Министерство науки и высшего образования Российской Федерации
Калужский филиал федерального государственного бюджетного образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «ИНФОРМАТИКА И УПРАВЛЕНИЕ»

КАФЕДРА «ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ, ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ НА ТЕМУ:

Проектирование мобильной версии и панели администратора социальной сети с функциями музыкального стримингового сервиса

Студент группы ИУК4-82Б _____ Н.С. Сафронов
(подпись, дата) (И.О. Фамилия)

Руководитель курсовой работы _____ Е.В. Красавин
(подпись, дата) (И.О. Фамилия)

Калуга, 2024

Министерство науки и высшего образования Российской Федерации
Калужский филиал федерального государственного бюджетного образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ
Заведующий кафедрой **ИУК4**
_____(_Ю.Е. Гагарин_)
«_09_» ____февраля____2024г.

З А Д А Н И Е
на выполнение курсовой работы

по дисциплине ***Проектирование программного обеспечения***

Студент группы ***ИУК4-82Б Карпуткин Александр Николаевич***
(фамилия, имя, отчество)

Тема курсовой работы ***Проектирование мобильной версии и панели администратора социальной сети с функциями музыкального стримингового сервиса***

Направленность КР ***учебная***
Источник тематики ***кафедра ИУК4***

Задание

Выполнить проектирование мобильной версии и панели администратора социальной сети с функциями музыкального стримингового сервиса
Разработать функциональную модель приложения;

Оформление курсовой работы

Расчетно-пояснительная записка на_____ листах формата А4.
Перечень графического материала КР (плакаты, схемы, чертежи и т.п.):
− Интерфейс мобильной версии приложения – 1 лист формата А3;
− Функциональная модель приложения – 2 листа формата А3;

Дата выдачи задания «_09_» февраля 2024 г.

Руководитель	09.02.2024	Е.В. Красавин
	(подпись, дата)	(И.О. Фамилия)
Студент	09.02.2024	Н.С. Сафронов
	(подпись, дата)	(И.О. Фамилия)

Министерство науки и высшего образования Российской Федерации
Калужский филиал федерального государственного бюджетного образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

КАЛЕНДАРНЫЙ ПЛАН
на выполнение курсовой работы

по дисциплине ***Проектирование программного обеспечения***

Студент группы ***ИУК4-82Б Сафронов Никита Сергеевич***
(фамилия, имя, отчество)

Тема курсовой работы ***Проектирование мобильной версии и панели администратора социальной сети с функциями музыкального стримингового сервиса***

№	Наименование этапов	Сроки выполнения этапов		Отметка о выполнении	
		план	факт	Руководитель	Куратор
1	Задание на выполнение	1-я нед.	1-я нед.		
2	Выполнение логического проектирования программного обеспечения	4-я нед.	4-я нед.		
3	Выполнение и окончательное оформление графической части и расчетно-пояснительной записки	7-я нед.	7-я нед.		
4	Защита	11-я нед.	11-я нед.		

Студент 09.02.2024г. Руководитель 09.02.2024г.
(подпись, дата) (подпись, дата)

РЕФЕРАТ

Расчетно-пояснительная записка 42 с., 22 рисунка, 17 источников.

Проектирование мобильной версии и панели администратора социальной сети с функциями музыкального стримингового сервиса.

Объектом курсовой работы являются социальные сети и музыкальный стриминг.

Цель работы – проектирование мобильной версии и панели администратора социальной сети с функциями музыкального стримингового сервиса.

Поставленные задачи решаются путем проектирования архитектуры и основных модулей приложения с последующей их реализацией.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Актуальность темы курсового проекта обусловлена тем, что в наше время музыка по праву занимает своё место одних из крупнейших разделов индустрии развлечений. Она стала неотъемлемой частью досуга множества людей: как музыкантов, так и слушателей. Разработка подобного программного продукта, которым смогут пользоваться как создатели, так и потребители музыкального контента, довольно трудоёмкая задача, требующая проработки многих аспектов. Одним из основных функциональных блоков такой социальной сети является музыкальный стриминговый сервис. Пользователи могут слушать свои любимые песни и создавать плейлисты. Кроме того, в социальной сети можно общаться с другими пользователями и делиться музыкой.

Объектом курсового проекта являются социальные сети и музыкальный стриминг.

Предметом исследования курсового проекта является реализация мобильного приложения социальной сети с функциями стримингового сервиса.

Целью проекта является разработка мобильного приложения социальной сети с функциями музыкального стриминга.

Для достижения поставленной цели решаются следующие задачи:

1. Выполнить анализ предметной области;
2. Провести сравнительный анализ существующих аналогов;
3. Определить оптимальную структуру системы;
4. Осуществить выбор средств реализации программного продукта, соответствующего выбранной структуре;
5. Реализовать базу данных и программные компоненты системы;
6. Осуществить тестирование компонентов;
7. Разработать сопроводительную документацию.

1. АНАЛИЗ ТРЕБОВАНИЙ И ТЕХНОЛОГИЙ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

1.1. Основные требования к разрабатываемой системе

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

- разрабатываемое приложение должно иметь мобильную и браузерную версии;
- разрабатываемое приложение должно иметь удобный интерфейс для взаимодействия с пользователем, быть клиент-ориентированным и интуитивно понятным в использовании;
- пользователь должен иметь возможность регистрации и авторизации в приложении;
- при вводе некорректных личных данных пользователь должен видеть сообщение об ошибке;
- пользователь должен иметь возможность просматривать и редактировать свои профиль после регистрации и авторизации;
- пользователь может прослушивать доступные для него плейлисты и управлять процессом воспроизведения с помощью интерфейса музыкального плеера;
- пользователь должен иметь возможность создания плейлистов, их редактирования и распространения среди других пользователей;
- исполнитель может выкладывать и распространять на платформе свои композиции;
- правообладатель обладает возможностью пожаловаться на нарушение его авторских прав, которые должен рассматривать администратор платформы;

- исполнитель должен иметь возможность объединять выложенные композиции в альбомы;
- пользователь должны быть доступны типовые функции социальных сетей: поставить отметку «Нравится», подписаться на автора, оставлять комментарии;
- приложение должно обладать рекомендательной системой на основе прослушанных пользователем композиций и тегов, установленных автором;
- пользователю должна быть доступной лента – последние выложенные композиции от отслеживаемых авторов и рекомендуемые для него новые исполнители в виде списка.

Система требует наличия пользователей двух категорий – рядовых пользователей сервиса, которые также могут являться и исполнителями, и администраторов, отвечающих за работу с жалобами пользователей и правообладателей на неправомерно выложенный контент.

Защита информации осуществляется разграничением прав доступа – обычные пользователи не должны иметь возможность вносить изменения, которые могут повлиять на функционирование системы. Также разрабатываемое приложение должно гарантировать пользователю защиту его данных. При указании соответствующих режимов приватности исполнителю также должна быть гарантирована защита объектов его творчества.

1.2. Анализ аналогов и прототипов

В настоящее время существуют аналоги, предоставляющие подобный функционал. Среди аналогов можно выделить следующие программные решения: SoundCloud, Spotify, Bandcamp.

SoundCloud

SoundCloud является уникальным стриминговым сервисом, который предлагает не только множество музыкальных треков, но также функции

социальных сетей. В отличие от других платформ, SoundCloud позволяет не только слушать музыку, но и делиться своими треками с другими пользователями. Это создает живое сообщество музыкантов, диджеев и слушателей, обеспечивая уникальный опыт.

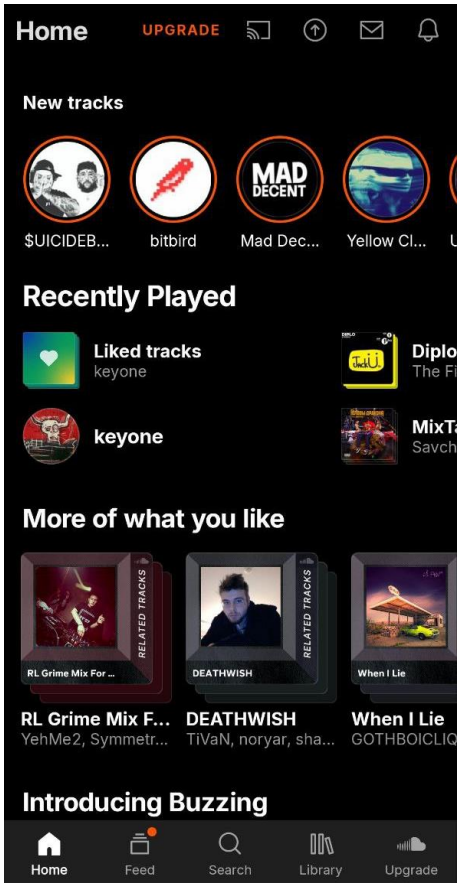

Рисунок 1.1. Интерфейс мобильного приложения «SoundCloud»

На SoundCloud пользователи могут стать активными участниками, оставлять комментарии, ставить лайки и репосты, а также подписываться на профили других пользователей. Это позволяет взаимодействовать с артистами, находить новые музыкальные таланты и строить свою аудиторию. Кроме того, SoundCloud предоставляет возможность создавать и прослушивать плейлисты.

В контексте анализа аналоговых стриминговых сервисов с функциями социальных сетей, SoundCloud является уникальным игроком, который акцентирует внимание на связи и взаимодействии между музыкантами и

слушателями. Он предлагает демократичную платформу для продвижения музыкальных талантов и способствует расширению музыкальной культуры. В целом, SoundCloud предоставляет возможность глубокого погружения в мир музыки, где пользователи сами становятся создателями и потребителями, а также находят нового вдохновения и источники музыкального творчества. В данный момент SoundCloud заблокирован на территории Российской Федерации.

Spotify

Spotify – это один из ведущих стриминговых сервисов мирового уровня, предлагающий широкий доступ к миллионам треков со всего мира.

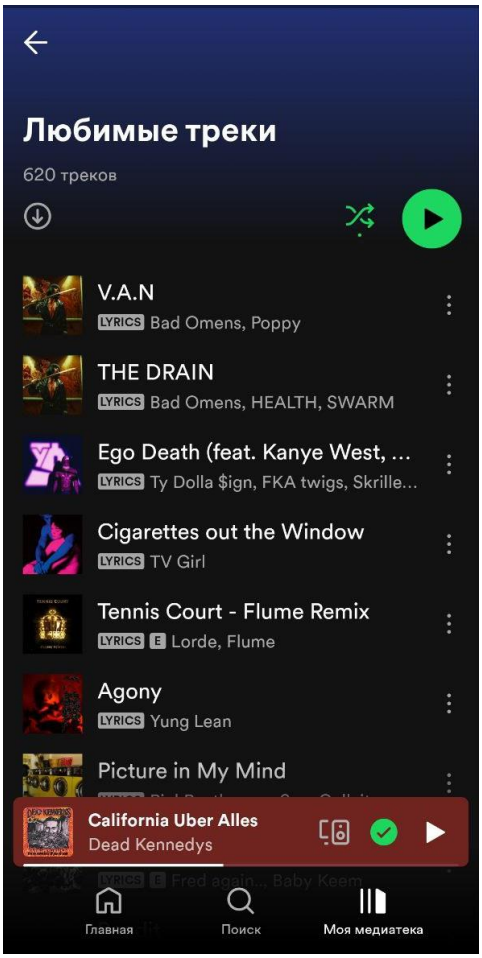

Рисунок 1.2. Интерфейс мобильного приложения «Spotify»

На Spotify пользователи имеют возможность создавать собственные профили, настраивать музыкальные предпочтения и выбирать плейлисты,

основанные на их вкусах. Пользователи могут следить за плейлистами других пользователей, делиться своими открытиями и даже совместно создавать плейлисты с друзьями. Это создает музыкальное сообщество, которое помогает пользователям находить новую музыку, обмениваться рекомендациями и поддерживать связи в мире музыки.

