Департамент образования и науки города Москвы

Государственное автономное образовательное учреждения высшего образования города Москвы «Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

Лабораторная работа 3.1.

Проектирование архитектуры хранилища больших данных.

Выполнил студент группы АДЭУ-221

Джамалова Сабина Шахиновна

Проверил доцент

Босенко Тимур Муртазович

Москва

2025

Оглавление

Введение

Цель работы: разработка комплексной архитектуры хранилища больших данных, обоснование выбора технологического стека и визуализация потоков данных.

Вариант задания: 6. Медицинская клиника: анализ электронных медицинских карт (EMR), обработка изображений (МРТ, КТ) для помощи в диагностике, прогнозировании эпидемий. Источники: данные EMR, лабораторные тесты, медицинские изображения.

Выполнение:

Цель: Построить отказоустойчивую, масштабируемую архитектуру хранилища больших данных для анализа EMR, обработки медицинских изображений (DICOM), лабораторных тестов и прогнозирования эпидемий; поддержать batch- и real-time-аналитику, ML-пайплайны и соблюдение требований конфиденциальности (HIPAA/GDPR/152-ФЗ).

Для стартапа:

Стартап разрабатывает систему аналитики и поддержки принятия решений для медицинской клиники.
Система собирает данные из электронных медицинских карт (EMR), лабораторных систем и диагностического оборудования (например, МРТ/КТ), а затем анализирует их, чтобы:

- улучшить точность диагностики,
- выявлять повторяющиеся случаи и закономерности,
- прогнозировать вспышки заболеваний и загруженность лабораторий.

1. Анализ требований:

1.1. Бизнес-сценарий

Медицинская клиника хочет построить унифицированную платформу для хранения и аналитики данных с целью:

- Анализа электронных медицинских карт (EMR) для улучшения обслуживания пациентов и отчетности.
- Хранения и обработки медицинских изображений (МРТ, КТ) для помощи в диагностике (включая предобработку и подачу в модели ML).
- Обработки лабораторных тестов и интеграции с EMR.
- Прогнозирования локальных вспышек заболеваний (эпидемиология).
- Поддержки как пакетной аналитики, так и real-time/near-real-time оповещений (например, тревожные показатели пациентов).

1.2. Источники данных

- EMR (электронные медицинские карты): реляционные БД клиники — структурированные данные (пациенты, визиты, диагнозы, назначения).
- Лабораторные тесты: структурированные данные, ежедневный и интерактивный поток результатов.
- Медицинские изображения (DICOM — МРТ/КТ): неструктурированные большие файлы (от десятков МБ до сотен МБ на исследование).
- Потоки мониторинга пациентов (временные ряды): потоковые данные от приборов (heart rate, SpO2) — возможно 1–1000 событий в секунду в зависимости от нагрузки.
- Внешние источники: эпидемиологические данные, демография, погодные данные (API) — полуструктурированные/структурированные.

1.3. Типы данных, объёмы и скорость

- EMR: 100–500 тыс. записей в год; ~10 GB/год структурированных данных.
- Лабораторные тесты: 1–5 млн записей в год; ~50 GB/год.
- Медицинские изображения: 50–200 тыс. исследований в год; средний размер 200 MB => 10–40 TB/год.
- Потоки мониторинга: пики до 5k событий/сек при полном онлайне; среднемесячный объем — 5–50 GB/мес.

1.4. Бизнес-цели и аналитические требования

- Отчеты: стандартные BI-отчеты (ежедневные/еженедельные) по загрузке клиники, времени ожидания, результатам лечения.
- ML: модели диагностики по изображениям, моделирование риска госпитализации, прогнозирование вспышек.
- Соответствие нормативам: HIPAA/GDPR— защита данных пациентов, аудит доступа, шифрование.

1. Требования к данным для медицинской клиники

1.1 Объем данных

- Ожидаемый объём: 20–40 ТБ в год (основной вклад - медицинские изображения DICOM: МРТ, КТ, УЗИ).
- Рост: 20–30% ежегодно. (В литературе по здравоохранению часто приводят оценки роста 20–40% в год для среднего уровня клиник, особенно если есть активная работа с изображениями.)

