МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет *компьютерных наук*

Кафедра *информационных систем*

Каталог музыкальных произведений, записанных в электронном формате «MusicZone»

Курсовой проект

по дисциплине

Технологии программирования

09.03.02 Информационные системы и технологии

Информационные системы в телекоммуникациях

6 семестр 2022/2023 учебного года

Зав. кафедрой _____ *к. т. н., доцент С.Д. Махортов*

Обучающийся _____ *ст. 3 курса оч. отд. Р. С. Шевцов*

Обучающийся _____ *ст. 3 курса оч. отд. М. В. Гончаренко*

Обучающийся _____ *ст. 3 курса оч. отд. М. С. Артемьев*

Руководитель _____ *В.С. Тарасов, ст. преподаватель __.__.20__*

Воронеж 2023

Содержание

Введение

В современном мире мобильные приложения стали неотъемлемой частью нашей повседневной жизни, предоставляя нам доступ к различным сервисам и удобствам на расстоянии нажатия кнопки. Одной из популярных категорий мобильных приложений являются музыкальные приложения, которые с успехом сочетают в себе развлекательную и практическую функции.

Музыкальные приложения – это программные приложения для мобильных устройств, предназначенные для прослушивания, организации и обмена музыкой. Они позволяют пользователям наслаждаться музыкой в любое время и в любом месте, предлагая широкий спектр возможностей, включая потоковую передачу музыки, создание плейлистов, поиск новых артистов и треков, радиостанции и многое другое. Благодаря мобильным музыкальным приложениям, музыка стала более доступной и персонализированной, позволяя нам настроиться на нужное настроение или создать собственный саундтрек к повседневным моментам нашей жизни.

В данной курсовой работе рассматривается процесс разработки собственного мобильного музыкального приложения

В рамках работы будут рассмотрены различные аспекты разработки мобильного музыкального приложения, начиная с анализа предметной области, определения его концепции и основных требований. Затем будет изучено проектирование пользовательского интерфейса и пользовательского опыта, с учетом современных тенденций и личных практик в этой области. Важное внимание будет уделено выбору и интеграции соответствующих технологий и API для обеспечения необходимых функций, таких как передача музыки, поиск треков и другие.

1 Постановка задачи

1.1 Цели создания системы

Целью данной работы является создание мобильного музыкального приложения для поиска и прослушивания музыки с дальнейшим сохранением на личной странице пользователя.

1.2 Требования к разрабатываемой системе

— обеспечение авторизации и аутентификации пользователей;

— использование механизмов защиты от SQL-инъекций;

— использование протокола передачи данных HTTP;

— приложение должно быть построено на трехуровневой архитектуре.

1.3 Задачи проекта

— поиск музыки;

— прослушивание найденной музыки;

— добавление музыки на личную страницу;

— удаление музыки из личной страницы;

— загрузка музыки администратором приложения;

— редактирование музыки администратором приложения;

— удаление музыки администратором приложения.

2 Анализ предметной области

2.1 Терминология

Таблица 1 - Глоссарий

Мобильное приложение	Программное обеспечение, предназначенное для работы на смартфонах, планшетах и других мобильных устройствах, разработанное для конкретной платформы.
Клиент	Это аппаратный или программный компонент вычислительной системы, посылающий запросы серверу.
Сервер	Выделенный или специализированный компьютер для выполнения сервисного программного обеспечения.
База данных	это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляется системой управления базами данных.
HTTP	Это протокол, позволяющий получать различные ресурсы, например HTML-документы. Протокол HTTP лежит в основе обмена данными в Интернете.
SQL-запросы	Это наборы команд для работы с реляционными базами данных.
Дизайн-макет	Это схематичное изображение финальной идеи с указанием всех деталей. В нем указываются концепция, шрифты, тексты, изображения, расположение всех элементов и общая картина продукта.