Несмотря на уход с российского рынка, Spotify остается значимым и востребованным стриминговым сервисом в мировом масштабе, предоставляя пользователям удобство и разнообразие в музыкальном контенте.

Bandcamp

Bandcamp — это уникальная платформа для музыкантов и независимых лейблов. Одной из ключевых особенностей Bandcamp является его поддержка независимых артистов, которые могут продавать свою музыку, мерч и билеты на концерты непосредственно своим поклонникам.

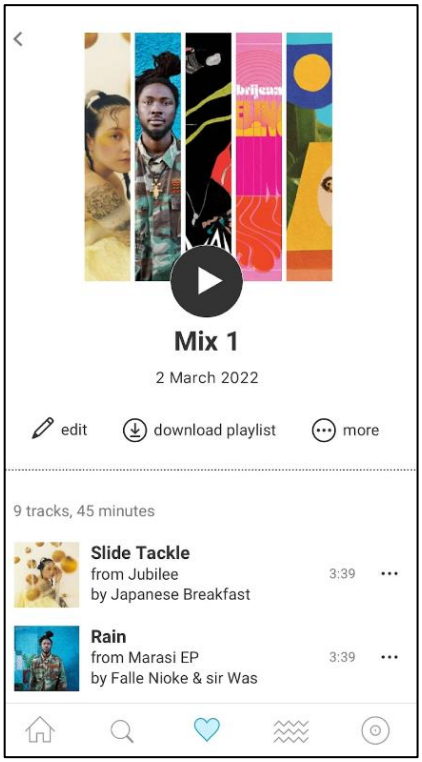

Рисунок 1.3. Интерфейс мобильного приложения «Bandcamp»

Пользователи Bandcamp имеют возможность открыть собственные профили и подписываться на музыкантов, чья музыка их интересует. Это

позволяет пользователям получать уведомления о новых релизах и мероприятиях, а также обмениваться комментариями, рецензиями и рекомендациями с другими фанатами и музыкантами. Функция социальных сетей Bandcamp стимулирует активное взаимодействие в музыкальном сообществе, способствуя обнаружению новых талантов и созданию тесной связи между музыкантами и их поклонниками.

Однако, Bandcamp не предоставляет полноценный стриминговый сервис, а вместо этого сосредотачивается на продаже и поддержке музыки независимых артистов. К сожалению, Bandcamp также не доступен на российском рынке.

Все рассмотренные аналоги имеют как браузерную версию приложения, так и мобильную. После анализа вышеперечисленных сервисов, было решено создать схожий сервис с учетом преимуществ и недостатков рассмотренных программных продуктов.

1.3. Перечень задач, подлежащих решению в процессе разработки

Для успешного выполнения поставленной задачи требуется определить оптимальную структуру приложения, выбрать соответствующий технологический стек, разработать дизайн приложения, учитывая все необходимые модули и потоки данных. Необходимо создать функции для эффективного взаимодействия между клиентским и серверным слоями приложения, а также подготовить проектную документацию для последующей разработки.

Для этого потребуется использовать СУБД для хранения данных, языки программирования клиентской и серверной части вместе с соответствующими фреймворками для работы с сервером и клиентом, специализированные библиотеки для разработки обеих частей системы, брокер сообщений для обмена данными между модулями, а также необходимое программное обеспечение для создания функциональной модели приложения.

1.4. Обоснование выбора инструментов для программирования мобильной части приложения

JavaScript

JavaScript — это полноценный динамический язык программирования, который применяется к HTML документу, и может обеспечить динамическую интерактивность на веб-сайтах.

Он обладает широким спектром возможностей и полезных библиотек, включая fetch, React и React Native, делая его привлекательным выбором для разработки веб- и мобильных приложений. Кроме того, JavaScript используется для создания мобильных приложений, расширяя его применение.

JavaScript является динамическим и слабо типизированным языком, что означает, что он динамически определяет типы данных во время выполнения программы и позволяет проводить операции над разными типами данных без явного преобразования. Это облегчает разработку и ускоряет процесс создания приложений.

Благодаря активному и большому сообществу разработчиков, JavaScript имеет огромное количество открытых библиотек и фреймворков, облегчающих создание веб-приложений. Это делает его одним из наиболее популярных и востребованных языков программирования в мире веб-разработки.

В целом, выбор JavaScript в качестве языка разработки обусловлен его универсальностью, активным сообществом разработчиков, большим количеством открытых библиотек и фреймворков, скоростью разработки и возможностью взаимодействия с HTML и CSS.

React Native

Для эффективной разработки мобильных приложений на основе React Native используется React.

React Native — это фреймворк с открытым исходным кодом для разработки мобильных приложений на базе JavaScript. Он позволяет создавать нативные приложения для платформ iOS и Android, используя ту же технологию, что и для веб-разработки на React. Главной целью React Native является обеспечение высокой производительности и переносимости кода между разными платформами.

Выбор React Native обусловлен его эффективностью, гибкостью и простотой использования. Фреймворк пользуется огромной популярностью благодаря активному сообществу разработчиков и наличию множества плагинов и библиотек, улучшающих его функциональность и облегчающих разработку.

Одним из примеров таких библиотек является Material Design, которая предоставляет готовые компоненты и стили, позволяя быстро создавать привлекательные пользовательские интерфейсы согласно принципам Material Design.

Важно отметить, что помимо правильной архитектуры, внимание к дизайну играет ключевую роль в создании мобильных приложений. Выбор React Native совместно с Material Design обеспечивает не только техническую часть приложения, но и качественное представление его элементов, что способствует приятному пользовательскому опыту.

Redux

Для эффективного управления состоянием приложений на React Native используется Redux.

Redux — это библиотека с открытым исходным кодом для управления состоянием приложений JavaScript. Она предоставляет предсказуемое хранилище данных для вашего приложения, что делает управление состоянием проще и более эффективным. Redux позволяет централизованно хранить данные приложения и управлять ими через четко определенные действия и редукторы.

Выбор Redux для React Native обусловлен его способностью обеспечить структурированное и масштабируемое управление состоянием приложения. Он позволяет разработчикам легко отслеживать и обновлять данные приложения, а также упрощает процесс отладки.

Кроме того, Redux имеет широкую поддержку в сообществе разработчиков и множество плагинов и инструментов для расширения его функциональности. Это делает его популярным выбором для управления состоянием в React Native приложениях.

Одним из примеров интеграции Redux с React Native является использование библиотеки react-redux, которая обеспечивает интеграцию Redux с React Native компонентами и упрощает процесс связывания данных из хранилища Redux с интерфейсом приложения.

1.5. Обоснование выбора инструментов для программирования серверной части приложения

Python

Для разработки серверной части приложения в совокупности с фреймворком Flask используется Python.

Python — это высокоуровневый интерпретируемый язык программирования, который обеспечивает простоту и эффективность в создании веб-приложений.

Выбор Python для программирования серверной части приложения обоснован его многофункциональностью, гибкостью и широкой поддержкой в сообществе разработчиков.

Кроме того, Python имеет обширное сообщество разработчиков и большое количество ресурсов для обучения и поддержки, что обеспечивает стабильность и надежность разработки на этом языке.

aiohttp

aiohttp — асинхронный HTTP-сервер для модуля asyncio. Это библиотека языка Python, которая нужна для выполнения клиентских запросов и создания веб-сервера с потоковой выдачей и веб-сокетами. Асинхронность нужна для выполнения нескольких операций, установления соединений, одновременно, без ожидания завершения предыдущих. Использование такого подхода увеличивает скорость работы веб-сервисов в несколько раз, причина низкой производительности которых, обычно, не проведение сложных вычислений, а именно ожидание ввода/вывода.

Flask

Совместное использование Python с фреймворком Flask обеспечивает еще большую гибкость и удобство при разработке серверной части. Flask — это легковесный и расширяемый фреймворк, который позволяет быстро создавать веб-приложения. Он предоставляет минимальный набор инструментов, что делает его идеальным выбором для маленьких и средних проектов, а также для создания прототипов и MVP (минимально жизнеспособных продуктов).

uWSGI

uWSGI — это быстрый скомпилированный серверный пакет с обширной конфигурацией и возможностями, выходящим за рамки базового сервера. Он очень производителен, поскольку является компилируемым. Широкий спектр функциональных возможностей в сочетании с относительной легкостью настройки делают uWSGI отличным вариантом для развертывания многих приложений, особенно в сочетании с Nginx.

SQLAlchemy

SQLAlchemy — это гибкий инструмент для работы с базами данных в языке программирования Python. SQLAlchemy предоставляет разработчикам высокоуровневый API для работы с базами данных, что позволяет создавать и выполнять сложные запросы к данным, управлять транзакциями и

моделировать структуру базы данных с помощью объектно-реляционного отображения (ORM).

Выбор SQLAlchemy обоснован его гибкостью, мощными возможностями и интеграцией с различными СУБД, включая PostgreSQL, MySQL, SQLite и многие другие. SQLAlchemy позволяет разработчикам писать код, который абстрагирует от конкретной СУБД, что облегчает переносимость приложений между разными базами данных.

Кроме того, SQLAlchemy предоставляет разработчикам инструменты для создания и управления миграциями баз данных, автоматического создания схемы базы данных на основе моделей Python и выполнения запросов на языке SQL в объектно-ориентированном стиле. Это делает SQLAlchemy мощным инструментом для разработки приложений, требующих сложной работы с данными.

SQLAlchemy также имеет обширную документацию и активное сообщество разработчиков, что обеспечивает поддержку и ресурсы для разработчиков, использующих этот инструмент. Эти факторы делают SQLAlchemy популярным и широко используемым инструментом для работы с базами данных в языке Python.

Обоснование выбора СУБД

PostgreSQL — это объектно-реляционная система управления базами данных с открытым исходным кодом. PostgreSQL широко используется для хранения и управления данными в различных типах приложений, включая веб-приложения, аналитические системы, геопространственные приложения и многое другое. Его мощные возможности и надежность делают его популярным выбором для проектов различных масштабов.

Выбор PostgreSQL обоснован его масштабируемостью, производительностью, расширяемостью и надежностью. Он поддерживает множество продвинутых функций, включая сложные запросы, транзакции, внешние ключи, ограничения целостности данных и многое другое. Это делает

PostgreSQL идеальным выбором для разработчиков, которым требуется надежное хранилище данных с расширенными возможностями.

Кроме того, PostgreSQL обладает обширным экосистемой инструментов и расширений, включая инструменты для мониторинга, резервного копирования, масштабирования и репликации данных. Это обеспечивает разработчикам широкие возможности для настройки и оптимизации их баз данных в соответствии с конкретными требованиями проекта.