1.2 Скорость получения данных

- Системы мониторинга пациентов (кардиомониторы, пульсоксиметры): поток в реальном времени, до 500 событий в секунду.

- PACS (снимки МРТ/КТ): загрузка партиями, несколько сотен исследований в день.
- Лабораторные системы (LIMS): обновления ежедневно/почасово.
- Электронные медицинские карты (EMR): транзакции в режиме работы врача (онлайн-запись, назначение лекарств).
- Внешние API (например, эпидемиологические данные): обновление раз в сутки.

1.3 Типы данных

- Структурированные: данные EMR, лабораторные тесты, расписание приёма (≈30%).
- Полуструктурированные: HL7/FHIR сообщения, JSON от медицинских приборов, логи систем (≈40%).
- Неструктурированные: DICOM-изображения, текстовые заключения врачей (≈30%).

1.4 Требования к обработке

- Оперативный мониторинг состояния пациентов - в реальном времени.
- Формирование отчётности для руководства - ежедневно/еженедельно.
- Аналитика эффективности лечения и прогнозирование заболеваний - ежеквартально.
- Поддержка ML-моделей (диагностика по снимкам, прогноз риска осложнений) - обучение пакетное, применение онлайн.

1.5 Доступность данных

- Время отклика для критичных запросов (например, история болезни) - ≤5 секунд.

- Доступность системы: 99.9% (допустимый простой ≤8.8 часов в год).
  (экономически и технически обеспечить 99.99% или 99.999% (четыре-пять «девяток») слишком дорого для средней клиники.)

1.6 Безопасность данных

- Шифрование при хранении и передаче (TLS, AES-256).
- Многофакторная аутентификация для врачей и администраторов.
- Аудит всех действий с медицинскими данными.
- Соответствие требованиям 152-ФЗ «О персональных данных» и международным нормам (GDPR, HIPAA).

2. Выбор компонентов архитектуры:

2.0. Источники данных

- Электронные медицинские карты (EMR) - структурированные данные о пациентах, назначениях, диагнозах.
- Лабораторные системы (LIMS) - результаты анализов.
- Медицинские изображения (PACS/DICOM) - МРТ, КТ, УЗИ, рентген.
- Телеметрия медицинских приборов и IoT - кардиомониторы, пульсоксиметры, мобильные приложения пациентов.
- Внешние API - эпидемиологические сводки, данные Минздрава, погода и экология (для прогнозов заболеваний).
- Журналы активности (лог-файлы систем EMR, PACS) - для аудита, анализа нагрузки и безопасности.

2.1. Слой сбора данных (Ingestion)

- Apache Kafka - для потоковой телеметрии, событий EMR, результатов лабораторий и стриминга метаданных изображений.
- Logstash — для загрузки структурированных и полуструктурированных данных (таблицы из EMR, LIS, внешние API).

- Orthanc (DICOM connector) - для приёма, хранения и аннотирования медицинских изображений.

Kafka обеспечивает высокую пропускную способность, задержку на миллисекунды и поддерживает интеграцию с обработчиками (Flink, Spark).

2.2. Слой хранения (Storage)

- MinIO — объектное хранилище (аналог S3) для сырых данных: медицинские изображения, большие архивы, файлы API.
- PostgreSQL — основное реляционное хранилище для данных EMR и лабораторных тестов (OLTP).
- ClickHouse - для низколатентной аналитики и дашбордов в реальном времени.

2.3. Слой обработки (Processing)

- Apache Flink - стриминговая обработка в реальном времени.
- Apache Spark (Databricks) - пакетная обработка, ETL, подготовка признаков и обучение моделей.
- MLflow + Seldon - управление ML-экспериментами и деплой inference-сервисов (анализ изображений, предсказания).

