**Документация, содержащая описание функциональных характеристик модуля программного обеспечения**

**Модуль «Сквозная аналитика»**

**г. Москва 2025**

**Содержание**

[1. Предназначение ПО 3](#_s2xpnm9x5zoo)

[2. Описание функциональных характеристик 3](#_7ep7zxhbxr7i)

[3. Архитектура модуля «Сквозная аналитика» 4](#_fuxsqccpbuer)

[4. Схема работы 6](#_spk4gfqhsfnh)

[5. Используемые языки программирования и фреймворки 7](#_pf377y6aljju)

[6. Эксплуатация системы 8](#_s1q4gko3eb1g)

# **Предназначение ПО**

Модуль «Сквозная аналитика» предназначен для комплексного сбора, обработки и визуализации данных о пользовательском поведении на медиаплатформе. Он позволяет анализировать эффективность маркетинговых каналов, отслеживать путь пользователя от первого касания до целевого действия, а также оптимизировать стратегию контентного и продуктового развития.

Модуль интегрируется с «Платформой данных» и модулем «Сегментатор данных», система поддерживает сегментацию аудитории, персонализированную аналитику и автоматизированную отчетность. Основные возможности модуля включают построение карт клиентского пути (CJM), анализ предпочтений пользователей, мониторинг подписок, оценку рентабельности контента и выявление ключевых факторов влияния на конверсию.

Модуль «Сквозная аналитика» помогает медиаплатформам повышать эффективность взаимодействия с аудиторией, улучшать качество контента и оптимизировать маркетинговые расходы, предоставляя инструменты для принятия решений на основе данных.

# **Описание функциональных характеристик**

2.1. Аналитика и визуализация данных

Анализ поведения пользователей: отслеживание действий, построение карт клиентских путей (CJM) и выявление поведенческих паттернов.

Аналитика пользовательских действий с разбивкой по сегментам и когортах.

Создание интерактивных дашбордов для мониторинга ключевых метрик (продажи, конверсии, маркетинговые расходы и др.).

Формирование аналитического профиля пользователя на основе истории его взаимодействий с медиаплатформой.

2.2. Оптимизация и поддержка бизнес-процессов

Оптимизация рекламных каналов и маркетинговых кампаний на основе данных.

Анализ популярности контента для корректировки стратегии закупок и производства собственного контента.

Мониторинг активности подписок для повышения эффективности стратегий удержания и привлечения клиентов.

2.3. Работа с сегментами данных

Разделение пользовательской базы на сегменты с предоставлением аналитики по ним.

Экспорт сформированных сегментов в CSV-файл с идентификаторами пользователей и их атрибутами.

Передача данных во внешние системы через API.

Возможность гибкого формирования новых сегментов с их периодическим обновлением.

2.4. Интерфейсы доступа

Доступ к аналитическим отчетам и инструментам сегментации через удобный интерфейс.

Интуитивно понятный пользовательский интерфейс с четкой навигацией, описаниями и возможностью настройки.

Управление правами доступа для безопасной работы нескольких пользователей одновременно.

2.5. Сбор и интеграция данных

Интеграция информации из различных источников, включая веб-аналитику (Yandex Metrica, AppMetrica), биллинговые системы, медиаплееры и другие сервисы.

Хранение агрегированных данных в масштабируемой базе с учетом высоких требований к производительности.

Реализация ETL-процессов (извлечение, трансформация, загрузка) для обеспечения чистоты и консистентности данных.

# **Архитектура модуля «Сквозная аналитика»**

Модуль «Сквозная аналитика» отвечает за обработку и анализ данных, связанных с маркетинговыми кампаниями, продуктами и контентом. Этот модуль использует несколько источников данных, включая информацию о пользовательских событиях из приложений и веб-сайтов, а также данные о метках и показателях рекламы. Внешние сервисы медиаплатформы, такие как сервисы для авторизации, подписки и сбора пользовательских данных, например, AppMetrica и Яндекс.Метрика предоставляют информацию, которая используется для анализа пользовательских сегментов и создания отчетности.

Платформа данных — это единая среда для хранения и обработки больших объемов информации, которая соединяет модули платформы. Модуль «Сегментатор данных» позволяет выделять пользовательские сегменты для анализа, а модуль «сквозная аналитика» обрабатывает и оптимизирует маркетинговые и продуктовые данные. Вся информация взаимодействует через интерфейсы, предназначенные для аналитиков, маркетологов и менеджеров, которые используют веб-приложение для работы с данными и получения аналитических отчетов.

Технологическая часть платформы включает базы данных, такие как PostgreSQL и ClickHouse, а также различные языки программирования и фреймворки, такие как Python, Java, React, Redis и JavaScript.



*Рис.1 Концептуальная архитектура модуля «Сквозная аналитика»*

# **Схема работы**

4. 1. Сбор данных

Модуль получает данные из различных внешних источников, подключённых к медиаплатформе.

Плеерные логи: информация о просмотрах, перемотках, паузах и досмотрах.

AppMetrica и Яндекс.Метрика: действия пользователей в мобильных и веб-приложениях.

Сервис авторизации: данные о регистрации и входе пользователей.

Сервис подписок: сведения об оплатах, продлениях, отменах подписок.

Маркетинговые инструменты: информация о каналах привлечения, кампаниях, бюджетах и показах рекламы.

4.2. Интеграция данных

Поступающие данные агрегируются и обрабатываются в модуле «Платформа данных», где:

* осуществляется ETL-процесс (извлечение, трансформация, загрузка);
* данные хранятся в масштабируемой базе (ClickHouse, PostgreSQL);
* очищаются, нормализуются и подготавливаются для аналитики.