PostgreSQL также активно развивается сообществом разработчиков и имеет обширную документацию, что обеспечивает поддержку и ресурсы для разработчиков, использующих эту СУБД. Эти факторы делают PostgreSQL одним из наиболее популярных и востребованных выборов для управления данными в различных приложениях.

Обоснование выбора брокера сообщений

Apache Kafka – это распределенная платформа для обработки и передачи потоков данных в реальном времени.

Apache Kafka широко используется для обработки и управления данными в реальном времени, таких как журналы действий, потоковые данные из сенсоров, данные мониторинга и многое другое. Его архитектура позволяет обрабатывать и передавать огромные объемы данных с высокой пропускной способностью и надежностью.

Выбор Apache Kafka обоснован его масштабируемостью, устойчивостью и эффективностью. Он позволяет обрабатывать огромные потоки данных и обеспечивает гарантированную доставку сообщений даже в условиях высоких нагрузок и сбоев системы.

Кроме того, Apache Kafka имеет богатое экосистема инструментов и библиотек, которые облегчают разработку и интеграцию с другими системами. Это включает в себя инструменты для мониторинга, управления и обработки данных, такие как Kafka Connect, Kafka Streams и многие другие.

Apache Kafka также имеет активное сообщество разработчиков и обширную документацию, что обеспечивает поддержку и ресурсы для

разработчиков, использующих эту технологию. Эти факторы делают Apache Kafka популярным и востребованным инструментом для обработки потоков данных в реальном времени.

Форма взаимодействия между клиентом и сервером, компонентами системы

В качестве взаимодействия сервера и клиент был была выбрана архитектура REST, так что сервер будет представлять собой REST API, обращающийся к базе данных за информацией и отправляющий ее клиенту по HTTP протоколу. Система считается спроектированной по REST, если:

1. Система имеет явное разделение на клиент и сервер;
2. Сервер хранит какой-либо информации о клиентах. В запросе должна храниться вся необходимая информация для обработки запроса;
3. Используется кеширование данных;
4. Используется единый интерфейс между клиентом и сервером;
5. Система разделена на несколько малосвязанных между собой слоев.

Там, где необходимо непрерывное обновление данных для пользователей (например, лента последних постов), используется библиотека Flask-SocketIO для микрофреймворка Flask, работающая на основе WebSockets и позволяющая в режиме реального времени оповещать пользователей об изменениях данных на сервере.

Приложение будет разделено на микросервисы, взаимодействие которых будет организовываться внутри общей сети, разворачиваемой с помощью docker-compose. Точкой входа извне будет являться веб-сервер Nginx, который также будет находиться в сети docker-compose и будет проксировать запросы непосредственно к модулям приложения.

1.6. Обоснование выбора инструментов и платформы для разработки информационных моделей приложения

ERwin Process Modeler – это инструмент для моделирования и управления бизнес-процессами.

ERwin Process Modeler предоставляет возможности для анализа, документирования и оптимизации бизнес-процессов организации. С его помощью пользователи могут создавать графические модели бизнес-процессов, описывать шаги процессов, определять взаимосвязи между ними, а также анализировать их эффективность и эффективность.

Выбор ERwin Process Modeler обоснован его интуитивным интерфейсом, богатым функционалом и возможностью визуализации бизнес-процессов. Он предоставляет удобные инструменты для работы с процессами, что делает процесс моделирования более эффективным и понятным.

Кроме того, ERwin Process Modeler интегрируется с другими инструментами управления бизнес-процессами и системами управления проектами, что обеспечивает согласованность и эффективность работы организации.

Обширная документация и поддержка сообщества пользователей делают ERwin Process Modeler популярным выбором для разработки информационных моделей бизнес-процессов.

Выводы

Таким образом, исходя из требований к заявленному приложению был проведен анализ нескольких инструментов для разработки, что послужило к формированию стека следующих технологий:

- для разработки информационной модели приложения было выбрано программное обеспечение ERWin Process Modeler, позволяющего проводить анализ, документирование, а также помогающего отследить информационные потоки на всех уровнях работы конечного продукта;

- для разработки мобильного приложения были выбраны библиотеки на базе языка программирования высокого уровня — JavaScript: React Native, Redux. Заявленные инструменты обеспечат быстрый и удобный интерфейс для пользователей приложения;
- для разработки серверной части приложения были выбраны следующие инструменты: язык программирования Python, СУБД PostgreSQL и SQLAlchemy для переноса моделей на код, Flask и aiohttp для разработки серверной части (в зависимости от требований к скорости работы модуля), uWSGI – в качестве самого веб-сервера, Apache Kafka – для организации межмодульного взаимодействия;
- перед клиентской и серверной частью будет стоять веб-сервер Nginx, а всё приложение будет контейнеризировано и запущено с использованием Docker.

2. ПРОЕКТИРОВАНИЕ КОМПОНЕНТОВ ПРОГРАММНОГО ПРОДУКТА

2.1 Проектирование функциональной модели

Для выстраивания грамотного процесса разработки приложения необходимо спроектировать его информационную модель, отражающие основные бизнес-процессы и потоки входных и выходных данных. Информационная модель IDEF0 «Работа социальной сети с функциями музыкального стримингового сервиса» представлена на рисунке 2.1:

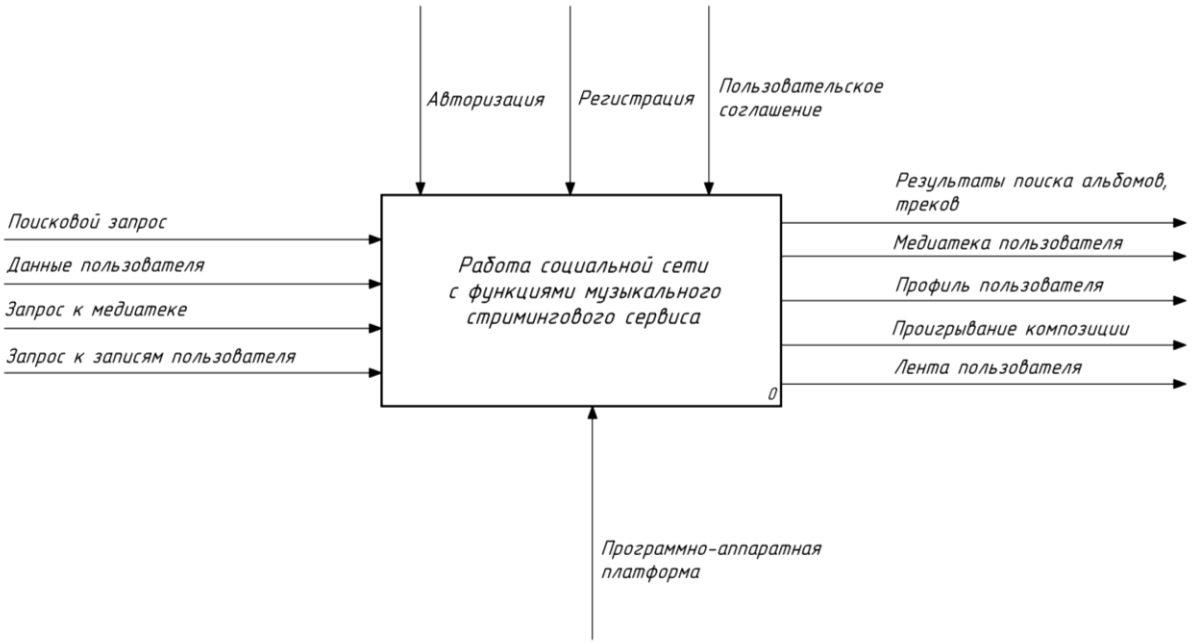

Рисунок 2.1. Диаграмма модели IDEF0 «Работа социальной сети с функциями музыкального стримингового сервиса»

Основными функциональными возможностями приложения является поиск альбомов, треков и профилей пользователей; просмотр и редактирование медиатеки пользователя; просмотр и редактирование профиля пользователя; проигрывание треков и управление очередью их воспроизведения; просмотр ленты пользователя, добавление реакций и собственных записей. В качестве входных данных используются данные пользователя, представляющие собой информацию, необходимую для входа в

приложение, такую как хэши сессий и токены авторизации; запросы для осуществления поиска по медиатеке сервиса, работы с медиатекой и управления записями пользователя. Помимо этого, механизмами управления являются процедуры регистрации и авторизации, а реализуются перечисленные функции средствами программно-аппаратной платформы. Для отслеживания отдельных этапов работы приложения была произведена декомпозиция модели. Декомпозиция контекстной диаграммы представлена на рисунке 2.2:

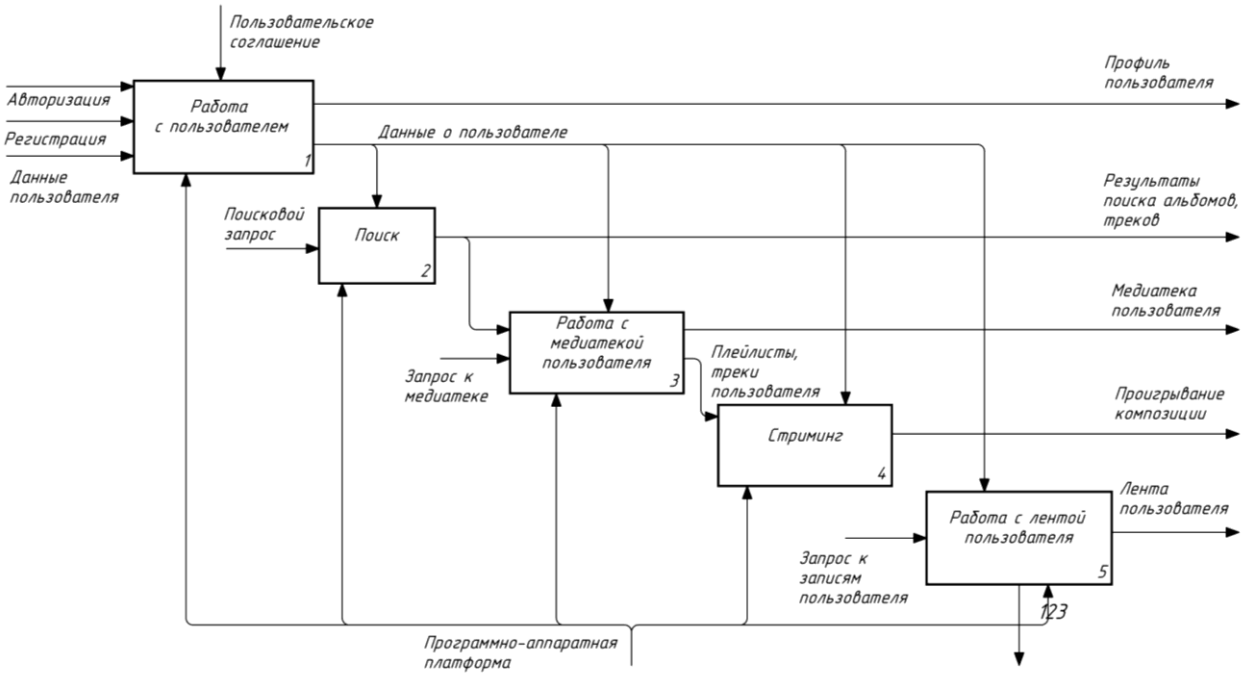

Рисунок 2.2. Декомпозиция модели «Работа социальной сети с функциями музыкального стримингового сервиса»

Для входа в приложение данные, необходимые для аутентификации пользователя, изначально попадают в механизм работы с пользователями. На основе этих данных и запроса пользователя выполняется либо авторизация, либо регистрация пользователя, после чего формируется сессия пользователя и в приложении сохраняются данные о соответствующем ей пользователе. После данного действия пользователь получает доступ к разрабатываемой системе и способен работать со следующими сценариями.

