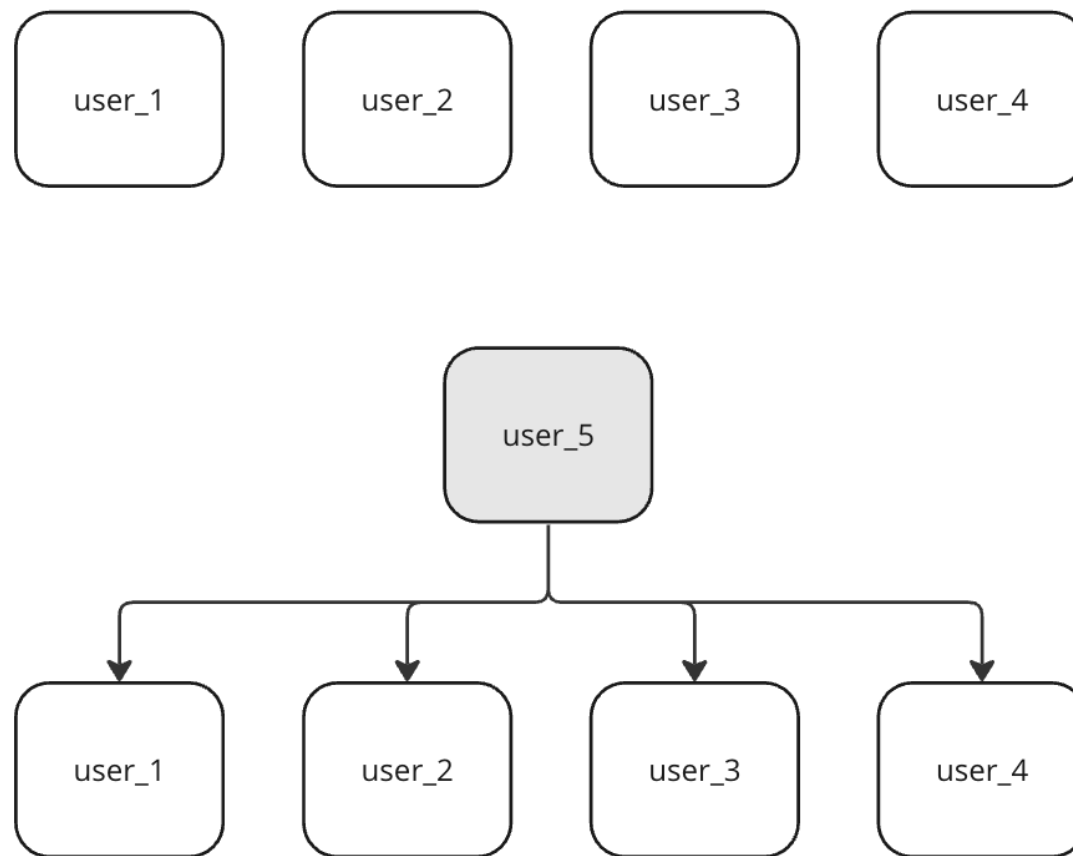


Резюме

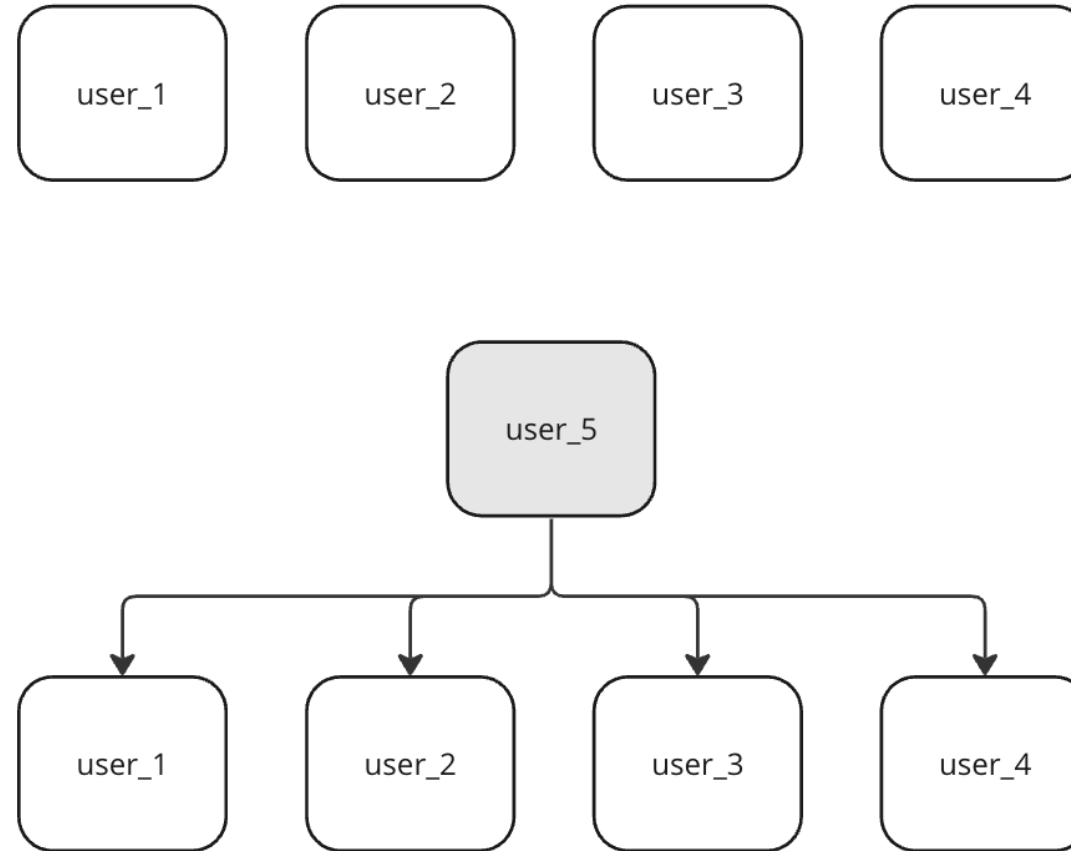
- Важно учитывать актуализацию данных
- Стоит балансировать между характером нагрузки (какая нагрузка преобладает чтение или запись)
- Также не забыть про отказоустойчивость

Алгоритмы вытеснения данных

Кого вытеснять?