4.3. Расчёт сегментов пользователей

С помощью модуля «Сегментатор данных», интегрированного с «Платформой данных», производится:

* выделение пользовательских сегментов по заданным критериям (возраст, активность, география и т. д.);
* обновление сегментов в реальном времени;
* экспорт сегментов в CSV и через API.

4.4. Обработка данных в модуле «Сквозная аналитика». На этом этапе модуль

* анализирует поведение пользователей;
* строит карты путей клиентов (Customer Journey Map);
* рассчитывает метрики (MAU, DAU, ROMI, удержание, досмотры и пр.);
* объединяет данные по различным источникам и каналам.

4.5. Визуализация данных

Пользователи получают доступ к данным через настраиваемый веб-интерфейс.

Интерактивные дашборды по продуктовой, маркетинговой и контентной аналитике. Метрики активности, удержания, трафика, подписок. Дашборды на уровне сегментов и отдельных пользователей. Инструменты фильтрации и сравнения сегментов.

# **Используемые языки программирования и фреймворки**

5.1. Java (Backend)

Язык: Java (OpenJDK)

Фреймворки и библиотеки:

Spring Framework / Spring Boot

FlywayDB (миграции БД)

clickhouse-jdbc (интерфейс к ClickHouse)

spring-boot-starter-graphql (GraphQL API)

postgresql (драйвер для PostgreSQL)

hibernate-validator (валидация)

java-telegram-bot-api (интеграция с Telegram)

5.2. JavaScript / TypeScript (Frontend)

Языки: JavaScript, TypeScript

Фреймворки и библиотеки:

React (основа UI)

Redux / @reduxjs/toolkit (управление состоянием)

styled-components (стилизация компонентов)

axios (работа с HTTP-запросами)

@apollo/client (GraphQL клиент)

primereact, rc-tooltip, recharts (визуальные компоненты и графики)

formik (формы), history, gqlmin, graphql

webpack, regenerator-runtime, web-vitals

5.3. Python (обработка данных, ETL, интеграция)

Язык: Python

Библиотеки и фреймворки:

Dagster (оркестрация данных)

PySpark (обработка Big Data)

Polars, Pandas (анализ данных)

deltalake (работа с lake-хранилищами)

PyArrow (работа с колоночными форматами)

5.4. СУБД и хранилища

ClickHouse — аналитическая СУБД (Yandex Managed Service for ClickHouse®)

PostgreSQL — хранение настроек и служебных данных

Redis — кэширование (open source, Docker или Yandex Managed)

Yandex Object Storage — облачное хранилище для данных

5.5. Инфраструктурные и вспомогательные компоненты

Контейнеризация: Docker, Kubernetes (Yandex Managed Service)

ОС в контейнерах: Ubuntu, Debian

Балансировка нагрузки: Yandex Application / Network Load Balancer

Веб-сервер / прокси: Nginx

5.6. Виртуализация и хостинг: Yandex Cloud (SaaS и Compute Cloud)

Авторизация: Yandex ID

# **6. Эксплуатация системы**

6.1 Запуск системы

Запуск модуля «Сквозная аналитика» осуществляется в рамках инфраструктуры Yandex Cloud с использованием контейнерной среды Kubernetes и Docker.

Развёртывание backend-сервисов (Java + Spring Boot), frontend-интерфейса (React), а также ETL-компонентов (Python, Dagster).

Инициализация подключений к базам данных (ClickHouse, PostgreSQL, Redis).

Подключение к источникам данных (AppMetrica, Яндекс.Метрика, логам плеера и т. д.) через API и коннекторы.

Активация пользовательского интерфейса с поддержкой авторизации через Yandex ID.

Проверка доступности сегментатора данных и «Платформы данных».

Для первой настройки предусмотрены шаблоны конфигураций, обеспечивающие автоматическое подключение к основным системам платформы «Предикто».

6.2 Управление

Модуль поддерживает централизованное управление через интерфейс администратора и встроенные инструменты доступа:

* Настройка ролей и прав пользователей (администраторы, аналитики, маркетологи и т. д.).
* Мониторинг состояния систем (веб-интерфейс, подписки, дашборды).
* Управление параметрами сегментации, фильтрами, отображением метрик.
* Отслеживание производительности ETL-процессов и аналитических вычислений.
* Управление подключением к внешним источникам данных через API.

6.3 Резервное копирование и восстановление

Модуль «Сквозная аналитика» поддерживает механизмы резервного копирования всех критически важных компонентов.

ClickHouse — автоматические snapshot'ы и экспорт таблиц на Yandex Object Storage.

PostgreSQL — регулярное бэкапирование настроек и метаданных (Yandex Managed Backup или pg\_dump).

Redis — дампы состояния in-memory кэша по расписанию. Файлы конфигураций и пользовательских дашбордов — хранение в объектном хранилище (Yandex Object Storage).

Восстановление может быть выполнено из любого из предыдущих snapshot'ов или экспортов по команде администратора. Все действия по восстановлению документированы и могут быть автоматизированы через скрипты CI/CD.

6.4 Обновление

Обновление системы осуществляется через систему управления контейнерами (CI/CD pipeline) с минимальным простоем.

Использование версионирования Docker-образов.

Тестирование обновлений в staging-среде.

Обновление производится поэтапно: backend, frontend, ETL-слой, интерфейс сегментации.

При необходимости применяется rolling update для минимизации влияния на пользователей.

Обновление дашбордов и метрик производится с сохранением пользовательских конфигураций.

Все обновления сопровождаются документацией, описанием новых функций и возможностью отката к предыдущей версии при необходимости.