Основными функциями блока «Поиск» являются поиск треков, альбомов, плейлистов и профилей пользователей.

Блок «Работа с медиатекой» отвечает за просмотр и изменение пользовательской медиатеки: редактирование пользовательских плейлистов, добавление новых треков, альбомов и плейлистов пользователя, а также редактирование существующих; добавление треков в плейлист «Любимые треки» после отметки «Нравится».

Блок «Стриминг» отвечает за воспроизведение и управление воспроизведением: добавление треков в очередь воспроизведения, установку трека в состояния «Пауза» и «Воспроизведение», прослушивание плейлистов, альбомов, исполнителей.

Блок «Работа с лентой пользователя» отвечает за создание новых записей, управление существующими записями, просмотр записей пользователя, а также установку реакций под ними.

На рисунке 2.3 представлена декомпозиция модуля «Работа с пользователем»:

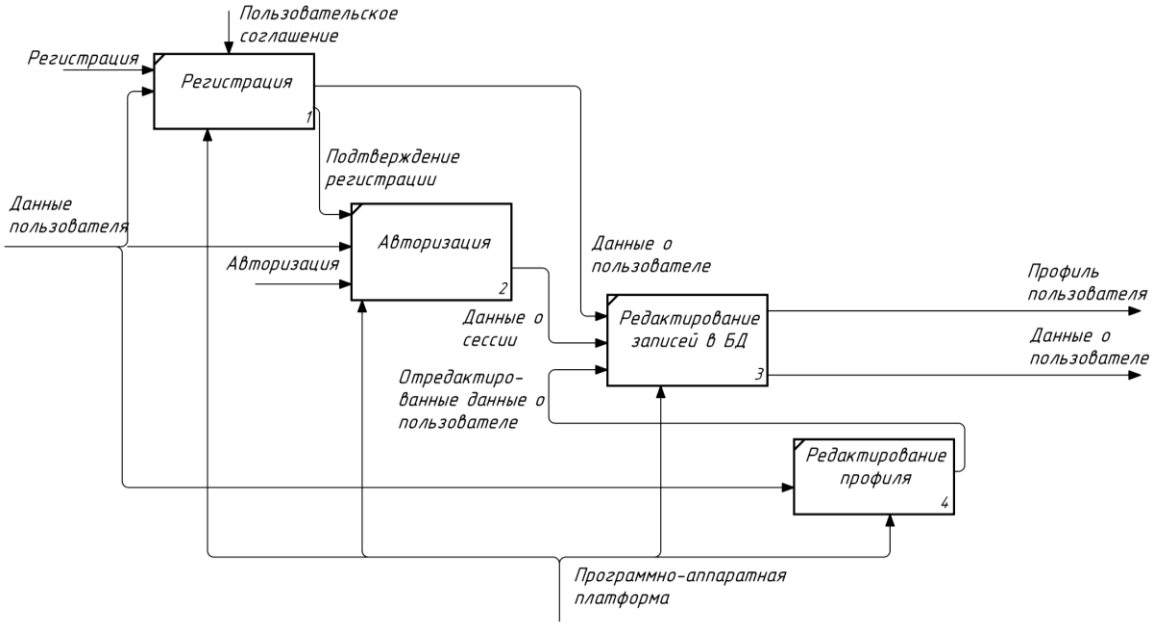

Рисунок 2.3. Декомпозиция блока «Работа с пользователем»

Декомпозиция данного модуля отражает то, что перед использованием приложения пользователю необходимо пройти авторизацию, а при отсутствии

учётной записи – регистрацию. После выполнения данных действий пользователь получит сессию, а приложение сохраняет данные о пользователе для выполнения проверок доступа и вывода требуемых пользователем данных.

Декомпозиция блока «Поиск», представленная на рисунке 2.4, отражает то, что для получения результатов поиска необходимо валидировать входные данные, а также проверить наличие сессии пользователя и его права доступа для того, чтобы в дальнейшем отправить сформированный на основе этих данных запрос к БД, результат которого в дальнейшем форматируется в соответствии с правилами.

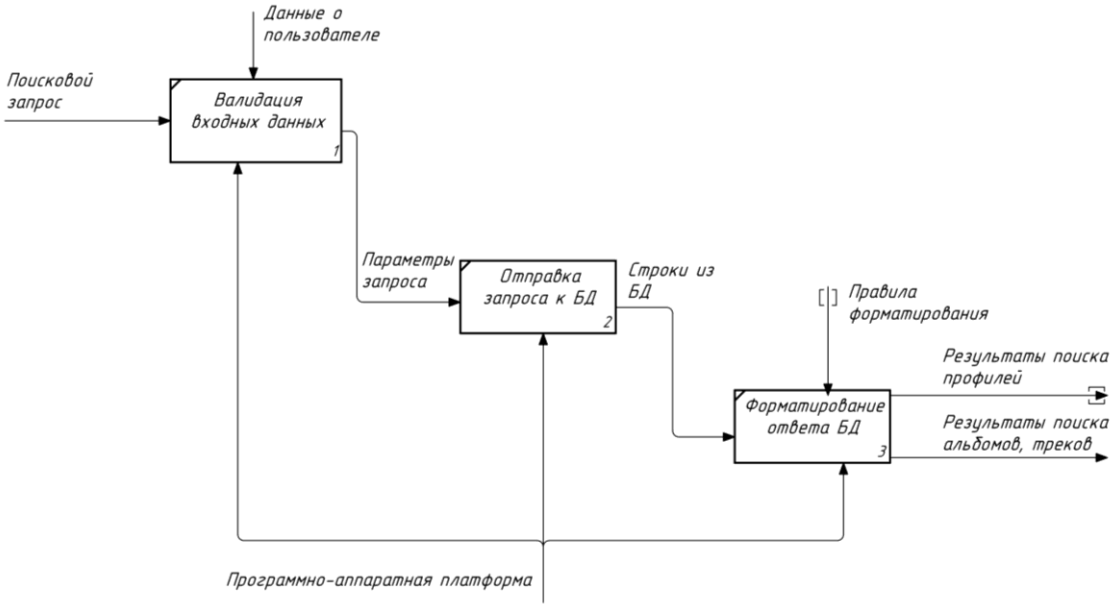

Рисунок 2.4. Декомпозиция блока «Поиск»

Работа с медиатекой представляется следующими действиями: созданием трека, на основе которых пользователь может создать альбом или плейлист; добавлением треков и альбомов, созданных пользователем, либо найденных их с использованием поиска в медиатеку; созданием записей в БД на основе данных действий (добавлением в служащие для связей «многие-ко-многим» или «один-ко-многим» промежуточные таблицы альбомов или плейлистов соответствующих записей, а также созданием и редактированием записей основных таблиц); по результатам всех действий происходит

формирование записей медиатеки пользователя, включающей плейлисты и альбомы.

Декомпозиция блока «Работа с медиатекой пользователя» представлена на рисунке 2.5.

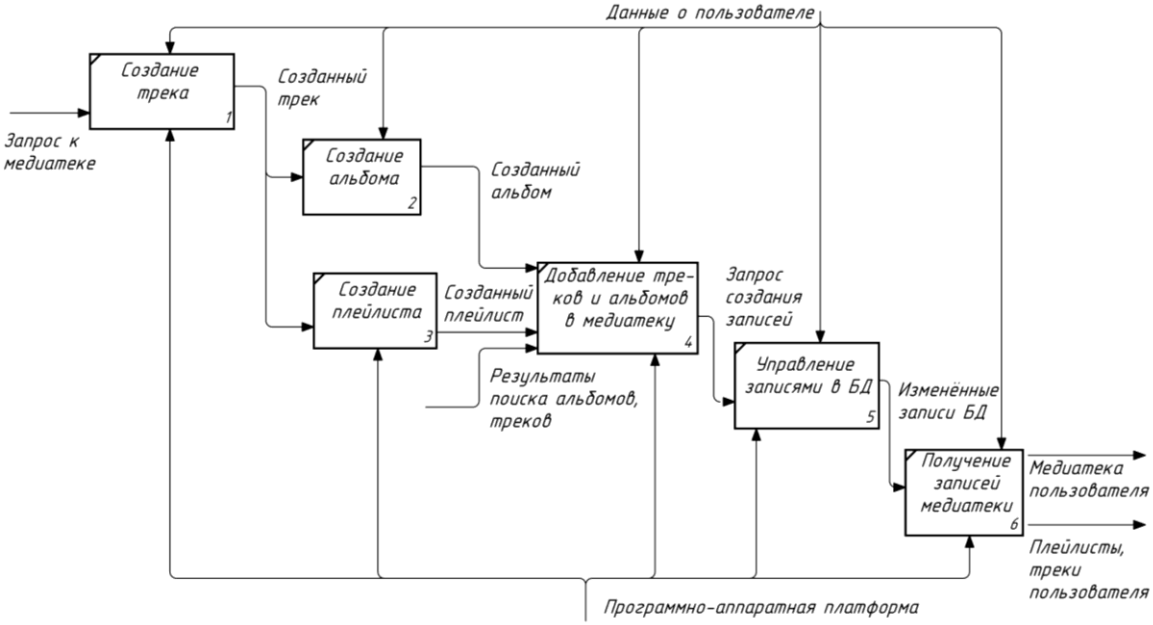

Рисунок 2.5. Декомпозиция блока «Работа с медиатекой пользователя»

Была выполнена декомпозиция блока «Стриминг», представленная на рисунке 2.6.

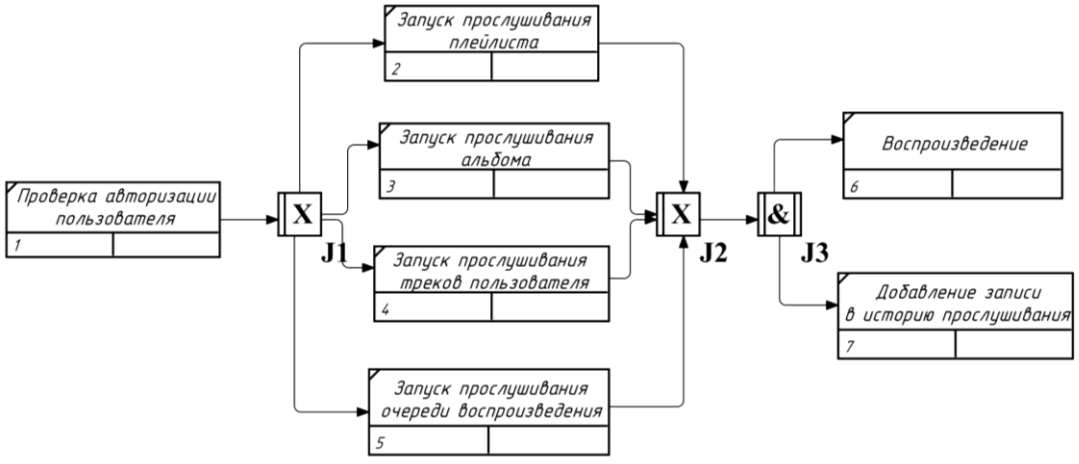

Рисунок 2.6. Декомпозиция блока «Стриминг»

Стриминг разделён на несколько этапов, начиная с проверки авторизации пользователя с последующим запуском проигрывания треков из

различных источников, таких как плейлист, альбом, треки пользователя, а также очередь воспроизведения, и заканчивая началом воспроизведения композиции вместе с добавлением соответствующей записи в плейлисте «История прослушивания».

