Дипломная работа

курса «Архитектор программного обеспечения»

Этап 2

разработчик Драчёв О.Е.

11. Список ADR. (Architecture decision records - Записи архитектурных решений)

011 Разработка проекта – «Компания для привлечения людей к спортивному образу жизни»

Статус – accepted (принят).

Контекст:

Компания имеет большой штат разработчиков, говорящих на различных языках, и охотно адаптирует новые технологии для экспериментальных приложений. 90% всех систем, используемых в компании, расположены у облачных провайдеров, при этом нет одного выбранного провайдера — используется то, что больше подходит под конкретную задачу.

Решения:

Провести разработку проекта «Компания для привлечения людей к спортивному образу жизни».

Причины принятия решения:

Достижение бизнес целей.

Последствия:

При своевременных корректирующих действиях на риски, компания получит прибыль и выполнит бизнес цели.

Комплаенс (проверка соответствия):

Еженедельное подведение итогов разработка и демонстрация результатов.

Заметки:

ИТ архитектор Драчёв О.Е. 11.03.2022.

012 Архитектурный стиль – Microservices.

Статус – accepted (принят).

Контекст:

Рассмотрены предложения архитектурных стилей монолит или микросервис.
(010 Анализ и описание архитектурных опций и обоснование выбора).

Решения:

Будем использовать Архитектурный стиль – Microservices**.**

Причины принятия решения:

Архитектурный стиль – Монолит не соответствует пункту 8 НФТ.

Последствия:

Увеличивается общая сложность разработки. Разделение проекта на сервисы упрощает работу команд разработчиков над отдельными модулями проекта. Вместе с тем повышаются требования к взаимодействию.

Комплаенс (проверка соответствия):

Необходима разработка сценариев тестирования взаимодействия модулей системы.

Заметки:

ИТ архитектор Драчёв О.Е. 14.03.2022.

013 Применение распределенной архитектуры.

Статус – proposed (предложен).

Контекст:

Компания предполагает большой охват населения разных частей мира.

Решения:

Для решения поставленной задачи, будем использовать облака региональных операторов**.**

Причины принятия решения:

Снижение трафика локальных клиентов.

Последствия:

Постепенное расширение охвата локальных территорий.

Комплаенс (проверка соответствия):

Необходимо контролировать трафик в разрезе локализации. Поднимать новые центры при увеличении трафика выше контрольных показателей.

Заметки:

ИТ архитектор Драчёв О.Е. 14.03.2022.

014 Хранение данных.

Статус – accepted (принят).

Контекст:

Наша компания работает в различных регионах мира.
В результате работы активных клиентов формируется много клиентской информации. При хранении данных в едином хранилище возникает проблема передачи данных из различных регионов мира.

Решения:

Организовать региональные хранилища клиентских данных**.** Данные домена заказ будем хранить в региональном хранилище, а также передавать в центральное хранилище.

Причины принятия решения:

Экономия трафика на пересылке клиентских данных. Бизнес данные о заказах хранятся в 2х хранилищах.

Последствия:

Синхронизация коммерческих данных в 2х хранилищах повысит надежность системы.

Комплаенс (проверка соответствия):

Провести тестовую выборку данных из различных хранилищ и сравнить их.

Заметки:

ИТ архитектор Драчёв О.Е. 14.03.2022.

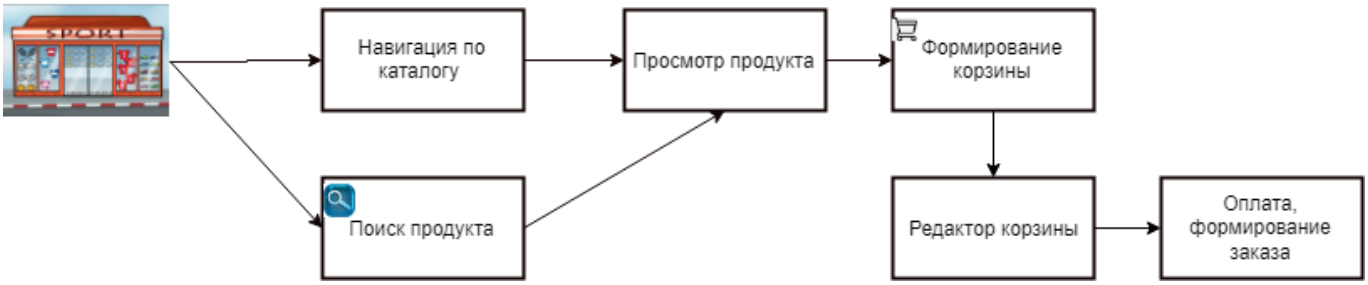
12. **Описание сценариев использования приложения**.

Стартовая страница приложения

Цель - увидеть состояние окружения и личной переписки.