Аутентификация	Процедура проверки подлинности, например, проверка подлинности пользователя путем сравнения введенного им пароля с паролем, сохраненным в базе данных.
Авторизация	Предоставление определенному лицу или группе лиц прав на выполнение определенных действий.
Android	Это операционная система с открытым исходным кодом, созданная для мобильных устройств на основе модифицированного ядра Linux.
Фреймворк	Программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.
SQL-инъекция	Внедрении в запрос произвольного SQL-кода, который может повредить данные, хранящиеся в БД или предоставить доступ к ним.
Аккаунт	Это персональная страница пользователя или личный кабинет, который создается после регистрации на сайте.
REST	Архитектурный стиль взаимодействия компонентов распределённого приложения в сети.
API	Описание взаимодействия одной компьютерной программы с другой.
Трек	Это любая звуковая дорожка,

	электронная музыкальная композиция.
Песня	Это композиция, состоящая из поэтического текста и мелодии. Они выполняют равные смысловые функции.

2.2 Обзор аналогов

Разрабатывая приложение, основной задачей которого является хранение и прослушивание музыки, необходимо рассматривать разработку с точки зрения актуальности и уникальности проекта. Для оценки этих качеств необходимо прибегнуть к рассмотрению аналогов разрабатываемого приложения, адекватно оценивая все положительные и негативные черты того или иного продукта.

2.2.1 Яндекс.Музыка

Яндекс.Музыка — это онлайн-сервис, предоставляющий простой и удобный доступ к миллионам аудио-треков, подборкам, плейлистам, радиостанциям и музыкальным подкастам. Яндекс.Музыка рекомендует своим пользователям новинки и композиции, исходя из их личных предпочтений, возраста, настроения и времени суток. Также сервис позволяет создавать персональные плейлисты, синхронизировать музыку между разными устройствами. Яндекс.Музыка доступна в большинстве стран.

Яндекс.Музыка обладает широким спектром предоставляемых услуг и с точки зрения авторского контента. На рисунке 1 представлен главный экран данного приложения.[1]

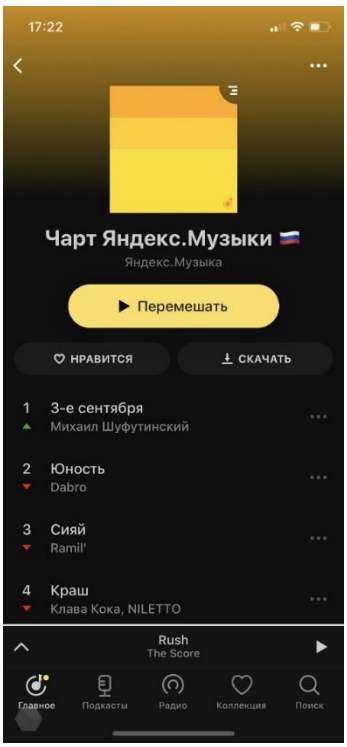


Рисунок 1 - Интерфейс страницы «Яндекс.Музыка»

Недостатки:

— ограниченный доступ к музыкальным трекам;

— высокая стоимость подписки на сервис;

— рекламные блоки на платформе могут быть раздражающими;

— неудобство в использовании приложения на некоторых устройствах;

— неполное совпадение исполнителей, альбомов и треков;

— не удается найти все композиции и исполнителей;

— невозможно загрузить личную музыку в плейлисты;

— вирусные подборки, которые появляются в рекомендациях и не соответствуют запросам пользователя.

2.2.2 YouTubeMusic

YouTubeMusic — это онлайн-музыкальный сервис, представленный компанией YouTube в 2018 году. Сервис предоставляет своим пользователям доступ к миллионам песен и музыкальным видео, а также плейлистам, радиостанциям и рекомендациям на основе предпочтений пользователя.

Стоит отметить, что на фоне многих стриминговых сервисов именно продукт Google отличается хорошо работающими рекомендациями, что идеально могут скрасить вечер слушателя. В плане работы с рекомендациями данный продукт конкурирует с такими стриминговыми сервисами как Spotify, Dizzer и т.д.

YouTubeMusic доступен для использования на различных устройствах, таких как компьютеры, смартфоны и планшеты, и поддерживает различные операционные системы, включая Android и iOS. Он также совместим с другими сервисами Google, такими как GooglePlay Музыка.

