Департамент образования города Москвы

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

Институт цифрового образования
Департамент информатики, управления и технологий

ПРАКТИЧЕСКАЯ РАБОТА №3

по дисциплине «Инструменты для хранения и обработки больших данных»
Направление подготовки 38.03.05 – бизнес-информатика
Профиль подготовки «Аналитика данных и эффективное управление»
(очная форма обучения)

Выполнила:
Студентка группы АДЭУ-221
Вознесенская В. Е.

Проверил:
Босенко Т. М., доцент

Москва
2025

КОМПЛЕКСНАЯ АРХИТЕКТУРА ХРАНИЛИЩА БОЛЬШИХ ДАННЫХ ДЛЯ КРУПНОГО ОНЛАЙН-РИТЕЙЛЕРА

Цель работы: разработать комплексную архитектуру хранилища больших данных для предложенного бизнес-сценария, обосновать выбор технологического стека и визуализировать потоки данных.

Вариант 1

Крупный онлайн-ритейлер: анализ поведения пользователей в реальном времени, прогнозирование спроса, персонализация рекомендаций. Источники: кликстрим с сайта/приложения, транзакции, отзывы клиентов, данные из CRM.

1. Описание бизнес-процесса

Бизнес-процесс: «Персональные рекомендации пользователю»

1.1 Цель процесса

Показывать каждому посетителю сайта/мобильного приложения персональные товары и акции, чтобы увеличить клики, конверсии и средний чек. Рекомендации должны работать в реальном времени (или near-real-time) и улучшаться по мере получения новых данных.

1.2 Участники процесса

— пользователь — заходит на сайт, просматривает товары, совершает покупки;
— сайт или приложение — показывает товары и собирает информацию о действиях пользователя;
— команда данных — собирает, хранит и анализирует информацию;
— команда маркетинга — использует результаты для акций и персональных предложений;
— аналитики — следят за показателями и оценивают эффективность рекомендаций

1.3 Источники данных

— кликстрим (clickstream) — все действия пользователя на сайте или в приложении: какие страницы он смотрит, куда нажимает, что добавляет в корзину;

— транзакции — информация о заказах: что куплено, когда и на какую сумму;

— отзывы — тексты и оценки, которые пользователи оставляют о товарах;

— CRM — данные о клиентах: профиль, история покупок, уровень лояльности и т. д.

1.4 Выход

— список персональных рекомендаций для каждого пользователя;

— статистика по тому, как часто пользователи кликают на рекомендации и покупают;

— отчёты и дашборды для аналитиков и менеджеров.

1.5 Основные этапы процесса

На рисунке 1.1 кратко показаны основные этапы данного бизнес-процесса.

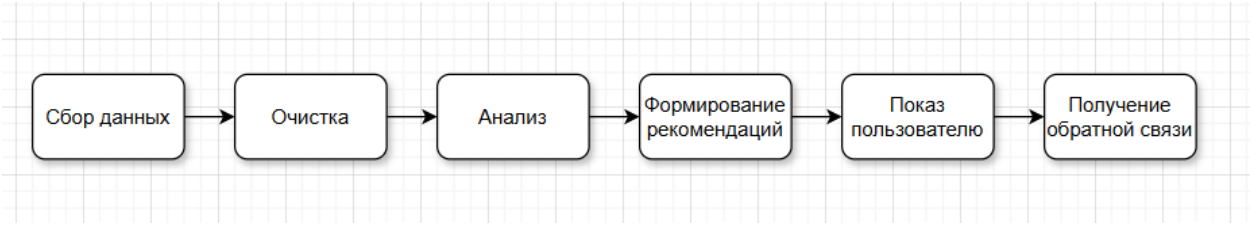

Рисунок 1.1 Основные этапы бизнес-процесса «Персональные рекомендации пользователю»

2. Анализ требований

2.1 Источники данных

Согласно бизнес-процессу у нас есть несколько источников данных, они представлены в таблице 2.1.

Таблица 2.1. Источники данных

Источник данных	Тип данных	Скорость поступления	Объем данных	Примечания/Использование
Кликстрим (clickstream)	Потоковые данные (события пользователя)	Реальное время / near-real-time	Высокий (много действий на сайте)	Используется для формирования персональных рекомендаций в реальном времени
Транзакции (заказы)	Структурированные (таблицы с покупками)	Пакетная загрузка (batch)	Средний	Используется для анализа покупок, формирования моделей рекомендаций
Отзывы	Неструктурированные (текст) + структурированные (оценки)	Периодически / batch	Небольшой — средний	Анализ отзывов для улучшения рекомендаций и оценки товаров
CRM (клиенты)	Структурированные (профиль, история покупок, уровень лояльности)	Периодически / batch	Небольшой	Используется для персонализации предложений и сегментации пользователей

Вывод по источникам: данные разнообразные по типу и скорости поступления. Кликстрим — потоковые и высокоскоростные, остальные — чаще загружаются пакетами (batch).