2.4. Слой аналитики и визуализации (Analytics & Visualization)

- Tableau - корпоративная отчётность и BI.
- Apache Superset - для медицинских дашбордов.

2.5. Оркестрация и автоматизация (Orchestration)

- Apache Airflow - управление ETL-пайплайнами, расписания и зависимости.

2.6. Управление данными и каталог (Data Governance)

- OpenMetadata - каталог данных, lineage, поиск и контроль качества.

3. Проектирование архитектуры:

3.1. Компоненты для обеспечения безопасности, мониторинга и логирования

Безопасность:

- Шифрование: TLS 1.2+ при передаче, AES-256, TDE для баз данных. (см. Таненбаум, *Распределённые системы*, гл. 9 - «Безопасность»).
- Управление доступом: IAM + RBAC (role); интеграция с корпоративным IdP, многофакторная аутентификация. (Таненбаум, гл. 9.3 «Контроль доступа»).
- Аудит: централизованный сбор логов, хранение журналов доступа, immutable storage.

Мониторинг и логирование:

- Prometheus + Grafana - метрики систем (Kafka, Flink, Spark, K8s), визуализация SLO.
  (Prometheus собирает метрики (нагрузка CPU, задержки в Kafka, время отклика Spark).
  Grafana рисует красивые дашборды (например, «задержка обработки телеметрии от кардиомониторов»).)
- ELK Stack (Elasticsearch + Logstash + Kibana) - централизованные логи и аудит действий.
  (Logstash собирает логи.
  Elasticsearch индексирует их, чтобы можно было быстро искать.
  Kibana — рисует отчёты по логам (например, сколько раз в день заходили в систему).)
- OpenTelemetry — трассировка межсервисных вызовов (распределённый трейсинг).

3.2. Стратегия масштабирования и отказоустойчивости

Масштабирование:

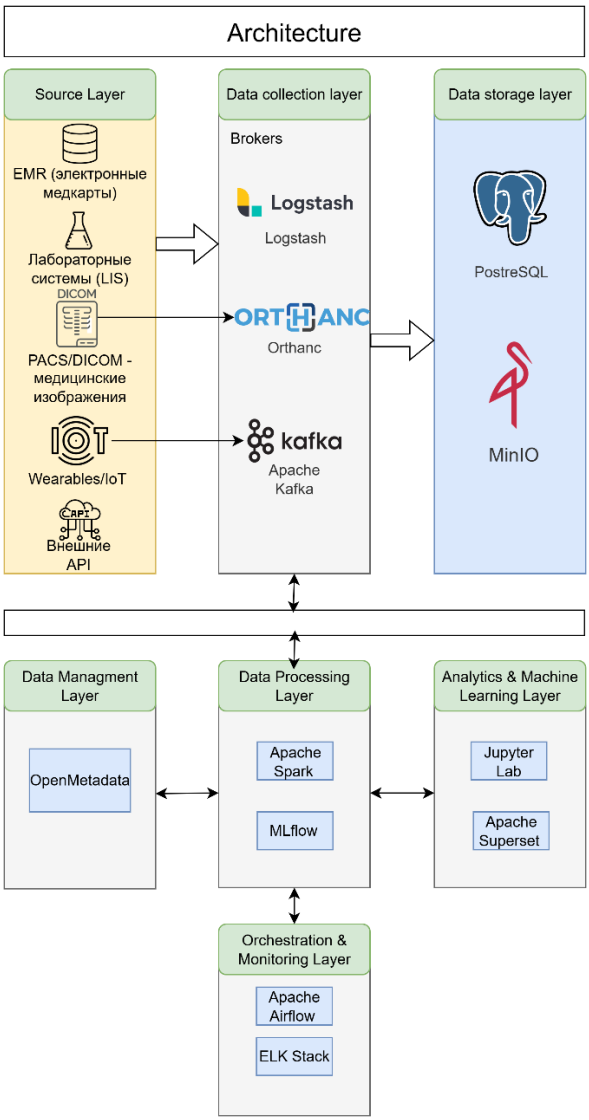
- Горизонтальное: Kafka (партиции и брокеры), Flink (parallelism, checkpointing (один воркер падает, задача переносится на другой)), Spark (autoscaling), Kubernetes (автоматически поднимает контейнеры).
- Вертикальное: временное увеличение ресурсов для master-узлов (Kafka Controller, Spark Driver, Flink JobManager) и сервисов.