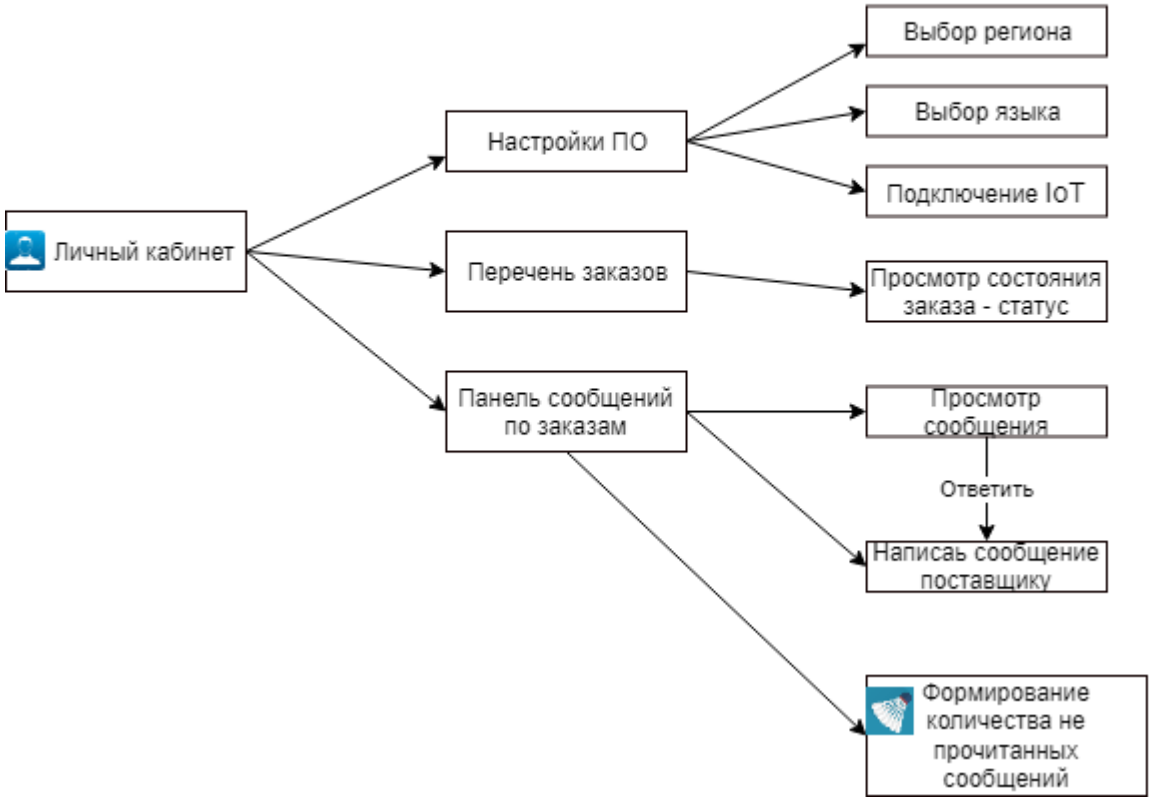
Страница "Витрина товаров"

Цель - просмотр товаров, поиск товаров и формирование корзины



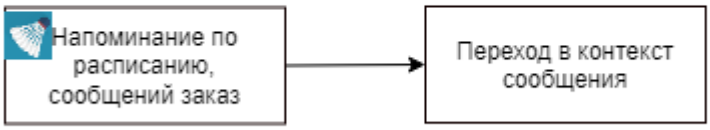
Страница "Спортивные показатели"
Цель - формирование планов тренировок, выбор группы по интересам, сбор материалов показателей организма, сохранение роликов и фотографий клиентов

Страница "Личный кабинет"
Цель - обеспечит работу клиента: с заказами, личной перепиской, настройками ПО (локация, языковая панель, подключение IoT)