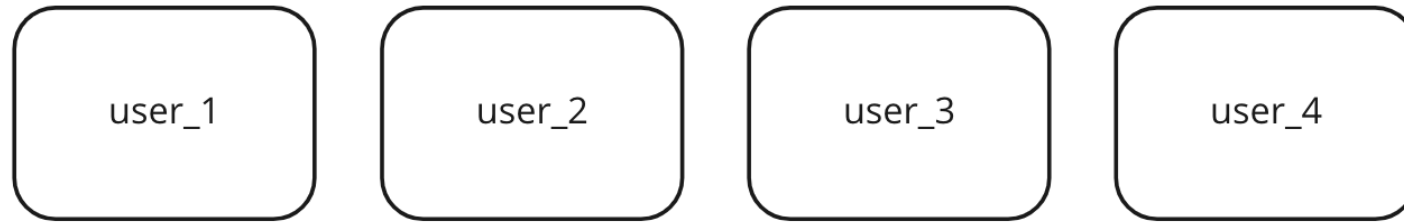


Random

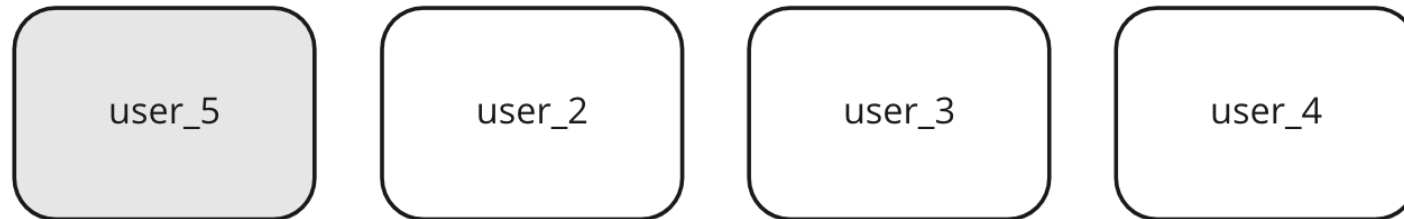


FIFO

add - user_1
add - user_2
add - user_3
add - user_4



add - user_5



LIFO

add - user_1
add - user_2
add - user_3
add - user_4



add - user_5



LRU

add - user_1
add - user_2
add - user_3
add - user_4

user_1

user_2

user_3

user_4

get - user_1
add - user_5

user_1

user_5

user_3

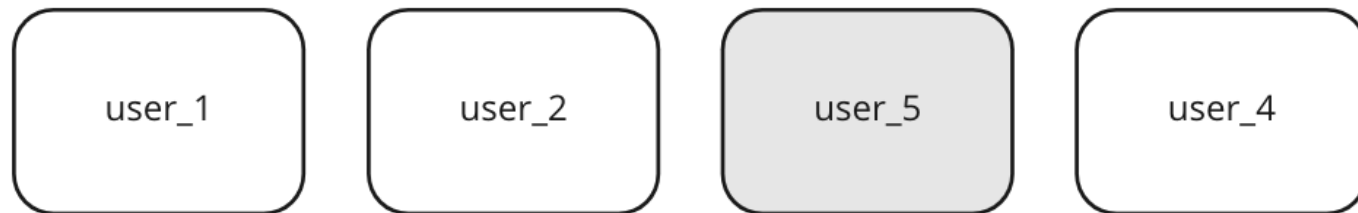
user_4

MRU

add - user_1
add - user_2
add - user_3
add - user_4



get - user_3
add - user_5



LFU

add - user_1
add - user_2
add - user_3
add - user_4

user_1

user_2

user_3

user_4

get - user_3
get - user_1
get - user_4
add - user_5

user_1

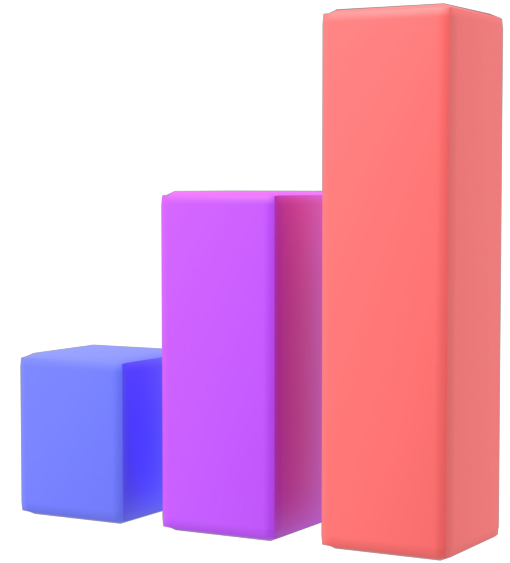
user_5

user_3

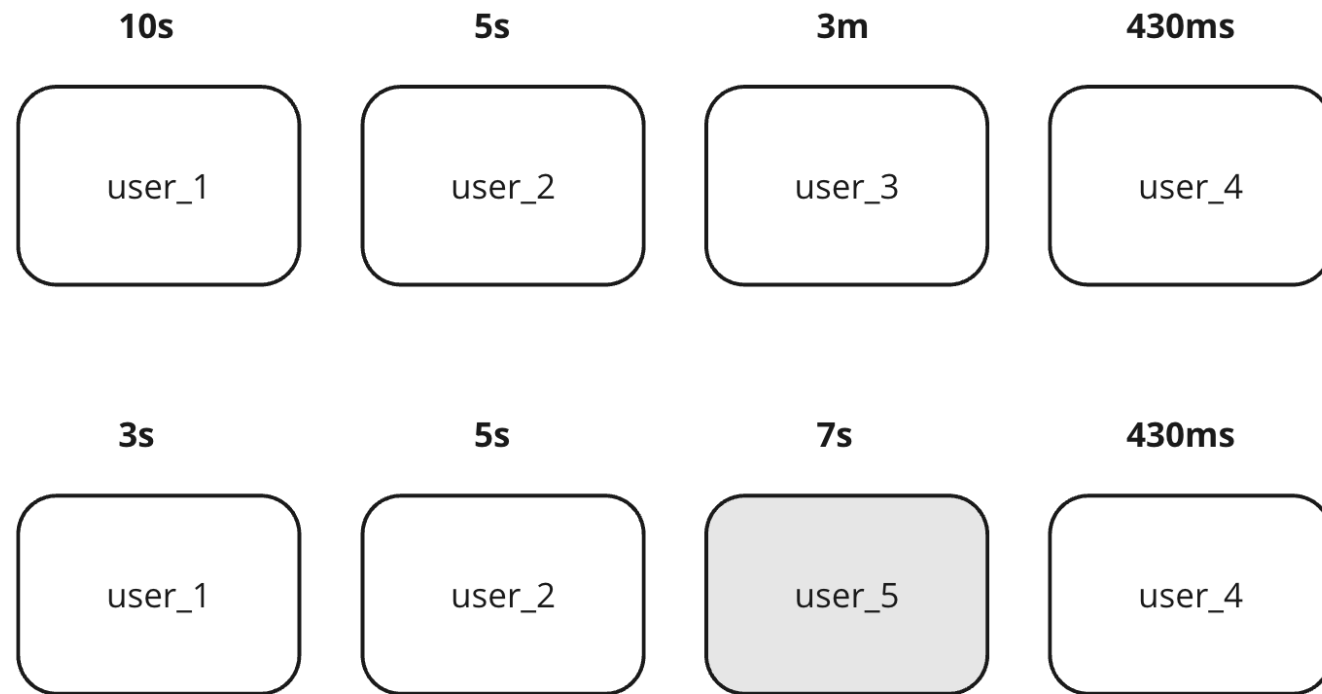
user_4

Резюме

- **LRU** — не использованный дольше всех вылетает из кеша
- **MRU** — последний использованный вылетает из кеша (специфичный кейс, бережем старье)
- **LFU** — реже всего использованный вылетает из кеша

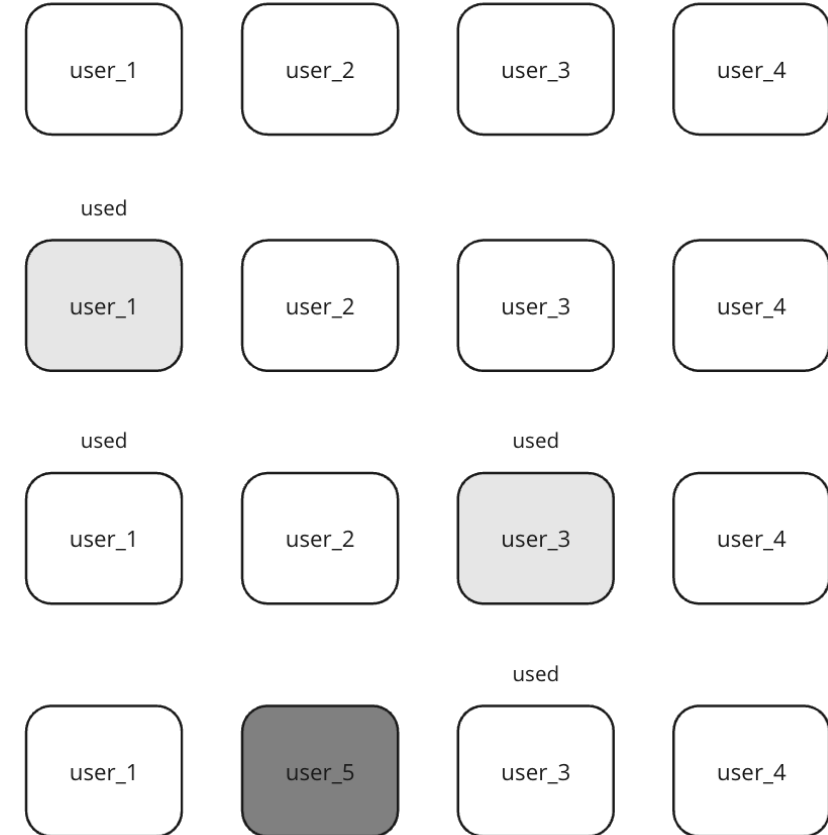


Алгоритм Беладди (ОРТ)