Отказоустойчивость:

- Репликация: данные в Kafka, ClickHouse и MinIO реплицируются между зонами доступности.
- Checkpointing и recovery: Flink и Spark сохраняют состояние (state) в MinIO для восстановления.

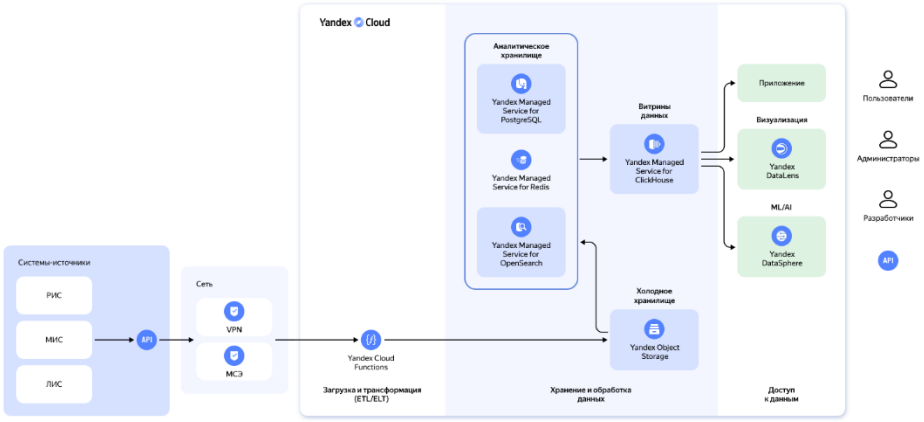
  (Flink: каждые N секунд делает снимок состояния (state snapshot) и сохраняет его в MinIO. При сбое задача продолжается с последнего снимка. Spark: аналогично использует контрольные точки)
- Disaster Recovery (DR): кросс-региональная репликация MinIO; резервные копии ClickHouse; регулярные тесты восстановления.
- SLA: 99.9% доступности (время простоя ≤8.8 ч/год);
  - RTO (время восстановления) ≤1 час для критичных сервисов;
  - RPO (точка восстановления) ≤5 минут для EMR-данных.

4. Создание диаграммы архитектуры:



Без кликхауса, потому что данные напрямую идут из PostgreSQL в Apache Superset.

На основе статьи и работы Сеченовского университета и Яндекса: https://habr.com/ru/companies/yandex_cloud_and_infra/articles/861698/



5. Описание компонентов и обоснование выбора:

Сбор данных (Ingestion)

- Apache Kafka: для потоковых данных (приборы, алерты). Выбран вместо RabbitMQ из-за лучшей масштабируемости.
- Logstash: для загрузки данных из EMR, LIS, API. Выбран вместо Airbyte, т.к. проще развернуть локально.
- Orthanc: специализированный DICOM-сервер. Выбран за открытость и простую интеграцию с MinIO.

Хранение (Storage)

- PostgreSQL: реляционные данные (EMR, LIS). Широко поддерживается, open-source.
- MinIO: основное хранилище для файлов и изображений. Выбран вместо S3, так как разворачивается локально.
- ClickHouse: для аналитики в реальном времени (дашборды ICU). Выбран вместо Snowflake.

Обработка (Processing)

- Apache Flink: для обработки триминг алертов. Выбран вместо Spark Streaming из-за true streaming (не micro-batching).
- Apache Spark: для пакетной обработки, ETL и ML. Отраслевой стандарт для работы с большими объемами данных.