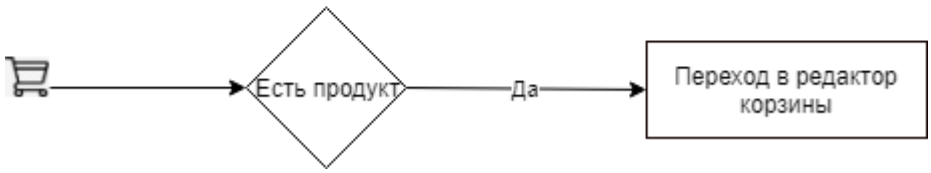


Страница "Региональных акций"
Цель - Демонстрация акций клиентам

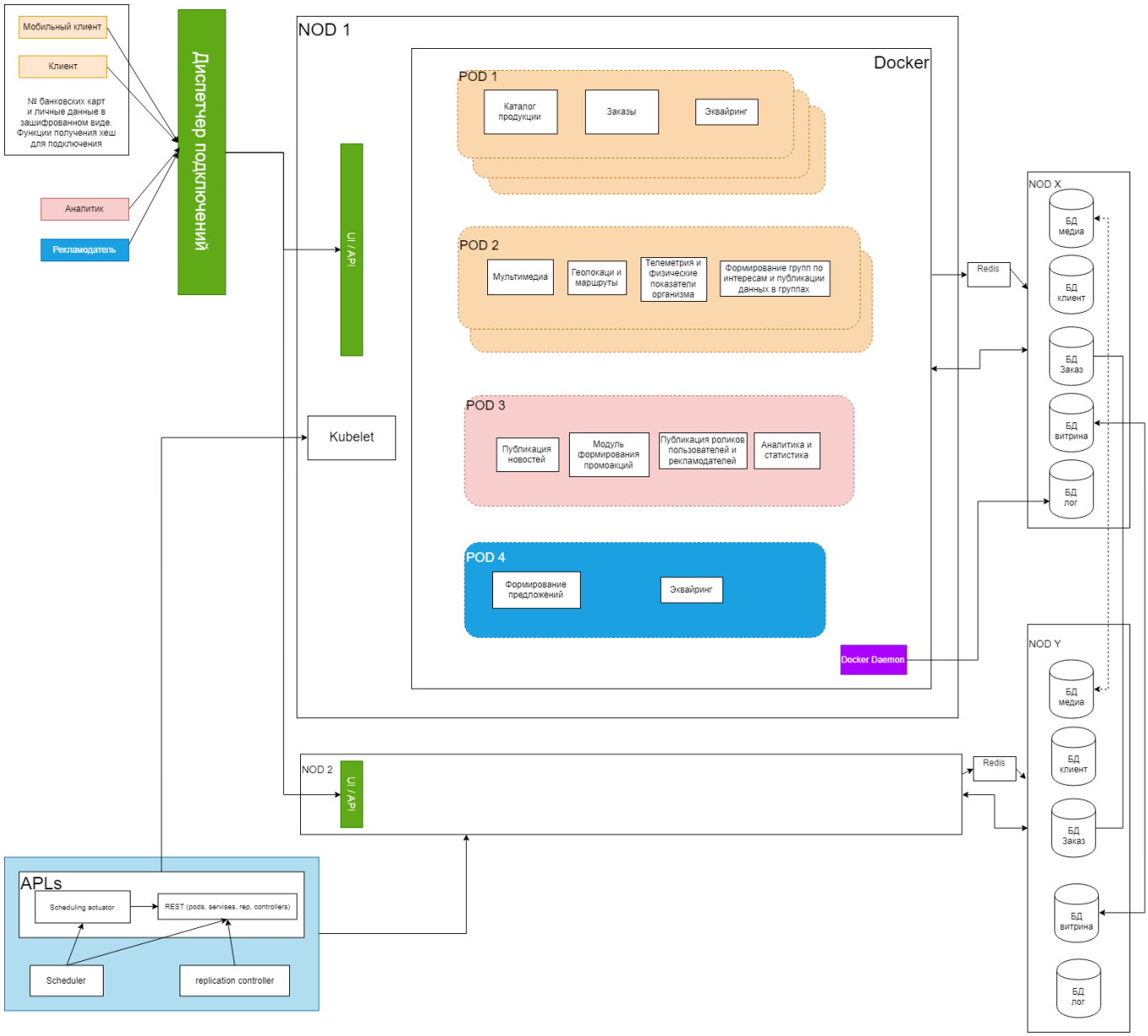
Страница "Индикатор сообщений"
Цель - Перейти на контекст сообщения



Страница "Корзина продукта"
Цель - Перейти в редактор корзины



13. **Базовая архитектура с учётом ограничений бизнес - требований, НФТ, выбранной архитектуры, адресация атрибутов качества**.