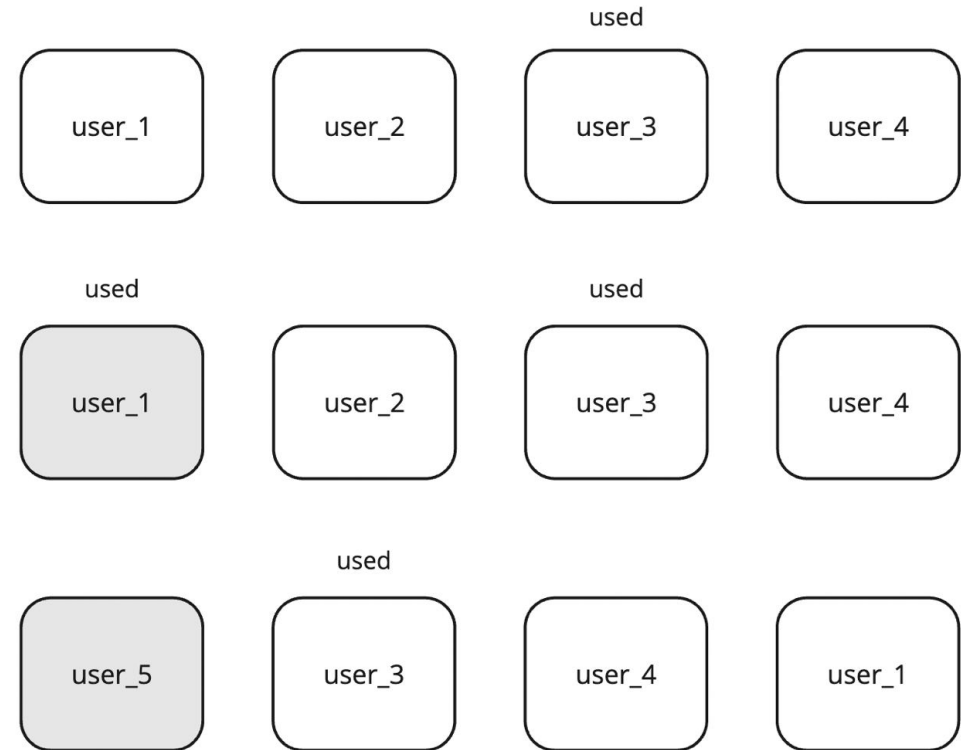
Second Chance

Очередь FIFO, только с битом использования. Для вытеснения делаем проход по очереди, если бит использования равен 0, то вытесняем, если бит равен 1, то устанавливаем в 0 и идем дальше, если не нашли ни одного бита равным 0 - делаем, делаем проход заново



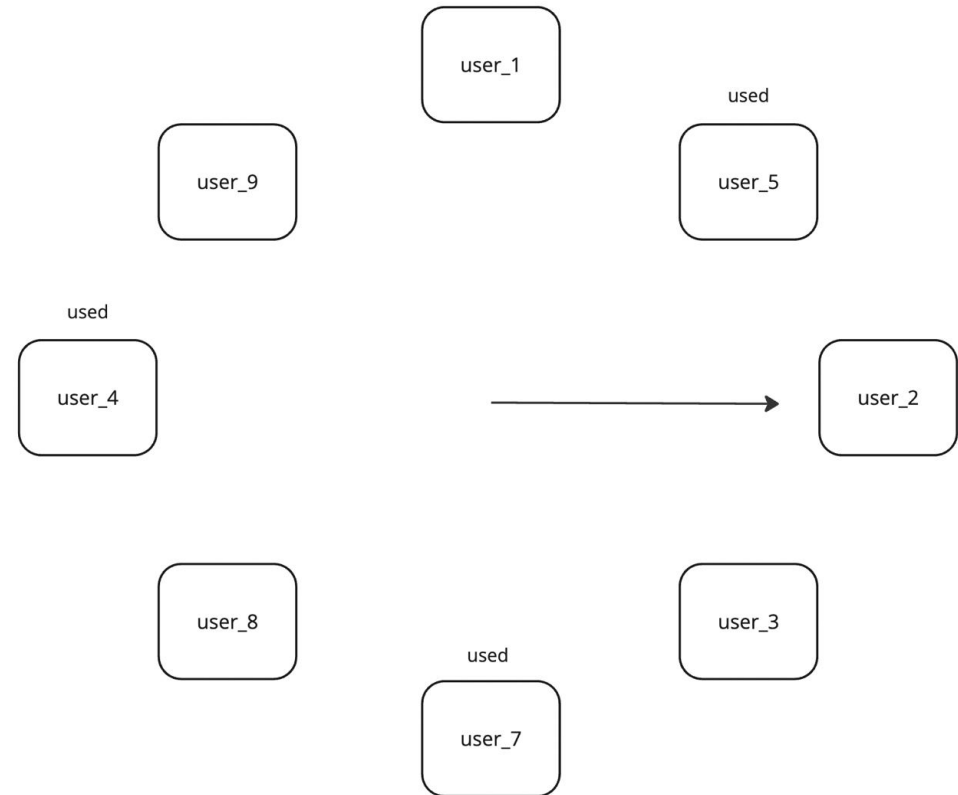
Second Chance

При обращении выставляем бит присутствия, а при вытеснении удаляем из начала очереди, если бит равен нулю, а если равен единице, то меняем бит на ноль, затем переносим элемент в конец очереди и идем к следующему



Clock

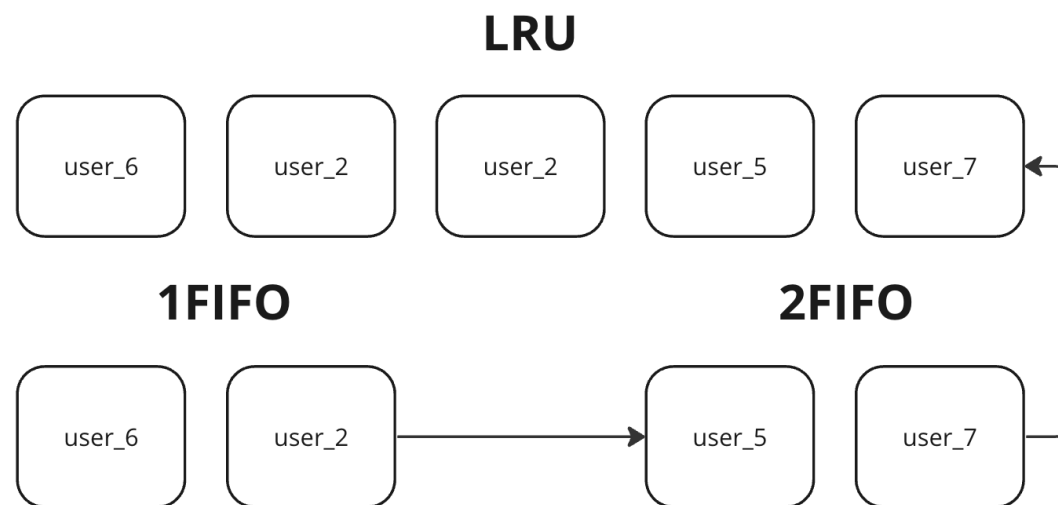
Тот же Second Chance, только не нужно двигать элемент из начала в конец очереди, если бит равен единице