Оркестрация, ML и Аналитика

- Apache Airflow: для оркестрации пайплайнов. Мощное сообщество.
- MLflow + Seldon Core: для управления ML-жизненным циклом и диплоя моделей. Open-source альтернатива облачным сервисам.

- Tableau / Superset: для визуализации. Tableau — для корпоративной отчетности, Superset (open-source) — для кастомных дашбордов.
- OpenMetadata: каталог данных для управления и аудита. Критично для соблюдения HIPAA/GDPR.

6. Анализ потенциальных проблем и их решений:

1. Проблема: Высокий объём хранения медицинских изображений (DICOM)

- **Риск:** Медицинские изображения занимают наибольшую часть хранилища (10–40 ТБ в год). При росте числа пациентов и внедрении новых методов диагностики (например, МРТ с высокой детализацией) нагрузка на MinIO может резко увеличиться. Это приведёт к росту затрат на дисковое пространство и снижению производительности при аналитике и обучении ML-моделей.
- **Решение:**
  - *Стратификация хранения:* в MinIO можно организовать «классы хранения» (горячее/холодное). Новые снимки хранятся на быстрых SSD, старые архивируются на более дешёвых дисках (SATA) или внешних носителях.
  - *Оптимизация работы с данными:* аналитика и дашборды работают не с полными DICOM-файлами, а с метаданными (Orthanc) или превью (JPEG/PNG). Полные изображения извлекаются только для задач машинного обучения или телемедицины.
  - *Сжатие и дедупликация:* использование встроенных средств сжатия в Orthanc и MinIO для снижения затрат на хранение.

2. Проблема: Безопасность и защита персональных данных (152-ФЗ, медицинская тайна)

- **Риск:** Архитектура состоит из нескольких компонентов (Kafka, PostgreSQL, MinIO, ClickHouse, Orthanc). Каждый из них может стать точкой атаки. Нарушение конфиденциальности персональных медицинских данных приведёт к серьёзным правовым последствиям.

- **Решение:**
    - *Сквозное шифрование:* TLS для всех соединений (Kafka → Postgres, Kafka → MinIO, ClickHouse → BI). Шифрование данных «на диске» (Postgres TDE, MinIO SSE).
    - *Управление доступом:* использование RBAC (Role-Based Access Control) с разграничением прав доступа врачей, администраторов и аналитиков. Orthanc поддерживает контроль доступа к исследованиям.
    - *Аудит и мониторинг:* централизованный логинг через ELK и аудит обращений (кто и когда открывал данные). Для соблюдения 152-ФЗ — протоколирование операций с персональными данными.
    - *Анонимизация/деперсонализация:* при выгрузке данных для аналитиков и ML-моделей из PostgreSQL и Orthanc — удаление ФИО, паспортных данных, адресов и другой идентифицирующей информации.

Выводы:

В ходе лабораторной работы была спроектирована комплексная, масштабируемая и отказоустойчивая архитектура хранилища больших данных для медицинской клиники.

Ключевым результатом стал выбор технологического стека, оптимально соответствующий требованиям медицинской клиники: использование Apache Kafka обеспечило надёжный сбор и маршрутизацию потоковых данных (телеметрия, события EMR, лабораторные результаты), а связка Apache Flink и Apache Spark дала возможность сочетать потоковую и пакетную аналитику. Для хранения медицинских изображений выбран MinIO (аналог S3 с локальным развёртыванием), а для аналитики в реальном времени — ClickHouse, что позволило обеспечить баланс между производительностью и затратами.

Были учтены критически важные аспекты информационной безопасности и соответствия нормам (HIPAA, GDPR, 152-ФЗ) за счет сквозного шифрования, централизованного аудита и управления доступом. Анализ потенциальных рисков (таких как стоимость хранения изображений и операционная сложность) показал, что архитектура обладает необходимыми механизмами для их минимизации.

Таким образом, разработанная архитектура является сбалансированным решением, способным поддерживать как текущие аналитические задачи клиники, так и будущий рост объемов данных и сложности аналитических сценариев.