Выполнение сценария «Работа с лентой пользователя» начинается с проверки авторизации пользователя и доступность с точки зрения прав доступности профиля пользователя. После выполнения проверок при успехе пользователь получает доступ к профилю, где он может просматривать записи, добавлять реакции на записи, а также изменять статус подписки на пользователя.

На рисунке 2.7 отражена декомпозиция модуля «Работа с лентой пользователя»:

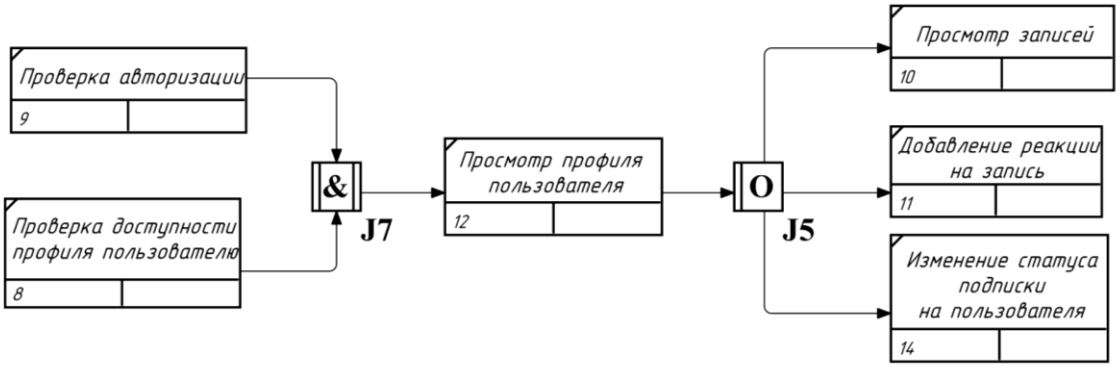

Рисунок 2.7. Декомпозиция блока «Работа с лентой пользователя»

В ходе проектирования сервиса была разработана функциональная модель приложения с использованием методологий IDEF0 и IDEF3. Разработанная модель позволяет рассмотреть основные бизнес-процессы сервиса, а также грамотно разработать план по разработке конечной системы, сверяясь с получившейся схемой на каждом этапе создания приложения. Вместе с тем модель может видоизменяться при добавлении новых модулей или изменении существующих, что обеспечивает гибкость при разработке системы.

3. ТЕСТИРОВАНИЕ И ИНТЕГРАЦИЯ КОМПОНЕНТОВ ПРОГРАММНОГО ПРОДУКТА

3.1. Назначение приложения

Приложение представляет собой мобильную версию социальной сети с функциями музыкального стримингового сервиса для прослушивания музыки, управления медиатекой, поиском новых композиций, исполнителей, альбомов и плейлистов, просмотром записей авторов, на которых пользователь подписан.

3.2. Руководство пользователя

Пользователю необходимо запустить мобильное приложение. В результате появится главное окно (см. рис. 3.1), в шапке снизу которого находятся основные кнопки меню управления.

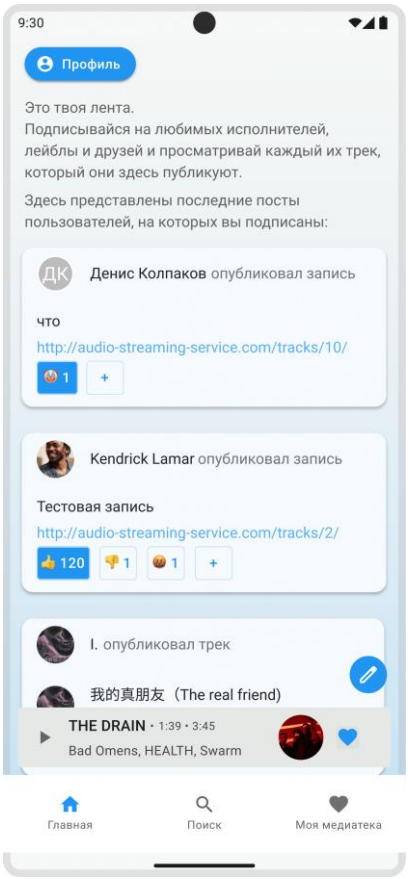

Рисунок 3.1. Главная страница приложения

На главной странице приложения пользователь может просмотреть записи тех пользователей, на которых он подписан, а также узнавать о выходах новых треков, альбомов, обновлениях плейлистов. На опубликованные записи пользователь может ставить свои реакции, либо переходить на прикреплённые элементы и в профиль пользователя. Помимо этого, пользователь может создать новую запись нажатием кнопки с иконкой карандаша – ему будет предложена форма

В нижней панели находятся кнопки навигации для перехода между основными частями приложения: на главную страницу, на страницу поиска и в медиатеку.

Страница «Поиск» представлена на рисунке 3.2.

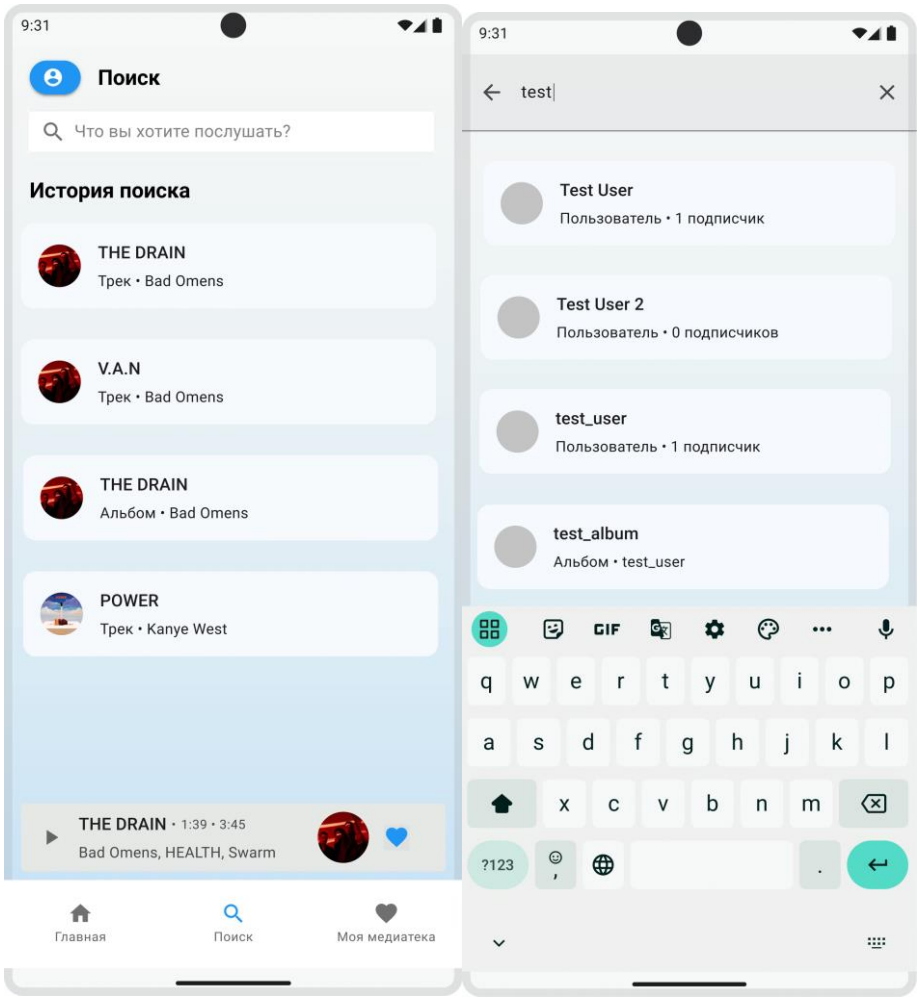

Рисунок 3.2. Страница «Поиск»

На странице «Поиск» пользователь может просмотреть свою историю найденных треков, альбомов, плейлистов или исполнителей, либо создать новый поисковой запрос для получения всех соответствующих ему записей.

Страница «Моя медиатека» представлена на рисунке 3.3.

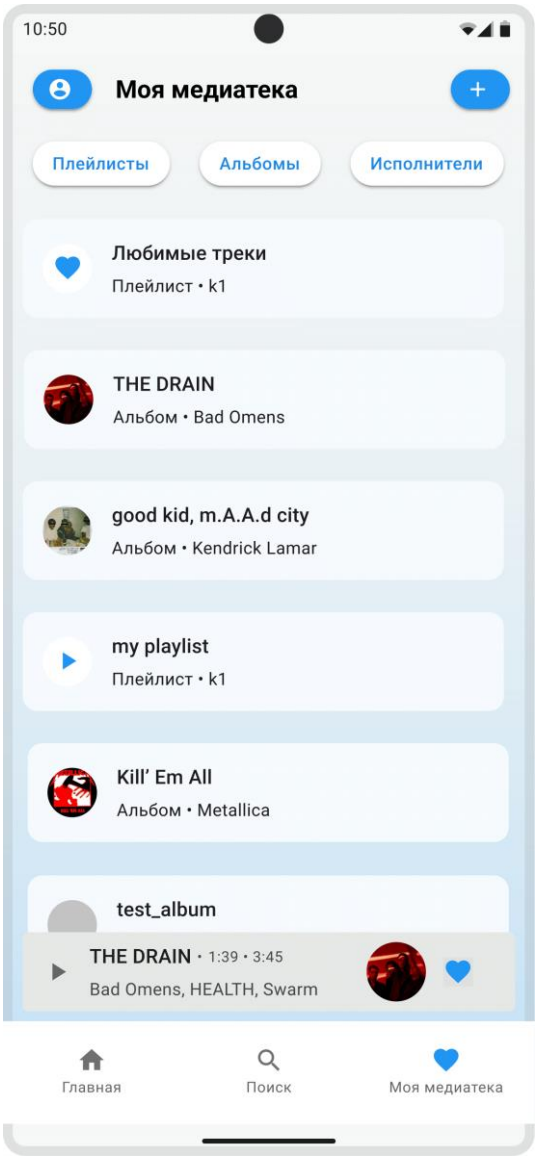

Рисунок 3.3. Страница «Моя медиатека»

Данная страница позволяет просматривать сохранённые в медиатеку пользователя элементы: те, которым он поставил отметку нравится, и те, которые он сам создал.

По нажатии кнопки с иконкой плюса пользователь может добавить новую запись (см. рис. 3.4).