2Q

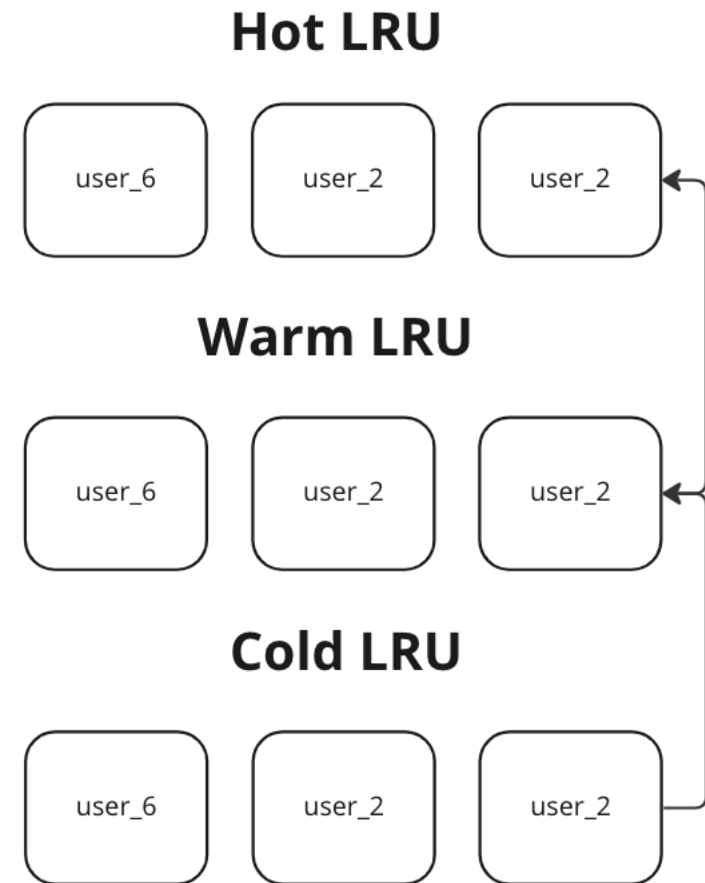
Элементы запрошенные из 1FIFO
никуда не двигаются. Вытесненные
из 1FIFO — перемещаются в 2FIFO

Элементы запрошенные из 2FIFO —
уходят в LRU. Вытесненные из 2LIFO
удаляются



SLRU

Сперва кладем в первую коробочку, при повторном запросе перекладываем во вторую, при еще одном повторном из второй — в третью



TLRU (Time aware LRU)

Абсолютно точной такой же алгоритм, только к данным добавляется их время жизни в кэше (TTL), по которому они автоматически удаляются из кэша

LRU-k

Удаляет страницу, K-й последний доступ к которой находится дальше всего в прошлом. Например, LRU-1 — это просто LRU, тогда как LRU-2 удаляет страницы в соответствии со временем их предпоследнего доступа

FAQ: Кэширование

- Внутреннее и внешнее
- Способы взаимодействия
- Алгоритмы вытеснения



API

CRUD

Акроним, обозначающий четыре базовые функции, используемые при работе с данными: создание (C), чтение (R), модификация (U), удаление (D)



REST

Глагол – метод запроса (GET, POST, DELETE, PUT)

Существительное – URI запроса

Дополнительная информация – хедеры запроса

Содержимое – тело запроса и ответа в Json

Результат – код ответа

```
1 Request:
2 GET ${address}/employee/14
3
4 Response:
5 200 - Ok
6 {
7   "name": "Petr",
8   "surname": "Ivanov",
9   "city": "Moscow"
10 }
11
```

SOAP

```
1 <?xml version="1.0"?>
2 <soap:Envelope
3   xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"
4   soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">
5 <soap:Body>
6   <m:GetPrice xmlns:m="https://online-shop.ru/prices">
7     <m:Item>Dell Vostro 3515-5371</m:Item>
8   </m:GetPrice>
9 </soap:Body>
10 </soap:Envelope>
```

```
1 <?xml version="1.0"?>
2 <soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"
3   soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">
4 <soap:Body>
5   <m:GetPriceResponse xmlns:m="https://online-shop.ru/prices">
6     <m:Price>37299</m:Price>
7   </m:GetPriceResponse>
8 </soap:Body>
9 </soap:Envelope>
```

RPC

Класс технологий, позволяющих программам
вызывать функции или процедуры в другом
адресном пространстве

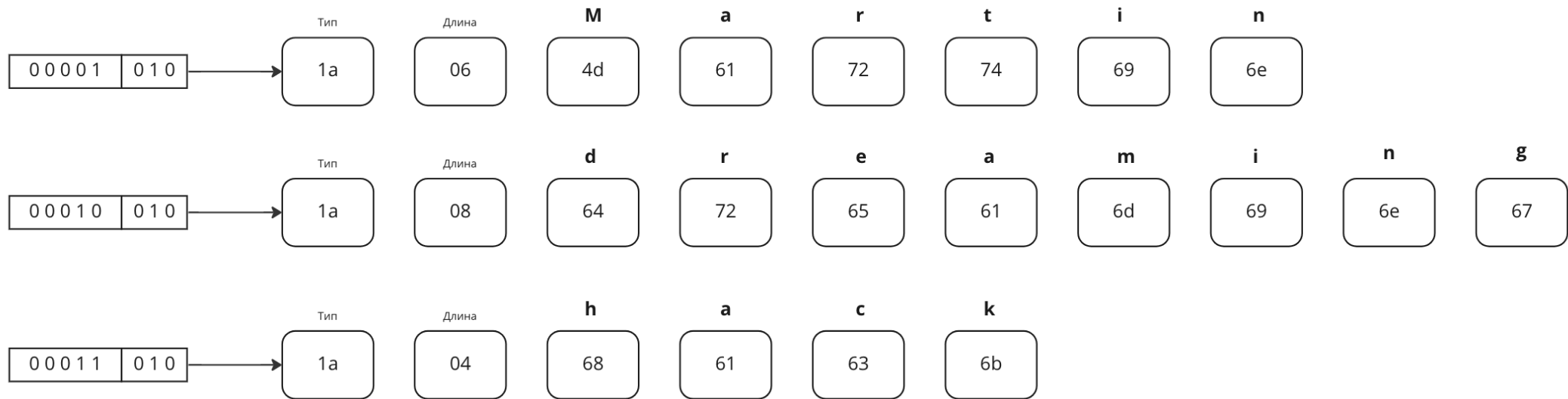


gRPC

```
1 message Person {
2   string name = 1;
3   int32 id = 2;
4   string email = 3;
5
6   enum PhoneType {
7     MOBILE = 0;
8     HOME = 1;
9     WORK = 2;
10  }
11
12  message PhoneNumber {
13    string number = 1;
14    PhoneType type = 2;
15  }
16
17  repeated PhoneNumber phones = 4;
18 }
19
20
21 service PersonsList {
22   rpc SavePerson(Person) returns (google.protobuf.Empty) {}
23 }
```

```
1 person := pb.Person{
2   Id: 1234,
3   Name: "John Doe",
4   Email: "jdoe@example.com",
5   Phones: []*pb.Person_PhoneNumber{
6     {Number: "555-4321", Type: pb.Person_HOME},
7   },
8 }
9
10 _, err := client.SavePerson(context.Background(), person)
11 if err != nil {
12   ...
13 }
```

Внутреннее устройство



Under / Over fetching

GraphQL

```
1 query {  
2   users {  
3     fname  
4     age  
5   }  
6 }
```

```
1 data : {  
2   users [  
3     {  
4       "fname": "Joe",  
5       "age": 23  
6     },  
7     {  
8       "fname": "Betty",  
9       "age": 29  
10    }  
11  ]  
12 }
```

GraphQL



```
1 mutation createUser{
2   addUser(fname: "Richie", age: 22) {
3     id
4   }
5 }
```



```
1 data : {
2   addUser : "a36e4h"
3 }
```