2.2 Бизнес-цели

— персональные рекомендации (рекомендации должны работать в реальном времени или почти в реальном времени (near-real-time), список товаров или акций, которые интересны каждому пользователю);

— аналитика эффективности(отслеживать клики на рекомендации и покупки, строить отчеты и дашборды для команды маркетинга и аналитиков;

— использование моделей машинного обучения (ML): (ML модели будут анализировать поведение пользователей и прогнозировать, что им может понравиться, модели должны обновляться по мере поступления новых данных).

Вывод: основная цель — увеличить доход компании за счёт персональных рекомендаций, которые могут увеличить средний чек у каждого пользователя.

2.3 Резюме требований

Скорость обработки данных:

— Кликстрим — real-time/near-real-time

— Транзакции, отзывы, CRM — batch (обновление периодически)

Типы данных:

— Структурированные: транзакции, CRM

— Потоковые: кликстрим

— Неструктурированные: тексты отзывов

Цели аналитики:

— Реальные рекомендации пользователям в режиме почти реального времени

— Отчеты и дашборды для аналитиков и маркетинга

3. Выбор компонентов архитектуры

3.1 Слой сбора данных (Ingestion)

Выбор: Apache Kafka

Обоснование:

— позволяет собирать потоковые данные в реальном времени (кликстрим);

— надежная система, поддерживает масштабирование, легко подключается к другим инструментам обработки;

— можно интегрировать с batch-данными через коннекторы (например, для транзакций или CRM).

Альтернатива: Amazon Kinesis — хорош для облака AWS, но Kafka универсальнее и не привязана к конкретному облаку.

3.2 Слой первичного хранения (Storage)

Выбор: PostgreSQL

Обоснование:

— используется для хранения сырых данных (raw data) после поступления из Kafka и других источников;

— обеспечивает надёжность, транзакционность и быстрый доступ для ETL-процессов;

Плюсы: открытое ПО, стабильность, возможность разворачивания локально.

3.3 Слой хранилища данных (Data Warehouse, DWH)

Выбор: MinIO + Delta Lake

Обоснование:

— MinIO — аналог S3, подходит для локального или частного облака, легко разворачивается в России;

— Delta Lake добавляет управление версиями, поддержку транзакций и совмещение batch и streaming данных;

— используется для построения витрин данных, формирования аналитических таблиц и обучения моделей.

Плюсы: гибкость, совместимость со Spark, независимость от зарубежных облаков.

3.4 Слой обработки (Processing)

Выбор: Apache Spark (Databricks)

Обоснование:

— Spark позволяет обрабатывать большие объемы данных быстро;

— поддерживает и batch, и streaming (важно для кликстрима);

— Databricks упрощает работу со Spark: настройка, управление, интеграция с S3/Delta Lake, ML;

Плюсы: мощно, универсально, можно запускать модели машинного обучения.

3.5 Слой аналитики и визуализации (Analytics & Visualization)

Выбор: Metabase

Обоснование:

— подключается к DWH и отображает готовые аналитические панели и дашборды;

— поддерживает интеграцию со Spark и PostgreSQL;

— используется аналитиками и маркетологами для оценки эффективности рекомендаций.

Плюсы: экономично, удобно, быстрый старт.

3.6 Слой оркестрации (Orchestration)

Выбор: Apache Airflow

Обоснование:

— управляет потоками данных и расписанием задач (например, обновление моделей, сбор batch-данных);

— популярен, много документации, легко интегрируется с Kafka, Spark и S3.

Плюсы: надежно, гибко, контроль всего процесса.

3.7 Управление данными (Data Governance)

Выбор: Amundsen

Обоснование:

— позволяет следить за качеством и структурой данных, понимать, откуда данные пришли и кто их использует;

— помогает соблюсти стандарты и ускоряет работу команды.

Плюсы: прозрачность, удобство для команды, ускоряет поиск нужных данных.