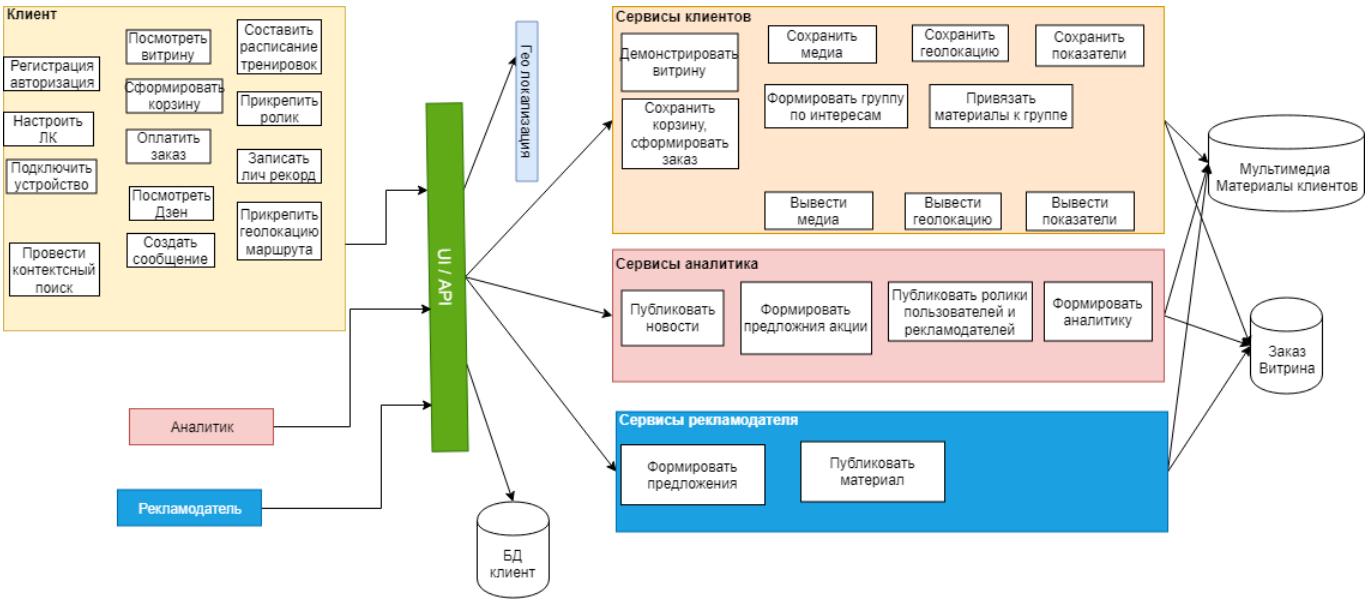


На рисунке представлена базовая схема проекта:

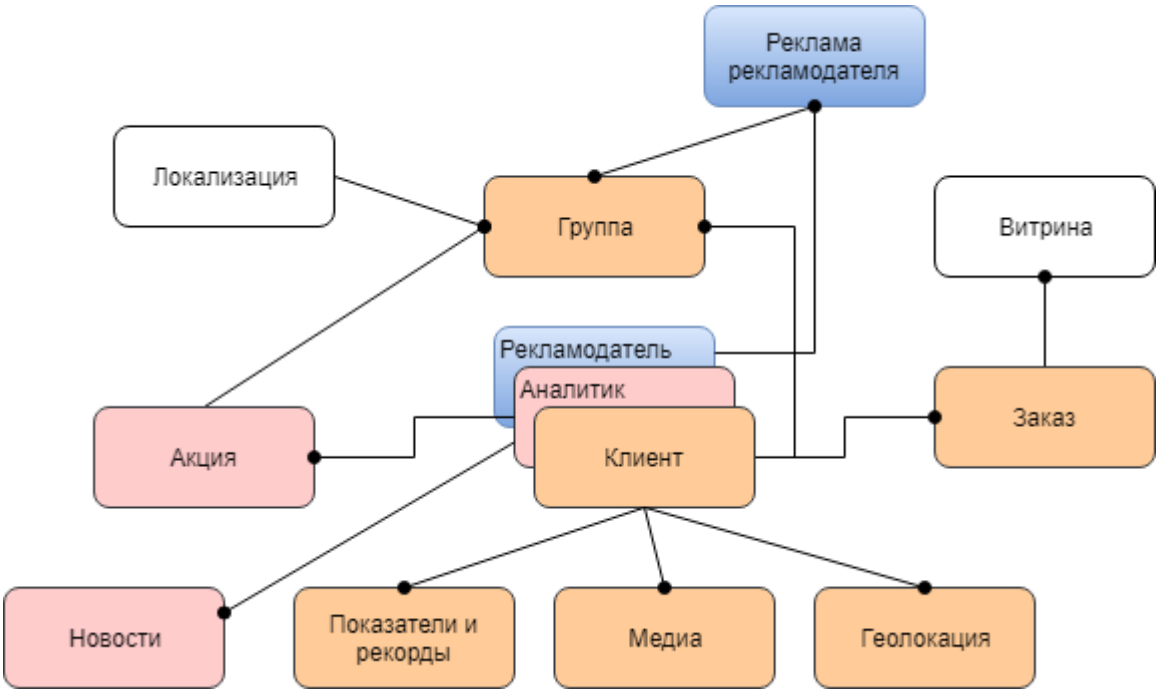
- Схема представляет распределенную систему под управлением Kubernetes.
- На схеме представлены основные блоки контейнеров.
- Система хранения данных и логов.
    - Данные разделены по типу.
    - БД Redis применяется для хранения ключей данных.
- Kubernetes обеспечивает оркестрацию подов при изменении нагрузки.
- Схема обеспечивает:
    - Хранение личных данных – фт п.1.
    - Оптимальное использование ресурсов ВТ (вычислительной техники) – нфт п. 8.
-

14. **Основные представления**:

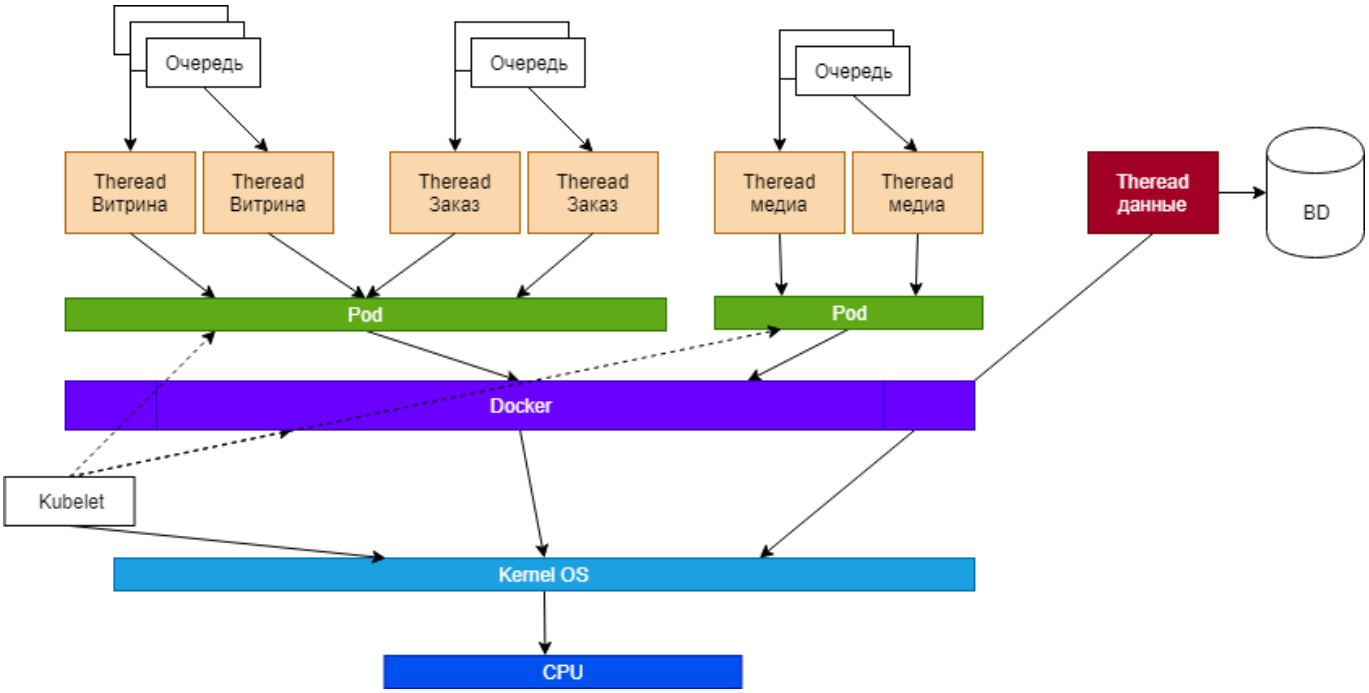
a. Функциональное.



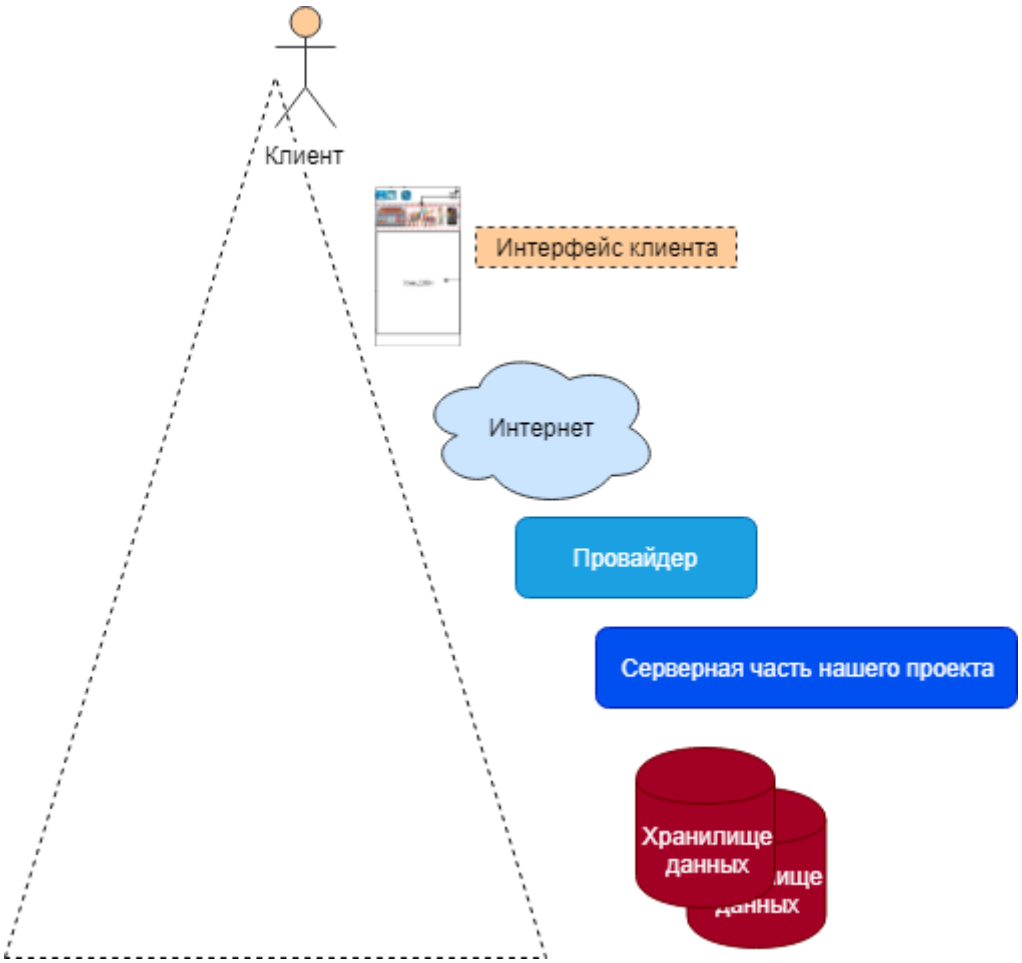
b. Информационное.