GraphQL



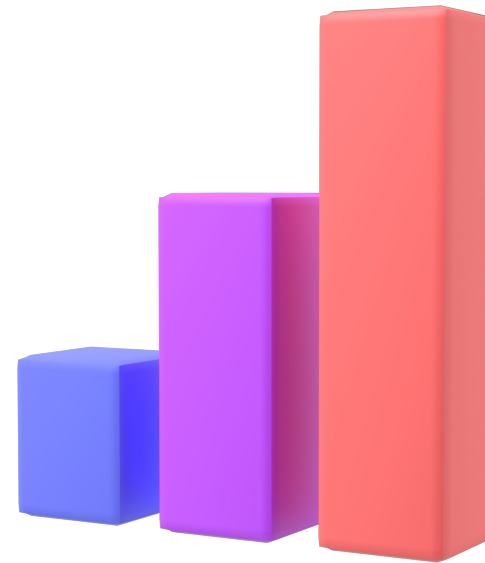
```
1 subscription listenLikes {  
2   listenLikes {  
3     fname  
4     likes  
5   }  
6 }
```



```
1 data: {  
2   listenLikes: {  
3     "fname": "Richie",  
4     "likes": 245  
5   }  
6 }
```

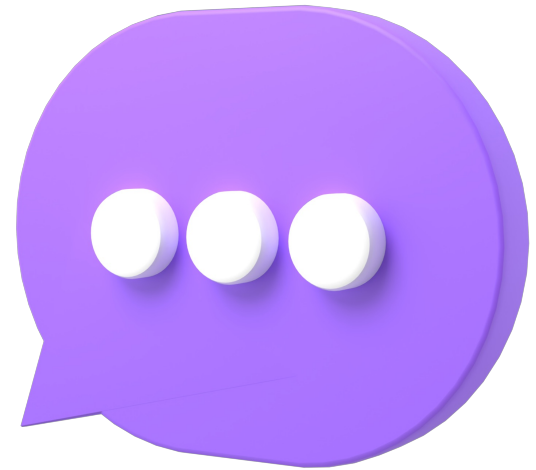
Резюме

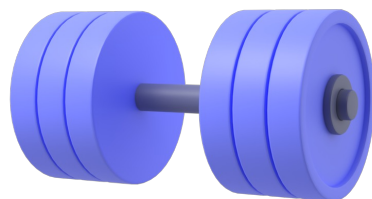
- SOAP уже умер
- REST для клиентского взаимодействия
- gRPC для внутреннего взаимодействия
- GraphQL для кастомизации запросов



FAQ: API

- REST
- SOAP
- gRPC
- GraphQL





Let's practise

Observability

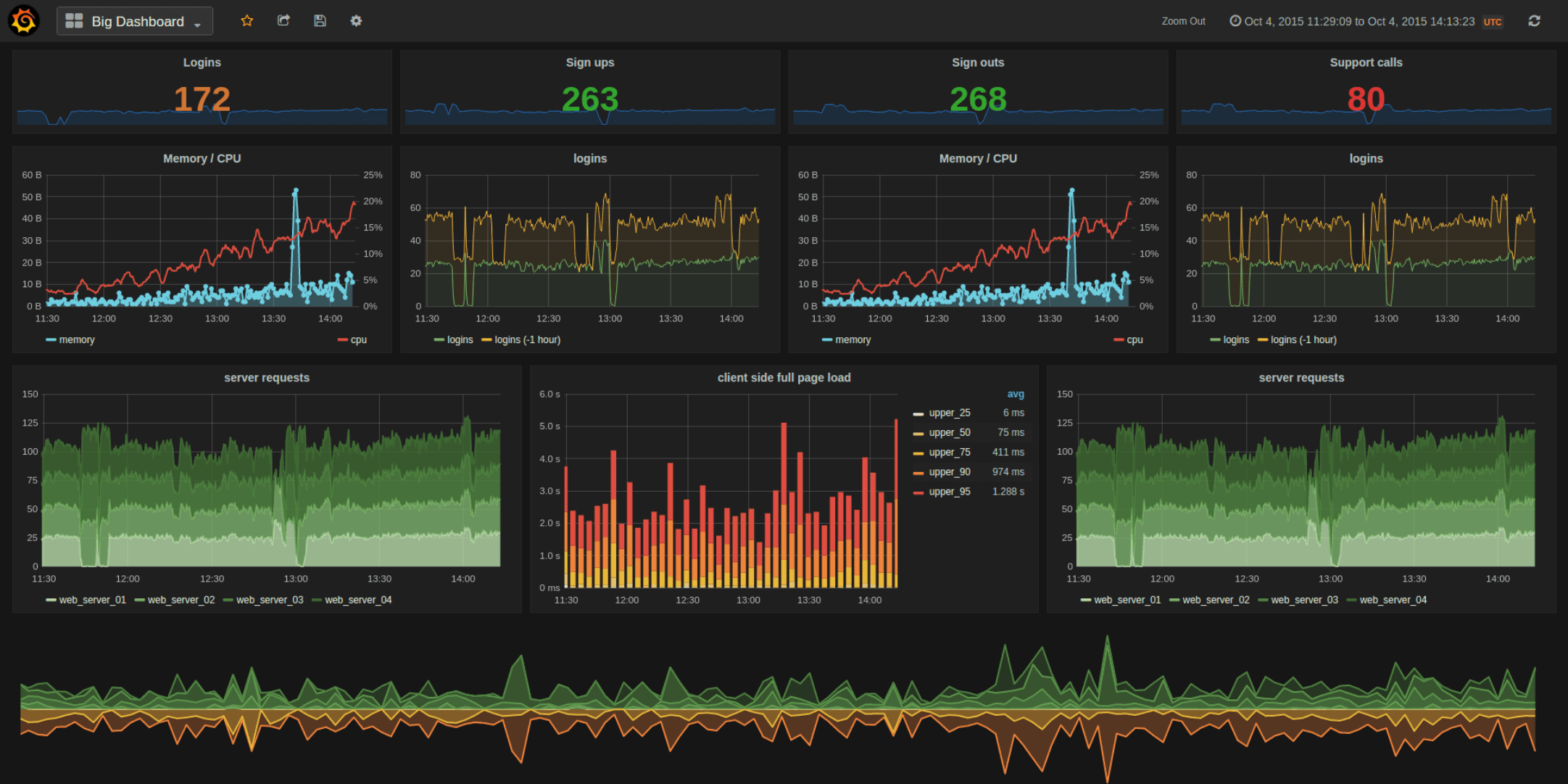
Мониторинг

Prometheus уже как стандарт...



```
1 func recordMetrics() {
2     go func() {
3         for {
4             opsProcessed.Inc()
5             time.Sleep(2 * time.Second)
6         }
7     }()
8 }
9
10 var (
11     opsProcessed = promauto.NewCounter(prometheus.CounterOpts{
12         Name: "myapp_processed_ops_total",
13         Help: "The total number of processed events",
14     })
15 )
16
17 func main() {
18     recordMetrics()
19
20     http.Handle("/metrics", promhttp.Handler())
21     http.ListenAndServe(":2112", nil)
22 }
```

```
1 # HELP myapp_processed_ops_total The total number of processed
  events
2 # TYPE myapp_processed_ops_total counter
3 myapp_processed_ops_total 5
```



Основные метрики

- RPS, TPS, QPS
- Response time
- Errors rate
- Traffic, CPU, RAM, HDD/SSD
- Uptime / Downtime
- Размеры очередей
- Количество процессов / потоков