Итоговая логика работы:

— Kafka принимает потоковые и batch-данные;

— сырые данные сохраняются в PostgreSQL (Raw Zone);

— после очистки и трансформаций данные попадают в DWH (MinIO + Delta Lake);

— Spark (Databricks) обрабатывает данные и обучает ML-модели → создаются персональные рекомендации;

— Metabase визуализирует результаты в виде дашбордов и аналитических панелей;

— все процессы управляются Airflow, а контроль качества ведётся через Amundsen.

В таблице 3.1 представлен итоговый стек.

Таблица 3.1. Выбранные компоненты архитектуры

Слой	Выбранный инструмент	Краткое обоснование выбора
Сбор данных (Ingestion)	Apache Kafka	Сбор потоковых данных (clickstream) в реальном времени, высокая производительность и масштабируемость.
Первичное хранилище (Raw Storage)	PostgreSQL	Хранение сырых данных, надёжная СУБД, упрощает ETL и интеграцию с DWH.
Хранилище данных (Data Warehouse, DWH)	MinIO + Delta Lake	Хранение очищенных и агрегированных данных, поддержка batch и streaming, независимость от зарубежных облаков.
Обработка и ML (Processing & Machine Learning)	Apache Spark (Databricks)	Очистка, трансформация и анализ данных, обучение ML-моделей для персональных рекомендаций.
Аналитика и визуализация (Analytics & Visualization)	Metabase	Создание дашбордов и аналитических панелей, доступно аналитикам и маркетологам.
Оркестрация (Orchestration)	Apache Airflow	Управление потоками данных, расписаниями и зависимостями задач.
Управление данными (Data Governance)	Amundsen	Каталогизация данных, контроль качества и прозрачности, отслеживание источников.

4. Проектирование архитектуры

Распишем поток данных от источников до пользователей.

Источники данных:

— Кликстрим — события пользователей на сайте и в мобильном приложении;
— Транзакции — данные о заказах и покупках;
— CRM — профили клиентов, история взаимодействий и уровень лояльности;
— Отзывы — тексты и оценки товаров.

Сбор данных (Ingestion)

— Потоковые данные (clickstream) поступают в Apache Kafka, где буферизуются и передаются дальше в систему;
— Пакетные данные (транзакции, CRM, отзывы) загружаются через Apache Airflow по расписанию;
— Все поступающие данные проходят первичную загрузку и сохраняются в PostgreSQL — как в зону сырых данных (Raw Zone).

Хранилище данных (Storage / Data Warehouse)

— Из PostgreSQL данные очищаются, нормализуются и переносятся в MinIO + Delta Lake, которые выполняют роль DWH;
— MinIO обеспечивает надёжное хранение в объектном формате, а Delta Lake управляет версиями, объединяет потоковые и пакетные данные, выполняет очистку и обеспечивает консистентность;
— В этом слое формируются витрины данных для аналитики и обучения моделей.

Обработка данных и машинное обучение (Processing & ML)

Apache Spark (Databricks) выполняет обработку данных из PostgreSQL и DWH:

— агрегирует и объединяет данные из разных источников;
— формирует аналитические таблицы;
— обучает и обновляет ML-модели персональных рекомендаций;
— записывает результаты обратно в Delta Lake (витрины данных) и передаёт результаты в аналитические панели.

Аналитика и визуализация (Analytics & Visualization)
— Metabase подключается к DWH (Delta Lake) и отображает ключевые показатели, отчёты и аналитические дашборды;
— Маркетологи видят эффективность акций и рекомендаций, аналитики — показатели моделей и пользовательское поведение;
— BI-запросы выполняются напрямую к агрегированным данным в Delta Lake.

Безопасность и управление доступом (Security & Access Control)
Данные шифруются при хранении и передаче:
— TLS — для Kafka, MinIO и PostgreSQL;
— встроенные механизмы шифрования MinIO.
Доступ разграничивается по ролям (RBAC) во всех компонентах:
— аналитики — доступ к витринам и отчётам;
— инженеры — к сырым и промежуточным слоям;
— маркетинг — только к дашбордам в Metabase.

Мониторинг и логирование (Monitoring & Logging)
— Prometheus + Grafana — отслеживают состояние Kafka, Spark, Airflow и MinIO (нагрузку, задержки, ресурсы).
— ELK Stack (Elasticsearch, Logstash, Kibana) — собирает логи и ошибки, помогает диагностировать сбои и контролировать стабильность процессов.

Стратегия масштабирования и отказоустойчивости (Scalability & Reliability)

Масштабирование:

— добавление брокеров Kafka,

— увеличение количества узлов Spark,

— горизонтальное расширение MinIO,

— масштабирование PostgreSQL при росте входных данных.