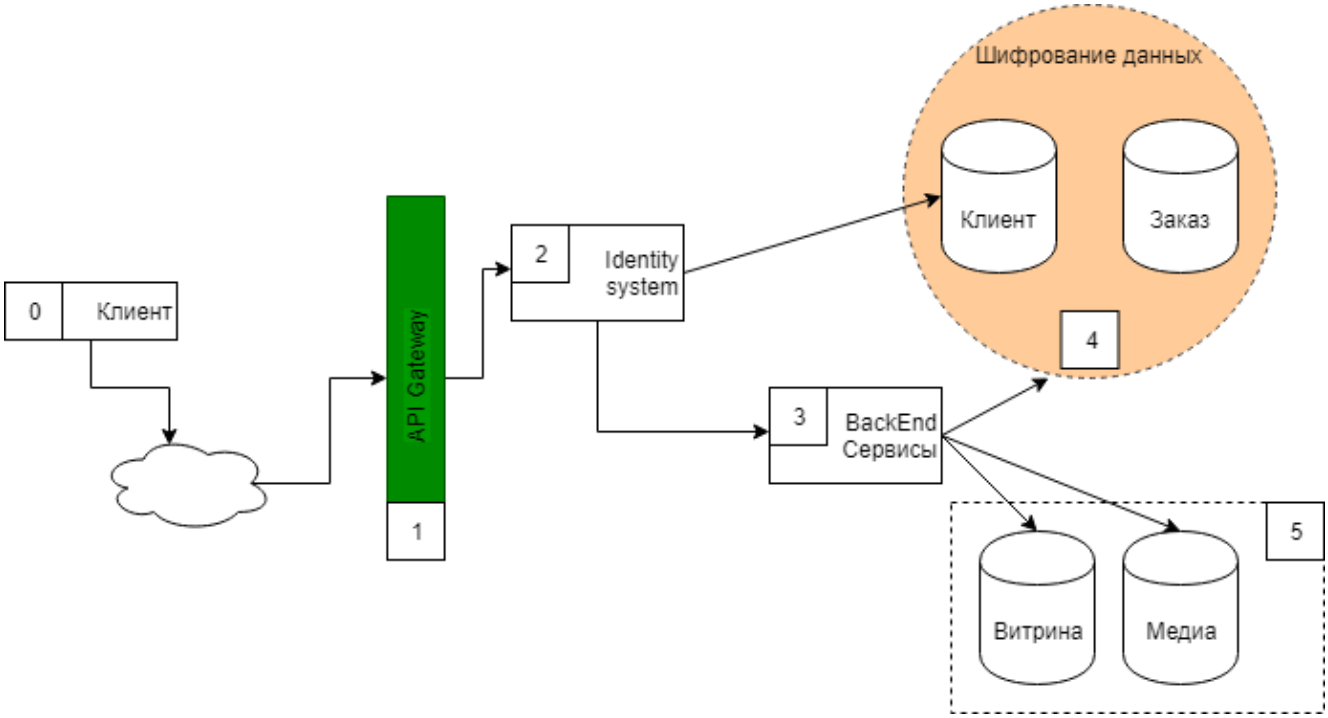
c. Многозадачность (concurrency).



d. Инфраструктурное.



e. Безопасность.



Элементы системы	№ вектора	Возможные векторы атак	Способы защиты от векторов атак
0,1	1	1. Инъекции — Injections	• Использование более безопасного API, исключающего использование интерпретатора. • Использование параметризованных запросов при кодировании. • Отделение команд от данных во избежание атак.
0,1	2	Sensitive Data Exposure (незащищённость конфиденциальных данных)	• Используя защищённый URL. • Использование надёжных и уникальных паролей. • Шифрование всей конфиденциальной информации, которую необходимо сохранить.
0,3	3	XML External Entities (XXE) Insecure Deserialization (внешние сущности XML, небезопасная десериализация)	• Использование менее сложных форматов данных, таких как JSON. • Обновление процессоров и библиотек XML. • Использование инструментов SAST.
2	4	Broken Access Control (нарушение контроля доступа)	• Удаление аккаунтов, которые больше не нужны или неактивны. • Отключение ненужных служб для снижения нагрузки на серверы. • Использование тестирования на проникновение.
Вся система	5	Небезопасная конфигурация (Security Misconfiguration)	• Использование динамического тестирования безопасности приложений (DAST). • Отключение использования паролей по умолчанию. • Следите за облачными ресурсами, приложениями и серверами.
0,1	6	Межсайтовый скриптинг – XSS (Cross Site Scripting)	• Использование соответствующих заголовков ответа. • Фильтрация ввода и кодирование вывода. • Использование политики безопасности контента.