Тре́йсинг

Jaeger или Zipkin по умолчанию

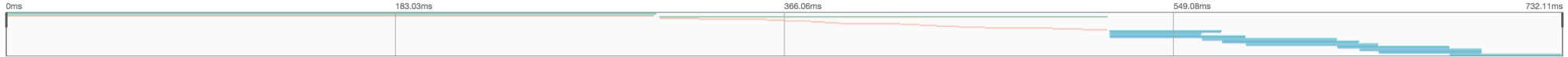


frontend: HTTP GET /dispatch

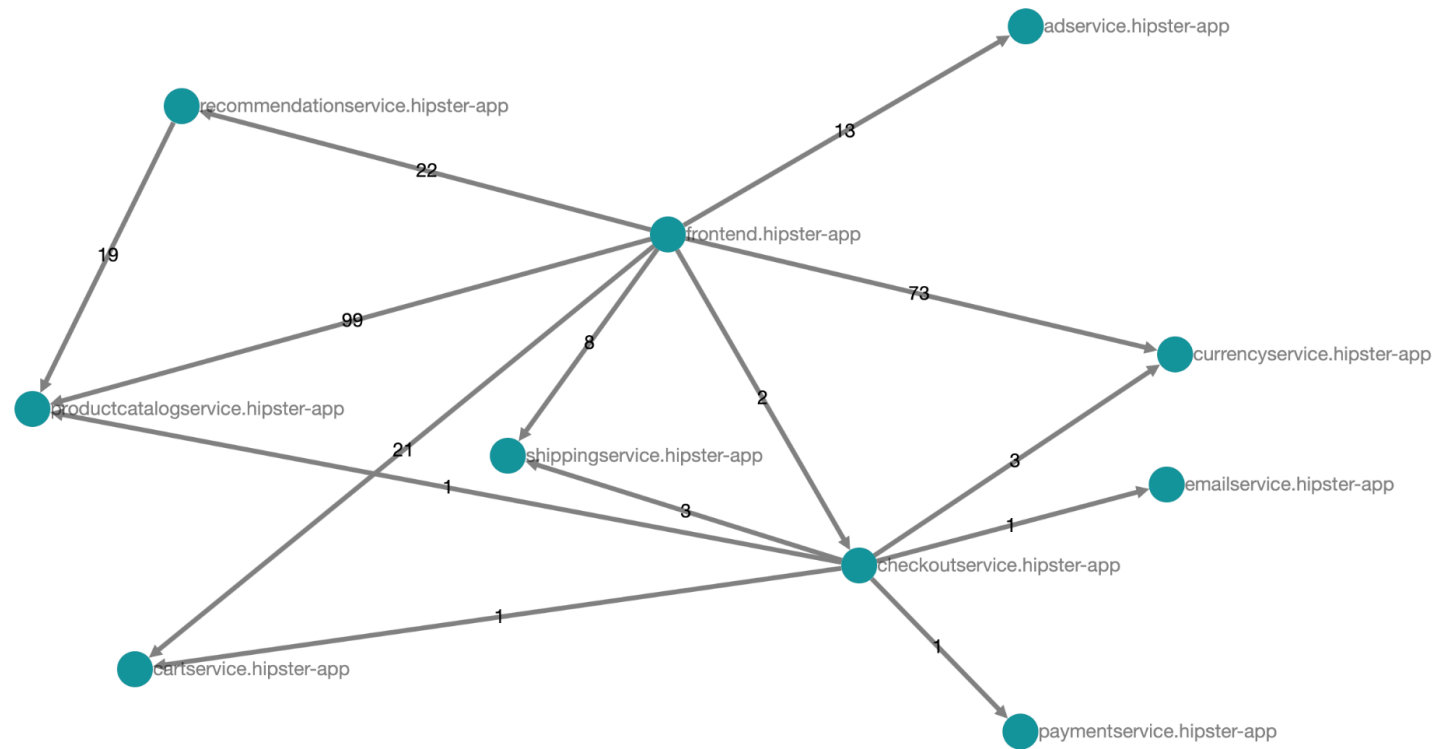
Search...

View Options

Trace Start: July 20, 2018 2:48 PM | Duration: 732.11ms | Services: 6 | Depth: 5 | Total Spans: 51



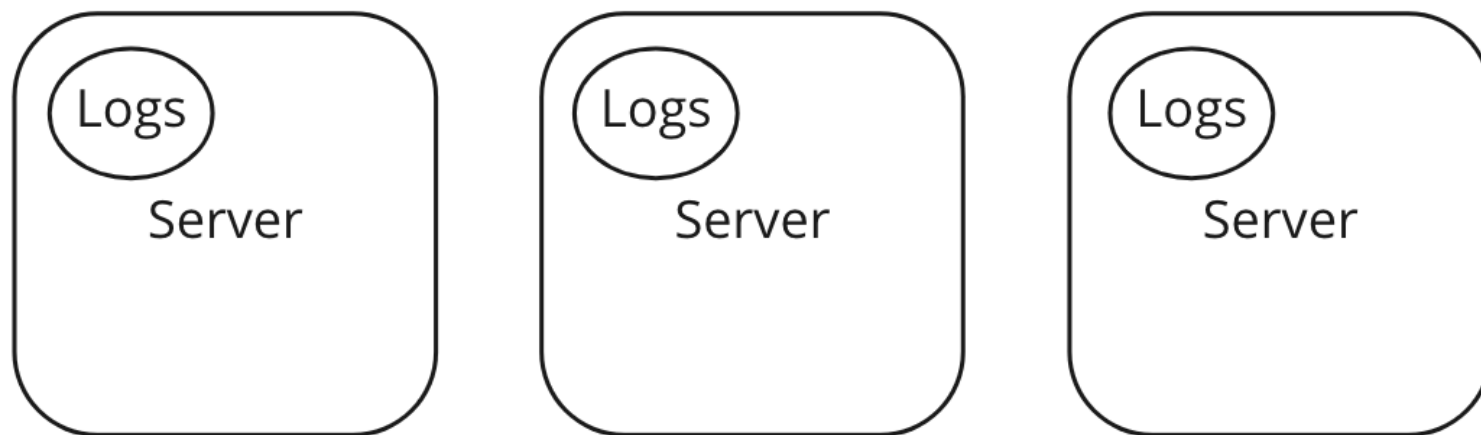
Service & Operation	0ms	183.03ms	366.06ms	549.08ms	732.11ms
▼ frontend HTTP GET /dispatch					
▼ frontend HTTP GET /customer					
▼ frontend HTTP GET					
▼ customer HTTP GET /customer					
mysql SQL SELECT					
▼ frontend Driver::findNearest					
▼ driver Driver::findNearest					
redis FindDriverIDs					
❗ redis GetDriver					
redis GetDriver					
redis GetDriver					
redis GetDriver					
redis GetDriver					
redis GetDriver					
❗ redis GetDriver					
redis GetDriver					
redis GetDriver					
redis GetDriver					
❗ redis GetDriver					
redis GetDriver					
redis GetDriver					
▼ frontend HTTP GET /route					
▼ frontend HTTP GET					
route HTTP GET /route					
▼ frontend HTTP GET /route					
▼ frontend HTTP GET					
route HTTP GET /route					
▼ frontend HTTP GET /route					
▼ frontend HTTP GET					
route HTTP GET /route					
▼ frontend HTTP GET /route					
▼ frontend HTTP GET					
route HTTP GET /route					