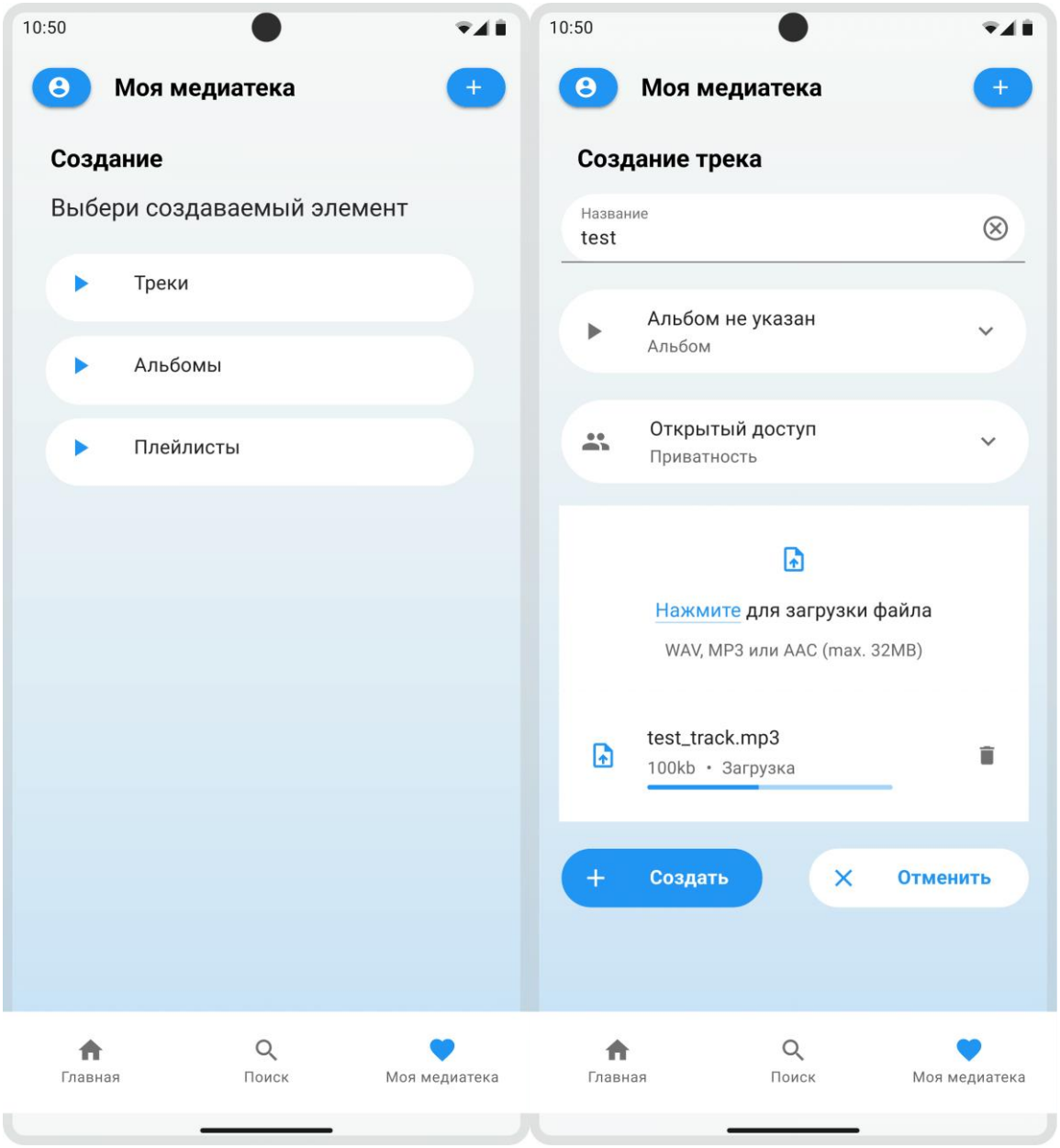

Рисунок 3.4. Страница создания нового трека

В окне создания новой записи пользователю будет предложено выбрать тип создаваемого элемента и отредактировать его характеристики. В некоторых видах записей пользователю также может быть предложено загрузить файл с локального хранилища устройства.

При нажатии на какой-либо из плейлистов или альбомов пользователю будет предложен перечень треков, входящих в него (см. рис. 3.5).

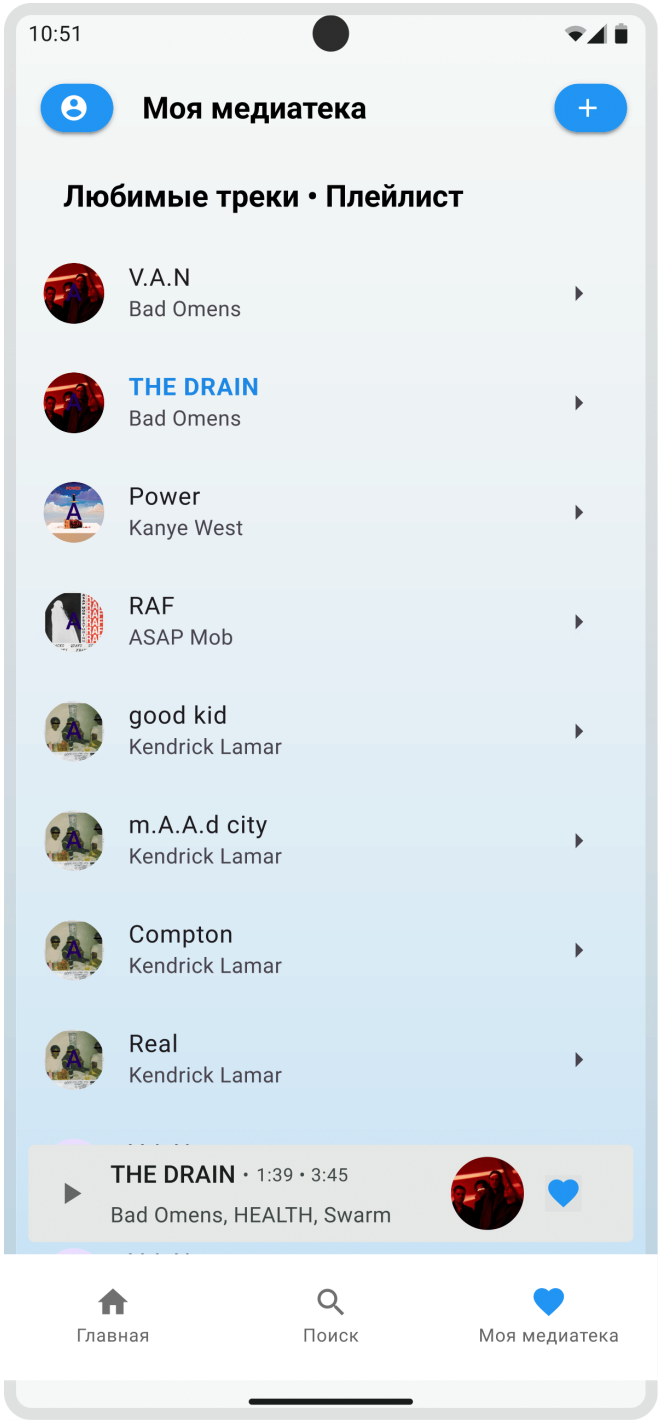

Рисунок 3.5. Плейлист «Любимые треки»

Пользователь также может перейти и просмотреть профиль другого пользователя. В нём он сможет увидеть популярные треки пользователя, последние записи; прослушать все его треки, подписаться на профиль пользователя (см. рис. 3.6).

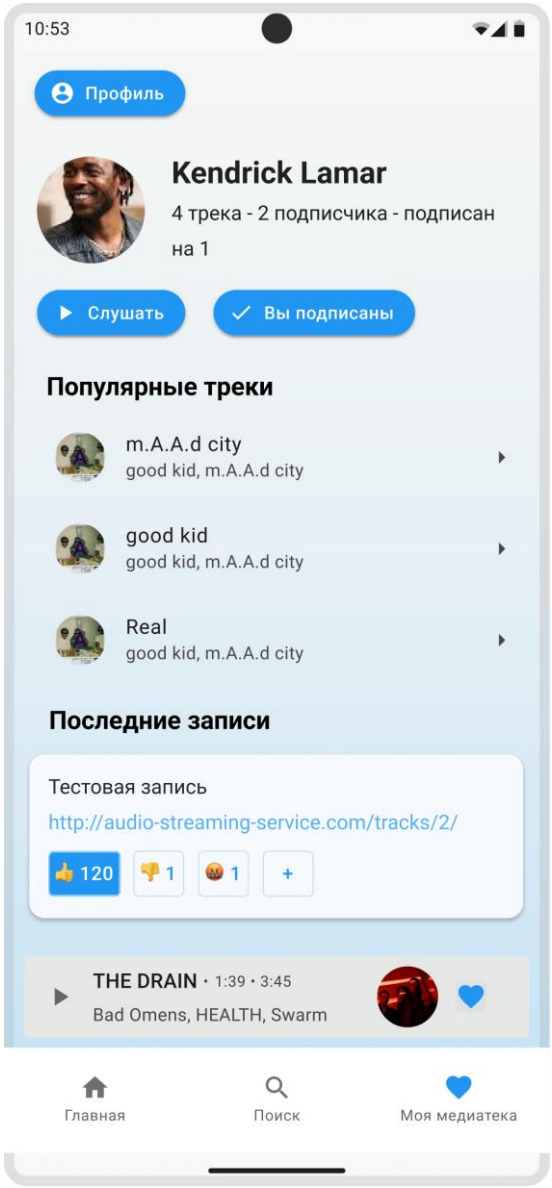

Рисунок 3.6. Профиль пользователя

По нажатию на любой из треков плейлиста, альбома или пользователя можно начать его прослушивания. Интерфейс проигрывателя доступен в нижней части интерфейса центральной части приложения, где отображается основная информация о проигрываемой записи, статус добавления в список любимых и возможность приостановить/воспроизвести трек. По нажатию на это панель пользователю будет доступен интерфейс плеер (см. рис. 3.7).

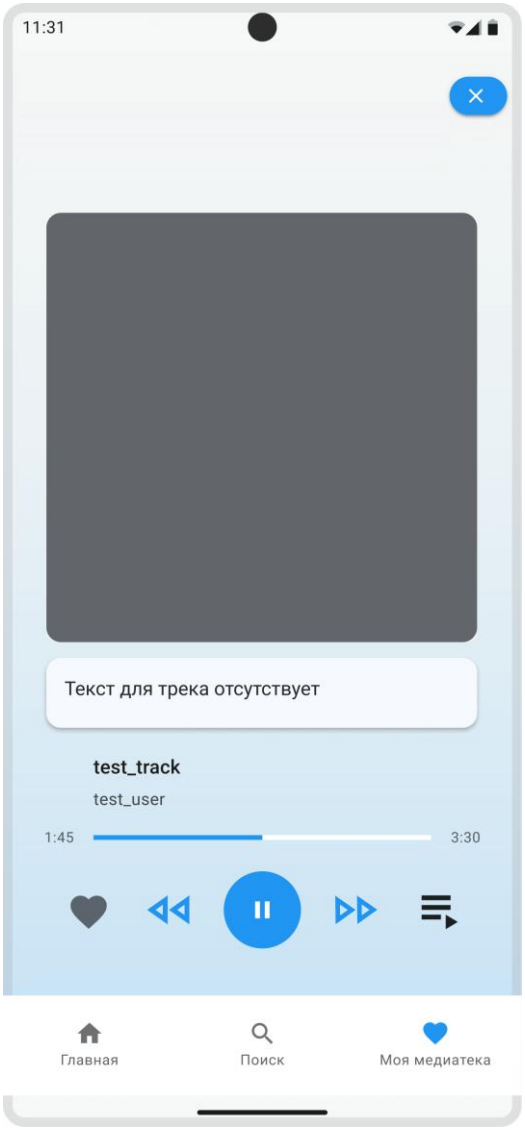

Рисунок 3.7. Интерфейс плеера

В плеере пользователь может управлять воспроизведение композицией (посредством кнопок «Проигрывание/пауза», «Вперёд», «Назад» и изменением позиции проигрывания слайдером), поставить отметку нравиться, посмотреть очередь воспроизведения и текст композиции.