			• Применение подхода с нулевым доверием к пользовательскому вводу.
2	7	Broken Authentication (нарушенная аутентификация)	• Реализация многофакторной аутентификации. • Защита учётных данных пользователя. • Отправка паролей через зашифрованные соединения.
0,2,3	8	Использование компонентов с известными уязвимостями	• Удаление всех ненужных зависимостей. • Использование виртуального исправления. • Использование компонентов только из официальных и проверенных источников.
3	9	Небезопасная десериализация	• Внедрение цифровых подписей. • Использование тестирования на проникновение. • Изоляция кода, который десериализует, и запуск его в средах с низким уровнем привилегий для предотвращения несанкционированных действий.
Вся система	10	Недостаточное ведение журнала и мониторинг	• Внедрение программного обеспечения для ведения журналов и аудита. • Создание эффективной системы мониторинга. • Думайте, как злоумышленник и используйте метод проверки на проникновение.

15. Анализ рисков созданной архитектуры, компромиссов.

№	Риск	Способы снижения рисков
1.	Нет провайдеров с требуемым набором услуг для развития нашего проекта.	• Пользоваться ближайшим провайдером (мирится с более дорогим трафиком) • Стимулировать развитие провайдеров (спонсорское участие).
2.	Потеря данных у локального провайдера	• Заключение договоров с более высокими требованиями хранения данных. • Создание подсистемы дублирования данных у ближайшего провайдера. Увеличение стоимости владения т.к. основная стоимость владения – это система хранения данных. • Создание дополнительных дата центров.
3.	Потеря ключей в БД Redis	• Восстановление ключей.
4.	Взлом сайда	• Применение последних практик безопасности
5.	Потеря данных на клиентских устройствах	• Частичное хранение личных данных, для восстановления связи.
6.	Остановка отдельных функций системы (не своевременное увеличение размеров хранилища)	• НФТ п.4. Оперативная отработка сообщений систем (Остановка записи данных). • Анализ логов, разработка мероприятий по достижению устойчивости системы.
7.		•

16. **Стоимость владения системой в первый, второй и пятый годы с учётом роста данных и базы пользователей**

Для решения данной задачи я решил использовать «Cloud» калькулятор яндекса (https://cloud.yandex.ru/prices).

За аналог расчета были использованы статистические показатели сети ВКонтакте:

- Использована статистика открытых источников.
- Зарегистрированные пользователи – 380 миллионов.
- Нас интересуют активные пользователи.
    - 1-й год 8 миллионов.
    - 2-й год 21 миллион.
    - 5-й год 50 миллионов.
- За основу расчета взяты следующие метрики (в месяц):
    - Объем исходящий трафик более 10гб (дальнейшее увеличение не учитывается).
    - Выделенный Ip адрес.
    - SSD диски для загрузки ПО
    - Ram = 6Гб
    - Get операции = 800000000
    - Post операции = 300000000
    - Размер хранилища = 500 Tb
    - Размер хранилища логов = 2 Tb
    - Redis Размер хранилища = 710 Гб

Итоги расчета первого года эксплуатации:

- Compute Cloud – 4072.95 руб.
- Object Storage – 1180183.29 руб.
- Kubernetes – 25891.67 руб.
- PostgreSQL – 34752.56 руб.
- Redis - 14339.37 руб.
- Elasticsearch - 40348.72 руб.
- Тех. Поддержка - 53,55 руб.
- Итого: **1 299 657,41 руб./мес.**

Рассчитать стоимость

Compute Cloud | Object Storage | Kubernetes® | PostgreSQL | ClickHouse | MongoDB | MySQL®
Redis™ | Elasticsearch | Greenplum® | SpeechKit | Translate | Техническая поддержка

Object Storage

Тип хранилища: Стандартное | Холодное

Размер хранилища: 500000 ГБ

GET-операции: 800 000 0C

POST-операции: 300 000 0C

Исходящий трафик: 20 ГБ

Добавить хранилище

Итого:

Compute Cloud × 1	4 072,95 ₽
Intel Ice Lake. 50% vCPU	1 843,20 ₽
Intel Ice Lake. RAM	806,40 ₽
Публичный IP-адрес	172,80 ₽
Быстрое сетевое хранилище (SSD)	1 250,55 ₽
Object Storage	1 180 183,29 ₽
Занятое место в стандартном хранилище	1 004 991,99 ₽
Стандартное хранилище — операции GET	31 196,10 ₽
Стандартное хранилище — операции POST	143 995,20 ₽
Kubernetes®	25 891,67 ₽
Managed Kubernetes. Zonal Master - small	6 336,00 ₽
Intel Ice Lake. 50% vCPU	3 686,40 ₽
Intel Ice Lake. RAM	1 612,80 ₽
Публичный IP-адрес	345,60 ₽
Быстрое сетевое хранилище (SSD)	13 910,87 ₽
PostgreSQL	34 752,56 ₽
PostgreSQL. Intel Cascade Lake. 100% vCPU	2 606,40 ₽
PostgreSQL. Intel Cascade Lake. RAM	5 644,80 ₽
Быстрое сетевое хранилище — PostgreSQL	26 501,36 ₽
Redis™	14 339,37 ₽
Redis. Intel Cascade Lake. 100% vCPU	2 419,20 ₽
Redis. Intel Cascade Lake. RAM	2 592,00 ₽
Быстрое сетевое хранилище — Redis	9 328,17 ₽
Elasticsearch	40 348,72 ₽
Elasticsearch. Intel Cascade Lake. 100% vCPU	2 419,20 ₽
Elasticsearch. Gold. Intel Cascade Lake. RAM	14 169,60 ₽
Быстрое сетевое хранилище — Elasticsearch	23 587,12 ₽
Публичный IP-адрес - Elasticsearch	172,80 ₽
Техническая поддержка	0,00 ₽
Техническая поддержка – тарифный план Базовый	0,00 ₽
Исходящий трафик из Object Storage в интернет	15,30 ₽
Исходящий трафик в интернет	53,55 ₽