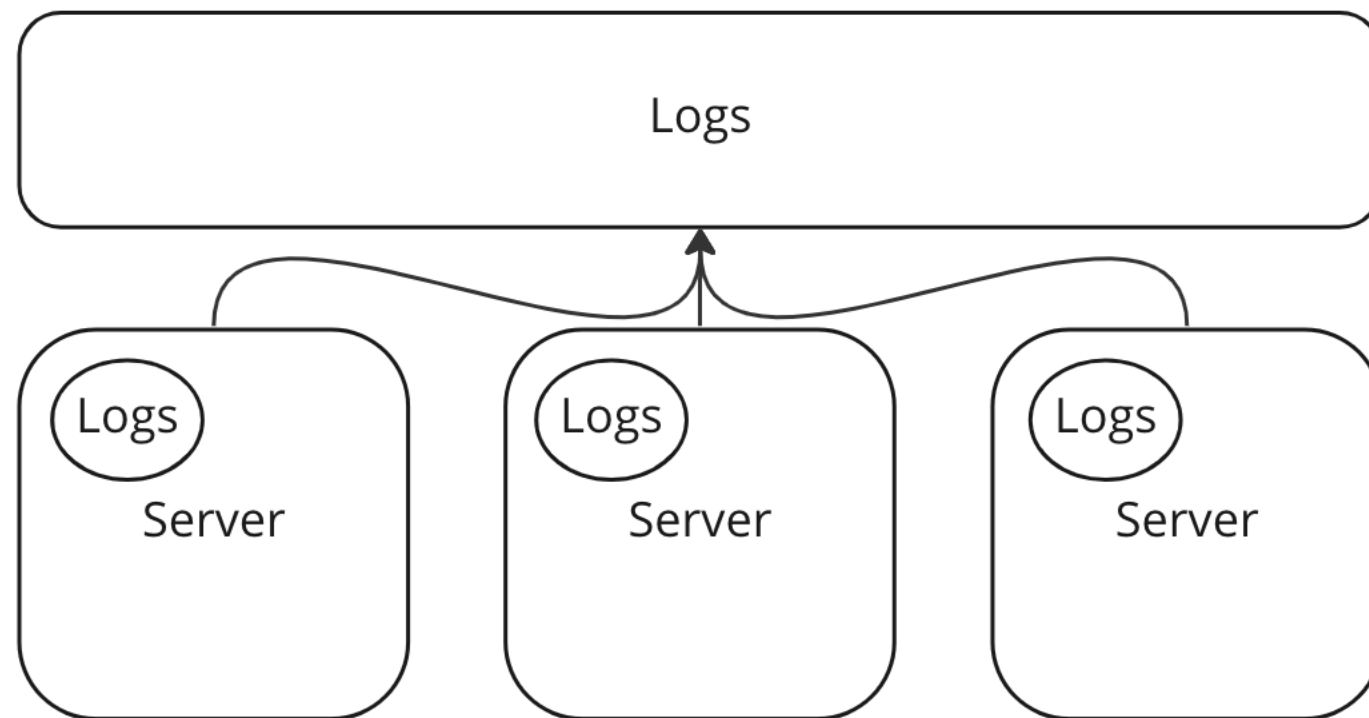


```
1 func isLoggedIn(ctx context.Context, r *http.Request) bool {  
2     span, ctx := opentracing.StartSpanFromContext(ctx, "isLoggedIn")  
3     defer span.Finish()  
4     ...  
5  
6  
7 func getCityByCountryName(ctx context.Context, countryName string) ([]City, error) {  
8     span, ctx := opentracing.StartSpanFromContext(ctx, "getCityByCountryName")  
9     defer span.Finish()  
10    ...
```

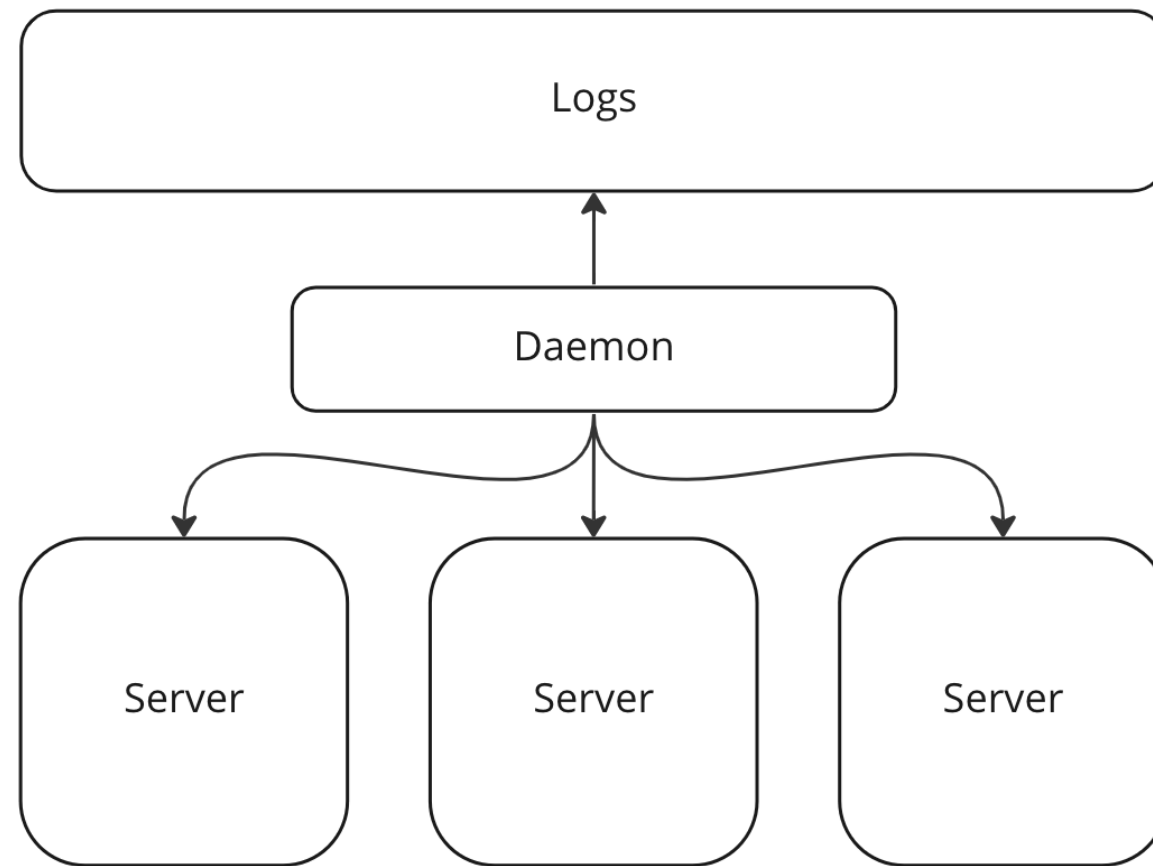
Логирование



Логирование



Логирование



Логирование



graylog

graylog

SearchStreamsAlertsDashboardsSourcesSystem1

In 0 / Out 0 msg/sHelpAdministrator

Search in the last 5 minutes

▶ Not updating

Saved searches

Q

gl2_source_input:58be0be8f0e1fc0e94f80450

Search result

Found 12,529 messages in 26 ms, searched in 1 index.
Results retrieved at 2017-04-11 00:00:31.

Add count to dashboard

Save search criteriaMore actions

FieldsDecorators

DefaultAllNoneFilter fields

▶ @timestamp

▶ @version

▶ category

▶ destIp

▶ destPort

▶ mac

▶ message

▶ NetworkSecurityGroup

▶ operationName

▶ protocol

List fields of current page or all fields.