3.3. Руководство администратора

После авторизации администратору будет доступна соответствующая панель управления пользователями, треками, альбомами, плейлистами, а также просмотр заявок пользователей. Изначально администратор попадает на страницу «Пользователи» (см. рис. 3.8). На ней он может выполнять поиск по учётным записям пользователей, выполнять фильтрацию по заданным фильтрам и переходить на найденную запись пользователя.

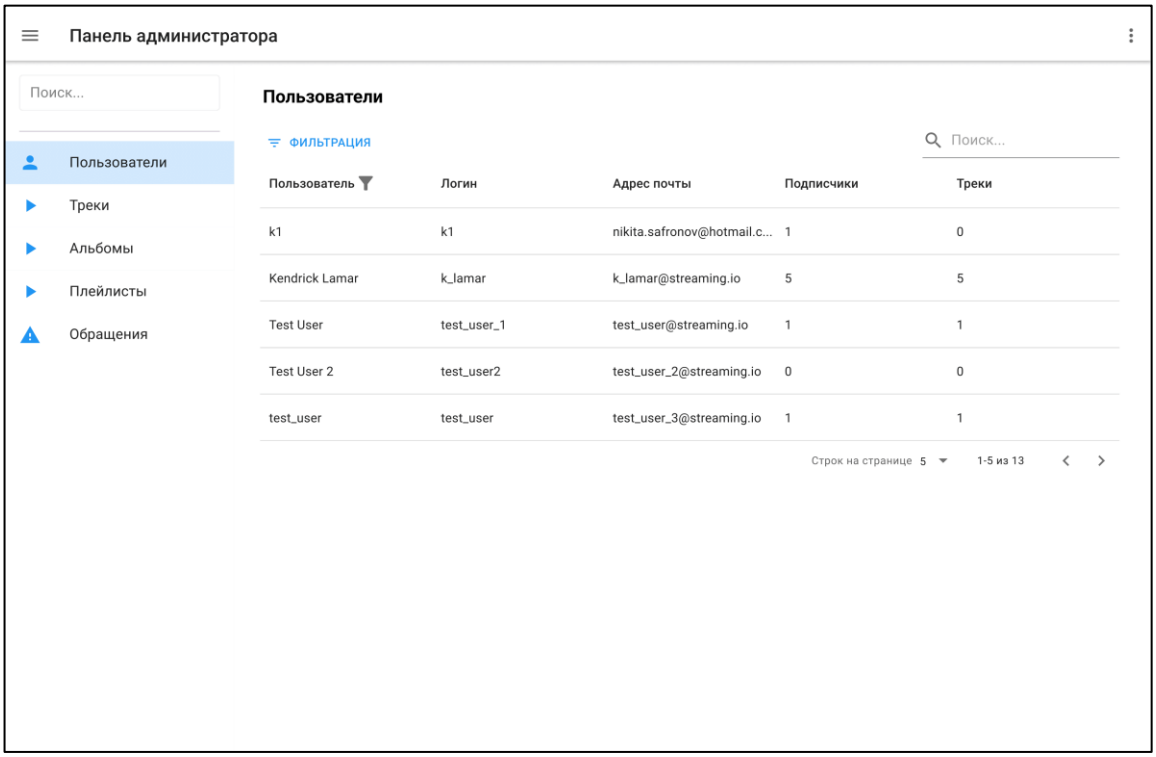

Рисунок 3.8. Страница «Пользователи»

Страница профиль отображает основную информацию о профиле пользователя, осуществить поиск его подписчиков, треков, альбомов, плейлистов и обращений (см. рис. 3.9). Для управления администратор может редактировать учётную запись пользователя, а также удалить его учётную запись.

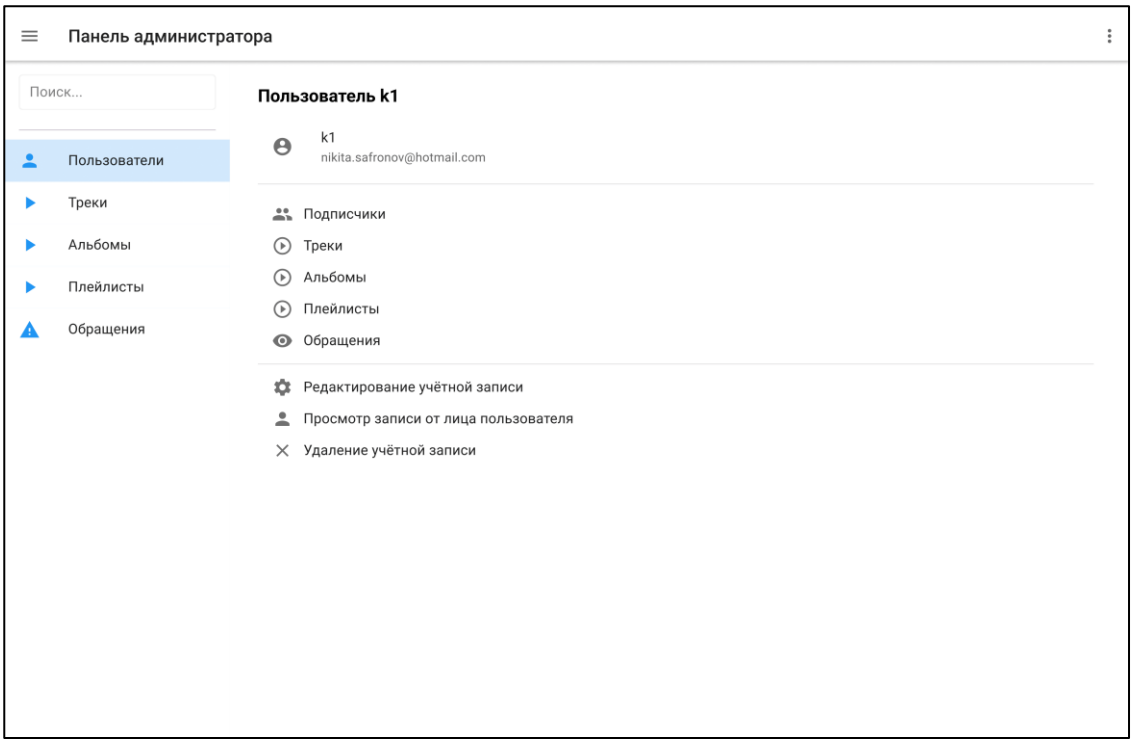

Рисунок 3.9. Страница «Профиль пользователя»

Редактирование профиля пользователя представлено диалоговым окном, представленным на рисунке 3.10.

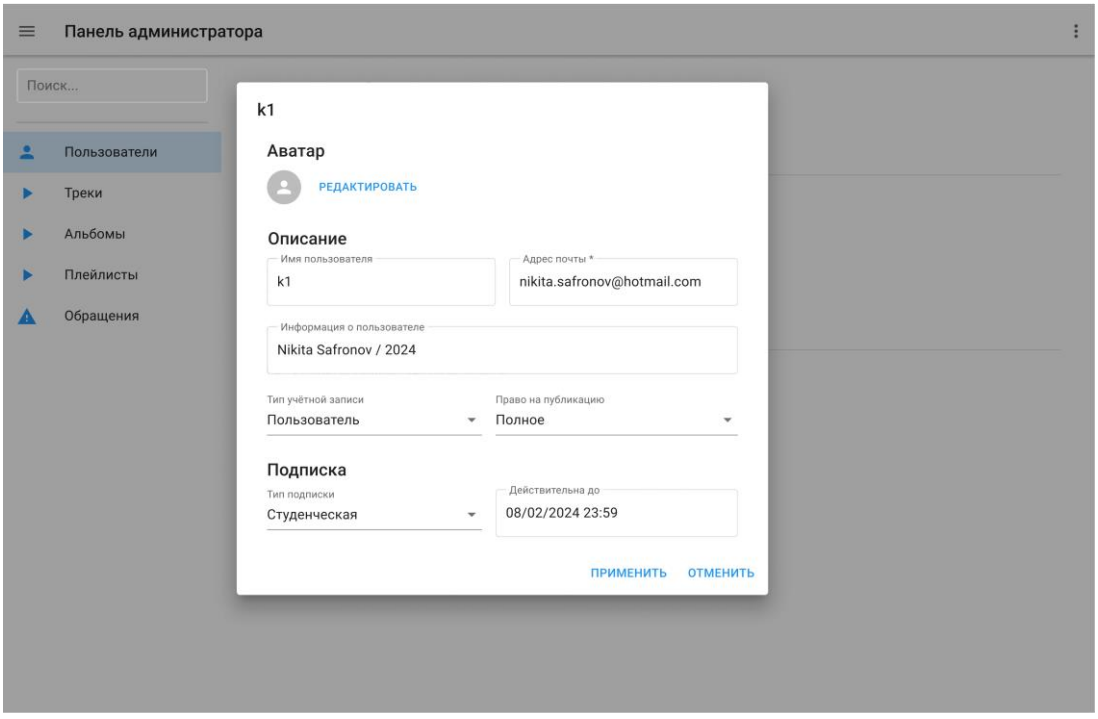

Рисунок 3.10. Диалоговое окно «Редактирование профиля пользователя»

Остальные страницы, посвящённые альбомам, трека и плейлистам копируют подобную структуру, имеющую центральной частью таблицу записей с возможностью фильтрации и поиска, а также редактирования выбранного элемента.

Элементом, являющимся основным для администратора, является страница «Обращения». Здесь будут собраны обращения пользователей, а также их жалобы. Администратор может выбирать то обращение, с которым он хочет работать, из таблицы (см. рис. 3.11).