1 299 657,41 ₽ в месяц

Второй год владения. Произойдет увеличение функций и возможностей пользователя, потребуются новые ресурсы.

- За основу расчета взяты следующие метрики (в месяц):
    - Объем исходящий трафик более 10гб (дальнейшее увеличение не учитывается).
    - Выделенный Ip адрес.
    - SSD диски для загрузки ПО
    - Ram = 64Гб
    - Get операции = 2400000000
    - Post операции = 1000000000
    - Размер хранилища = 4000 Tb
    - Размер хранилища логов = 2 Tb
    - Redis Размер хранилища = 710 Гб

Кроме увеличения вычислительных мощностей, нужно увеличить количество нодов. Устанавливаем 2 дополнительных нода. Снижение стоимости можно ожидать за счет подключения холодного хранилища.
Итого: 4 867 369,05 * 3 = **14 602 107,15 руб**.

Рассчитать стоимость

Compute Cloud | Object Storage | Kubernetes® | PostgreSQL | ClickHouse | MongoDB | MySQL®
Redis™ | Elasticsearch | Greenplum® | SpeechKit | Translate | Техническая поддержка

Compute Cloud

Операционная система: Ubuntu 20.04 LTS

Платформа: Intel Ice Lake

Гарантированная доля vCPU: 20% | 50% | 100%

Количество vCPU: 16 (2 — 96)

Объём RAM: 64 ГБ (16 ГБ — 256 ГБ)

Прерываемая ВМ: ☐

Публичный IP-адрес: ☑

Исходящий трафик: 20 ГБ

Диск – 1 (Загрузочный): ○ HDD ● SSD

Итого:

Compute Cloud × 1	26 421,75 ₽
Intel Ice Lake. 100% vCPU	12 096,00 ₽
Intel Ice Lake. RAM	12 902,40 ₽
Публичный IP-адрес	172,80 ₽
Быстрое сетевое хранилище (SSD)	1 250,55 ₽
Object Storage	4 690 044,71 ₽
Занятое место в стандартном хранилище	4 116 453,41 ₽
Стандартное хранилище — операции GET	93 596,10 ₽
Стандартное хранилище — операции POST	479 995,20 ₽
Kubernetes®	36 922,07 ₽
Managed Kubernetes. Zonal Master - small	6 336,00 ₽
Intel Ice Lake. 100% vCPU	9 072,00 ₽
Intel Ice Lake. RAM	7 257,60 ₽
Публичный IP-адрес	345,60 ₽
Быстрое сетевое хранилище (SSD)	13 910,87 ₽
PostgreSQL	41 764,95 ₽
PostgreSQL. Intel Cascade Lake. 100% vCPU	2 606,40 ₽
PostgreSQL. Intel Cascade Lake. RAM	5 644,80 ₽
Быстрое сетевое хранилище — PostgreSQL	33 513,75 ₽
Redis™	25 801,17 ₽
Redis. Intel Cascade Lake. 100% vCPU	2 419,20 ₽
Redis. Intel Cascade Lake. RAM	2 592,00 ₽
Быстрое сетевое хранилище — Redis	20 789,97 ₽
Elasticsearch	46 307,30 ₽
Elasticsearch. Intel Cascade Lake. 100% vCPU	2 419,20 ₽
Elasticsearch. Gold. Intel Cascade Lake. RAM	14 169,60 ₽
Быстрое сетевое хранилище — Elasticsearch	29 545,70 ₽
Публичный IP-адрес - Elasticsearch	172,80 ₽
Техническая поддержка	0,00 ₽
Техническая поддержка – тарифный план Базовый	0,00 ₽
Исходящий трафик из Object Storage в интернет	15,30 ₽
Исходящий трафик в интернет	91,80 ₽

4 867 369,05 ₽ в месяц

Пятый год владения – предполагаем расширение за счет периферийных нодов, в количестве 6 штук. Основные затраты - это системы хранения данных.
Стоимость 1 го месяца системы на пятый год эксплуатации.
Итого: 4 867 369,05 * 7 = **34 071 583,35 руб/месяц**.