Histogram

Add to dashboard

YearQuarterMonthWeekDayHourMinute

3K

2K

1K

23:55

23:56

23:57

23:58

23:59

Tue 11

Messages

Previous12345678910Next

Timestamp

source

2017-04-11 00:00:29.027

unknown

2017-04-11 00:00:29.027

unknown

2017-04-11 00:00:29.027

unknown

2017-04-11 00:00:29.027

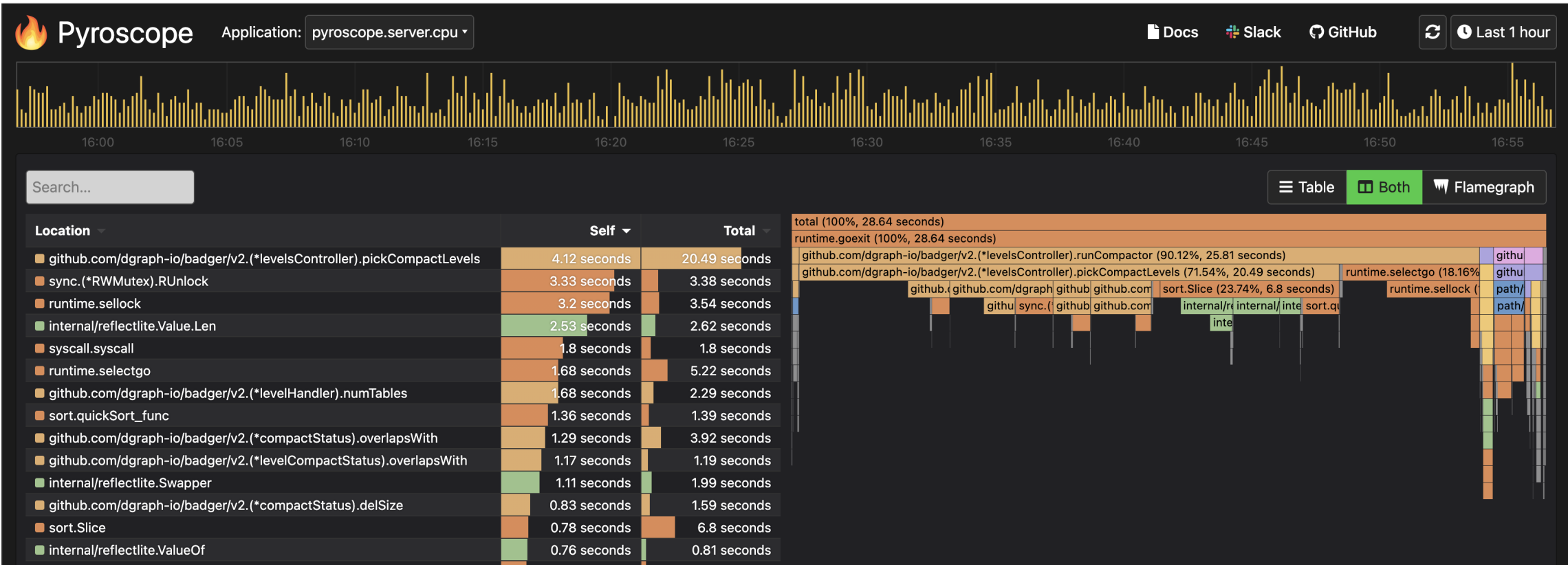
unknown

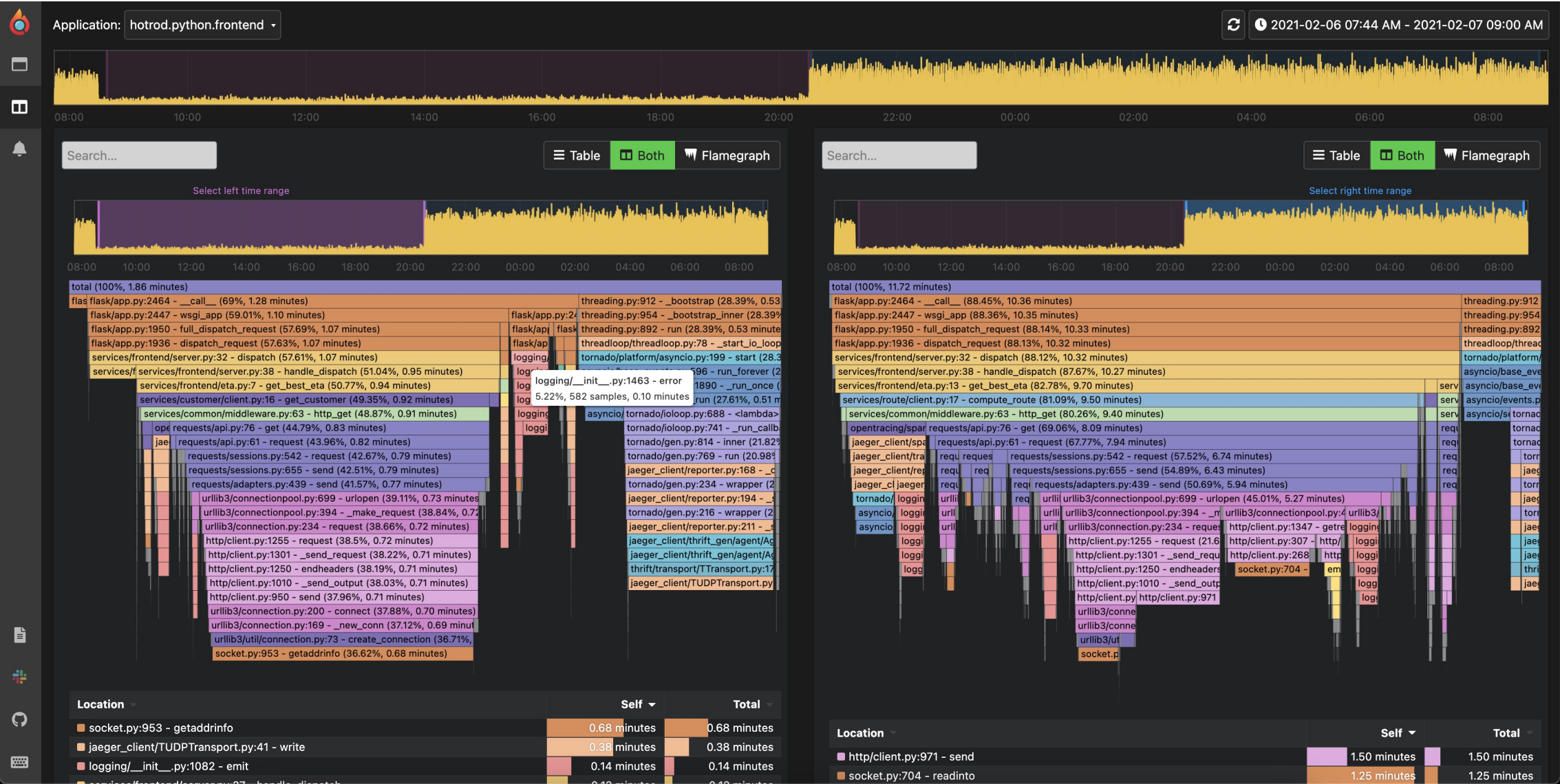
Непрерывное профилирование

parca



Pyroscope





 Empower Plant
Jane Schmidt

Projects

Issues

Discover

Performance

Alerts

Releases

User Feedback

Dashboards

Activity

Stats

Settings

Help

What's new

Collapse

All Projects

All Environments

Last 14 days

IntegrityError

pytest.runtest.call tests/sentry/api/endpoints/test_sentry_apps_stats

DJANGO-32

5

0

ASSIGNEE

Resolve

Ignore

Share

Open in Discover

Details

Activity

User Feedback

Attachments

Tags

Events

Merged Issues

Similar Issues

Event

bc74353509dn098y08hf

March 24, 2021 9:10:21 PM UTC

JSON (14.4kb)

?

charlotte_alameda@sentry.io

ID: 387936

Chrome

Version: 85.0.5353

Mac OSX

Version: 10.15.1

TAGS

browser

Chrome 89.0.4389

browser.name

Chrome

duration

16675

environment

prod

groupID

2292909355

interval

1000

level

warning

organization

557498

organization.slug

caffeinatedduo

os.name

Linux

plan

am1_f

plan.category

metered

plan.max_members

1

plan.name

Developer

plan.tier

am1

EXCEPTION (most recent call first)

App Only

Full

Raw

IntegrityError

IntegrityError('duplicate key value violates unique constraint "sentry_project_organization_id_slug_0ac4d5ae_uniq"\nDETAIL: Key (organization_id, slug)=(1538, climbing-beetle) already exists.\n')\nSQL: INSERT INTO "sentry_project" ("slug", "name", "forced_color", "organization_id", "public", "date_added", "status", "first_event", "flags", "platform") VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s) RETURNING "sentry_project"."id"

sentry/db/postgres/base.py in execute at line 75

70. @capture_transaction_exceptions

71. @auto_reconnect_cursor

72. @less_shitty_error_messages

73. def execute(self, sql, params=None):

74. if params is not None:

75. return self.cursor.execute(sql, clean_bad_params(params))

All Environments

LAST 24 HOURS

LAST 30 DAYS

LAST SEEN

3 minutes ago in release 25c62c66dd9e

FIRST SEEN

7 months ago in release 026451250466

Linked Issues

Link GitHub Issue

Link Jira Issue

Link Asana Issue

Link Azure DevOps Issue

Tags

browser

Chrome 89.0.4389

100%

browser.name

Chrome

100%

device

Mac

100%



```
1 func() {  
2     defer sentry.Recover()  
3     // do all of the scary things here  
4 }()
```



```
1 f, err := os.Open("filename.ext")  
2 if err != nil {  
3     sentry.CaptureException(err)  
4 }
```

FAQ:

Observability

- Трейсинг
- Мониторинг
- Логирование
- Анализ сбоев
- Непрерывное профилирование



Домашнее задание