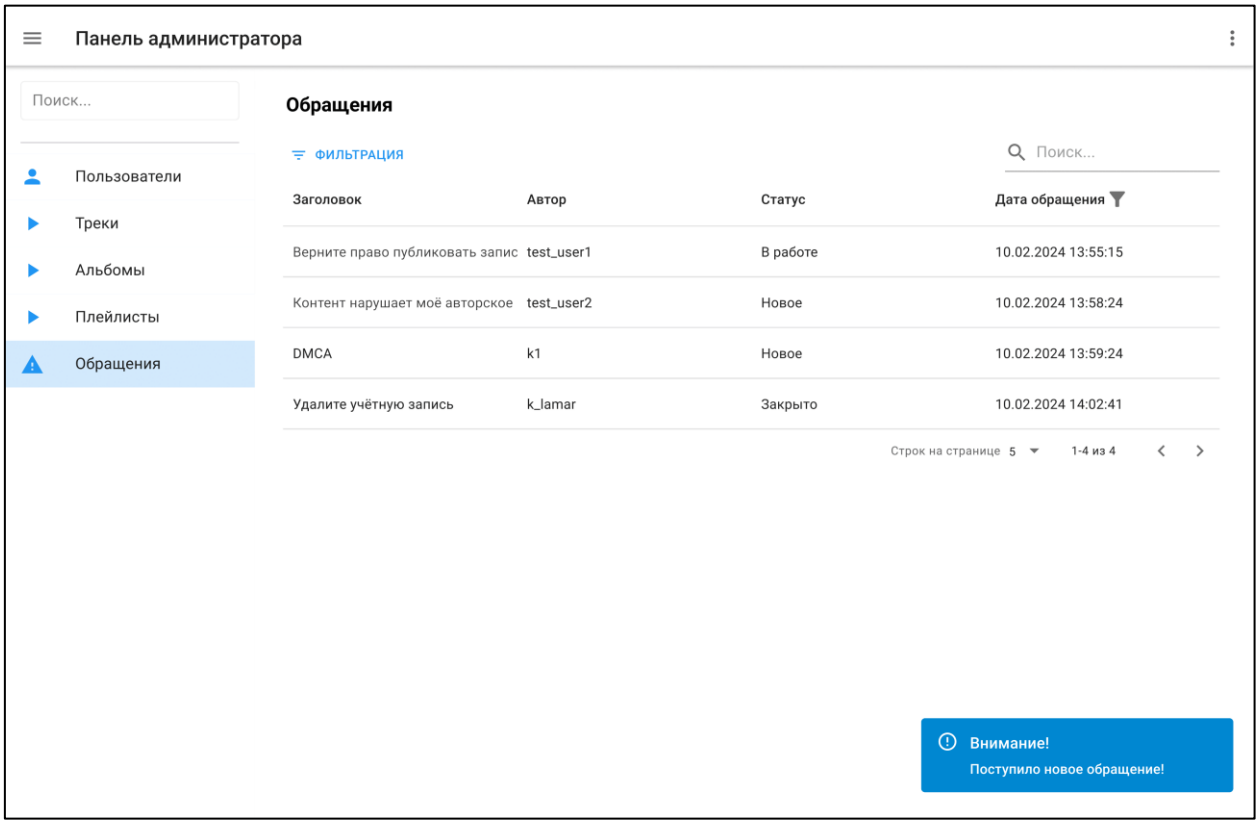

Рисунок 3.11. Страница «Обращения»

По нажатию на одно из обращений откроется окно «Обращение» (см. рис. 3.12), где администратор может просмотреть текст обращения, связанные с ним объекты, его статус, представленный одним из трёх значений: «Принято», «В работе», «Закрыто». Администратор может отвечать пользователю внутри создаваемого для работы по заявке чата, а также менять статус заявки.

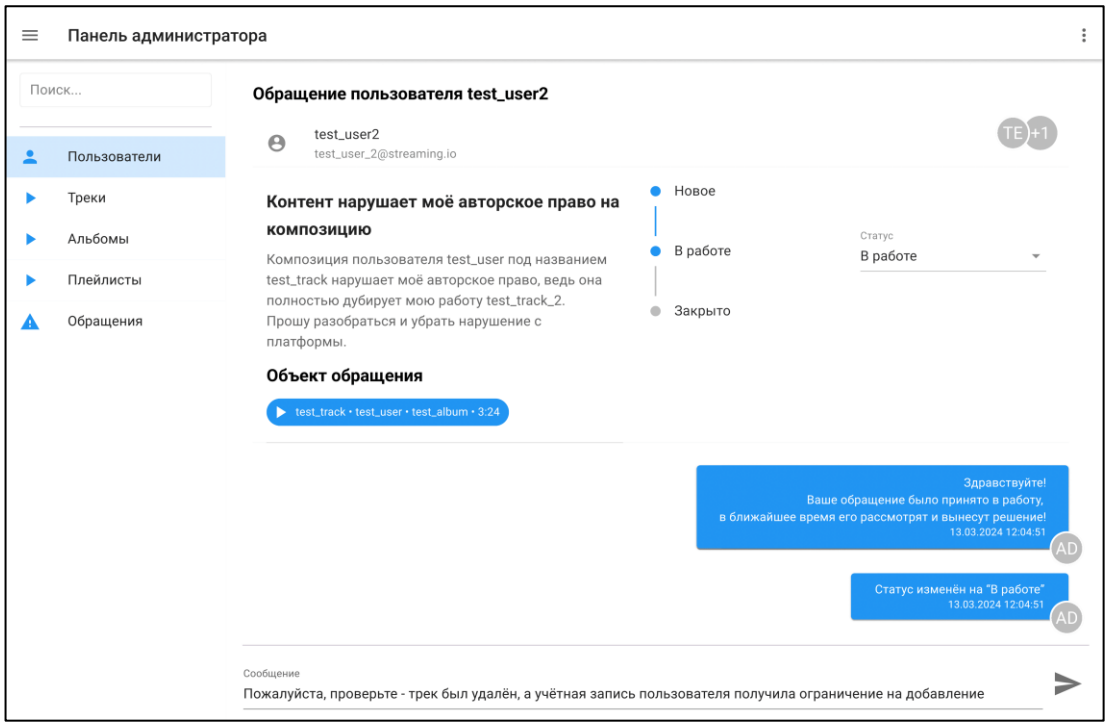

Рисунок 3.12. Страница «Обращение»

Вывод

Реализованная мобильная версия и панель администратора сервиса предоставляет заявленный функционал. Разработанный интерфейс является интуитивно понятным и позволяет пользователем ориентироваться в приложении без сложностей. В целях упрощения понимания использования сервиса указаны основные сценарии взаимодействия пользователя с мобильным приложением и интерфейсом панели администратора.

ЗАКЛЮЧЕНИЕ

В результате выполнения курсовой работы была создана мобильная версия и панель администратора социальной сети с функциями музыкального стримингового сервиса. Результат соответствует всем заранее определенным требованиям технического задания.

Созданное приложение основано на следующем стеке технологий: для разработки клиентской части: React Native и Redux; для разработки серверной части - СУБД PostgreSQL и SQLAlchemy, Flask и aiohttp; веб-сервер Nginx; брокер сообщений Kafka; Docker для удобства развёртывания.

В работе были подробно описаны развёртывание и настройка приложения.

Разработанное приложение обладает простым и понятным пользовательским интерфейсом, что позволит использовать веб-приложение любому пользователю.

Благодаря удачно спроектированной архитектуре клиентского и серверного приложений на основе микросервисов, при необходимости можно улучшить и расширить разработанную систему путем добавления дополнительного функционала, к примеру:

— Нейросетевой модуль, генерирующие тексты к трекам, для которых они не были добавлены;

— Модуль рекомендательной системы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Основная литература

1. Влацкая И.В. Проектирование и реализация прикладного программного обеспечения [Электронный ресурс]: учебное пособие / И.В. Влацкая, Н.А. Заельская, Н.С. Надточий. — Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2015. — 119 с. — Режим доступа: http://www.iprbookshop.ru/54145.html

2. Грекул В.И. Проектирование информационных систем. Курс лекций [Электронный ресурс] : учебное пособие для студентов вузов, обучающихся по специальностям в области информационных технологий / В.И. Грекул, Г.Н. Денищенко, Н.Л. Коровкина. — Москва, Саратов: Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017. — 303 с. — Режим доступа: http://www.iprbookshop.ru/67376.html.

3. Золотов С.Ю. Проектирование информационных систем [Электронный ресурс] : учебное пособие / С.Ю. Золотов. — Томск: Томский государственный университет систем управления и радиоэлектроники, Эль Контент, 2013. — 88 с. — Режим доступа: http://www.iprbookshop.ru/13965.html.

4. Митина О.А. Методы и средства проектирования информационных систем и технологий [Электронный ресурс]: курс лекций / О.А. Митина. — М. : Московская государственная академия водного транспорта, 2016. — 75 с. — Режим доступа: http://www.iprbookshop.ru/65666.html.

Дополнительная литература

5. Антонов В.Ф. Методы и средства проектирования информационных систем [Электронный ресурс] : учебное пособие / В.Ф. Антонов, А.А. Москвитин. — Ставрополь: Северо-Кавказский федеральный университет, 2016. — 342 с. — Режим доступа: http://www.iprbookshop.ru/66080.html

6. Проектирование информационных систем. Проектный практикум [Электронный ресурс] : учебное пособие для студентов дневного и заочного отделений, изучающих курсы «Проектирование информационных систем», «Проектный практикум», обучающихся по направлению 230700.62 (09.03.03) / А.В. Платёнкин [и др.]. — Тамбов: Тамбовский государственный технический университет, ЭБС АСВ, 2015. — 80 с.— Режим доступа: http://www.iprbookshop.ru/64560.html

7. Стасышин В.М. Проектирование информационных систем и баз данных [Электронный ресурс] : учебное пособие — Новосибирск : Новосибирский государственный технический университет, 2012, с. 100. — URL: http://www.iprbookshop.ru/45001.html

8. Беллермар, А. Создание событийно-управляемых микросервисов. – СПб.: БХВ-Петербург, 2022 – 320 с.

9. Гамма, З., Хелм, Р., Джонсон, Р., Влиссидес, Дж. Приёмы объектноориентированного проектирования. Паттерны проектирования. — СПб.: Питер, 2016. — 368 с.: ил.

10. Хортон, А. Разработка веб-приложений в ReactJS / А. Хортон, Р. Вайс ; перевод с английского Р. Н. Рагимова. — Москва : ДМК Пресс, 2016. — 254 с. — URL: https://e.lanbook.com/book/97339

11. Лоре, А. Проектирование веб-API : руководство / А. Лоре ; перевод с английского Д. А. Беликова. — Москва : ДМК Пресс, 2020. — 440 с. — URL: https://e.lanbook.com/book/179498

12. Осипов, Д. Л. Технологии проектирования баз данных / Д. Л. Осипов. — Москва : ДМК Пресс, 2019. — 498 с. — URL: https://e.lanbook.com/book/131692

13. Ричардсон, К. Микросервисы. Паттерны разработки и рефакторинга – СПб.: Питер, 2019. – 544 с.

14. Диков, А. В. Web-программирование на JavaScript : учебное пособие для спо / А. В. Диков. — Санкт-Петербург : Лань, 2021. — 168 с. — URL: https://e.lanbook.com/book/156625

15. Зудилова, Т. В. Web-программирование JavaScript : учебно-методическое пособие / Т. В. Зудилова, М. Л. Буркова. — Санкт-Петербург : НИУ ИТМО, 2012. — 68 с. — URL: https://e.lanbook.com/book/43561

16. Баланов, А. Н. Комплексное руководство по разработке: от мобильных приложений до веб-технологий : учебное пособие для вузов / А. Н. Баланов. — Санкт-Петербург : Лань, 2024. — 412 с. — URL: https://e.lanbook.com/book/394577.

17. Заяц, А. М. Введение в гибридные технологии разработки мобильных приложений : учебное пособие для спо / А. М. Заяц, Н. П. Васильев. — 2-е изд., стер. — Санкт-Петербург : Лань, 2022. — 160 с. — URL: https://e.lanbook.com/book/200459.