Отказоустойчивость:

— репликация данных в Kafka и MinIO,

— версионность и возможность отката в Delta Lake,

— автоматические перезапуски задач в Airflow и Spark.

5. Создание диаграммы архитектуры

На рисунке 5.1 представлена визуальная схема архитектуры.

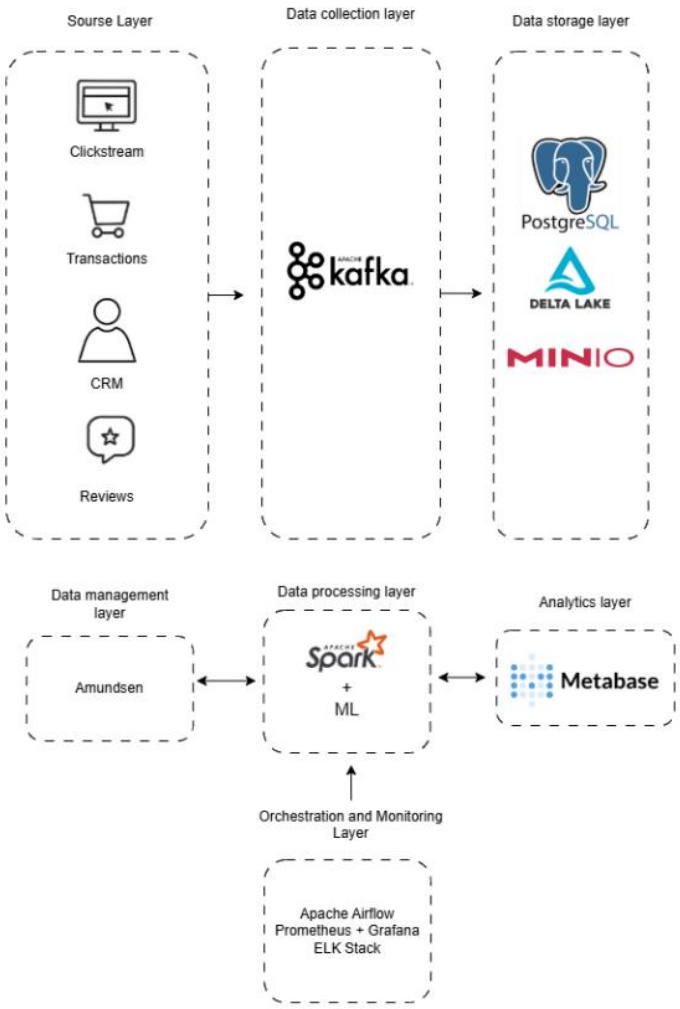

Рисунок 5.1. Диаграмма архитектуры хранилища больших данных онлайн-ритейлера согласно выбранному бизнес-процессу

6. Анализ потенциальных проблем и их решений

В таблице 6.1 представлены потенциальные проблемы и способы их решения.

Таблица 6.1. Возможные проблемы и их решения

Потенциальная проблема	Описание	Возможное решение
1. Сложность управления потоками и зависимостями задач	При большом объёме событий (clickstream) и регулярных загрузках из CRM и транзакций сложно синхронизировать работу Kafka, PostgreSQL, DWH и Spark.	Использовать Apache Airflow для оркестрации ETL-процессов и обновления моделей, а также Prometheus + Grafana для мониторинга состояния потоков и задержек.
2. Рост объёма хранения данных (PostgreSQL и MinIO)	По мере накопления сырых и исторических данных увеличивается нагрузка на PostgreSQL и DWH (MinIO + Delta Lake).	Ввести политику жизненного цикла данных: архивировать старые данные, хранить только актуальные версии в Delta Lake, использовать отдельные бакеты в MinIO для архивов.
3. Обеспечение качества и согласованности данных	Разные источники (CRM, транзакции, отзывы) могут содержать ошибки, дубликаты или несогласованные форматы.	Применять Delta Lake для версионирования и очистки данных, а также Amundsen для контроля происхождения (lineage), описания и качества данных.

Вывод:

Основные риски архитектуры связаны со сложностью управления потоками, ростом объёма хранения и поддержанием качества данных.

Для их решения предусмотрены надёжные инструменты:

— Airflow и Prometheus + Grafana обеспечивают контроль процессов и своевременное оповещение о сбоях;

— Delta Lake и MinIO помогают управлять объёмами и версиями данных;

— Amundsen поддерживает прозрачность и качество данных на уровне всей системы.

Совокупное использование этих решений делает архитектуру устойчивой, управляемой и масштабируемой при работе с большими данными и моделями машинного обучения.