Сервис предоставляет возможность прослушивания музыки как в режиме онлайн, так и офлайн, в зависимости от настроек пользователя. Он также включает функции автоматического плейлиста и адаптивного режима, который позволяет пользователям настроить оптимальное качество звука в зависимости от доступной связи с интернетом. На рисунке 2 представлен главный экран данного приложения.[2]

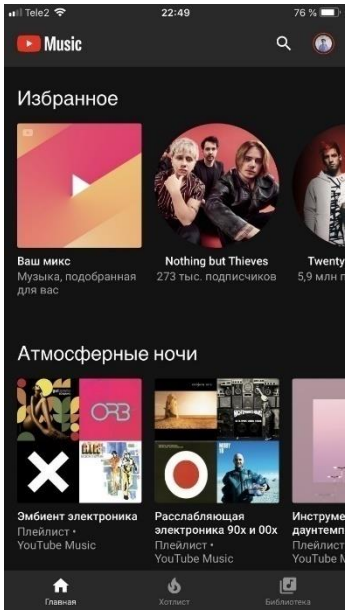


Рисунок 2 - Интерфейс страницы «YouTubeMusic»

Недостатки:

— громоздкий интерфейс;

— плохая работа с карточками музыкантов;

— различное качество звука от трека к треку;

— отсутствие некоторых функций, таких как создание временного плейлиста.

2.2.3 Spotify

Spotify — это стриминговый сервис, предлагающий доступ к миллионам песен, подкастам и аудиокнигам. Spotify позволяет пользователям создавать плейлисты, слушать музыку в режиме онлайн или офлайн (сохранение в кэш), делиться треками с друзьями, искать новых исполнителей и многое другое. Spotify также имеет мобильное приложение и многочисленные интеграции с другими платформами и устройствами.

На данный момент Spotify — это флагман от мира стриминговых сервисов. Его алгоритмы подборки очень точны, а работа с артистами, посредством взаимодействия через цифровой музыкальный дистрибьютор, самая удобная из существующих на данный момент.

Spotify предлагает широкий спектр услуг по распространению музыки определенных исполнителей, включая треки даже малоизвестных исполнителей по жанрам и стилю подходящие под образец в подборки, которые рекомендуются пользователям с соответствующими музыкальными предпочтениями. На рисунке 3 представлен главный экран данного приложения.[3]

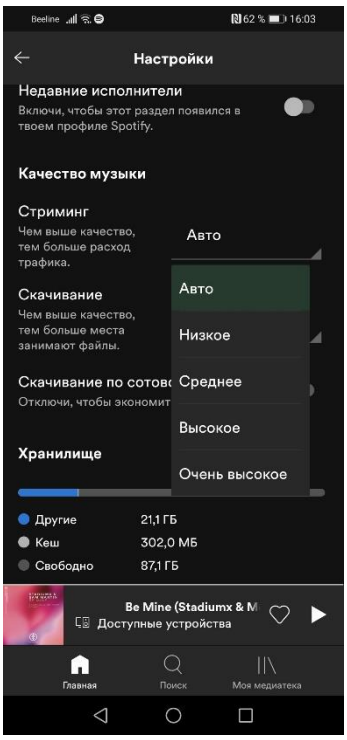


Рисунок 3 - Интерфейс страницы «Spotify»

Недостатки:

— Spotify не доступен в некоторых странах мира, что ограничивает число потенциальных пользователей;

— ограниченный доступ к музыке, в частности, к новым альбомам известных артистов;

— Spotify потребляет намного больше интернет-трафика, чем другие сервисы.

2.3 Моделирование системы

2.3.1 Диаграмма в стиле методологии IDEF0

IDF0 диаграмма представляет собой графическое представление бизнес-процесса в виде иерархической структуры функций. Основная цель IDF0 диаграммы состоит в том, чтобы показать, как различные функциональности взаимодействуют друг с другом и как они влияют на достижение целей организации. Она помогает улучшить понимание процессов и оптимизировать их для повышения эффективности организации. Данная диаграмма представлена на рисунке 4.

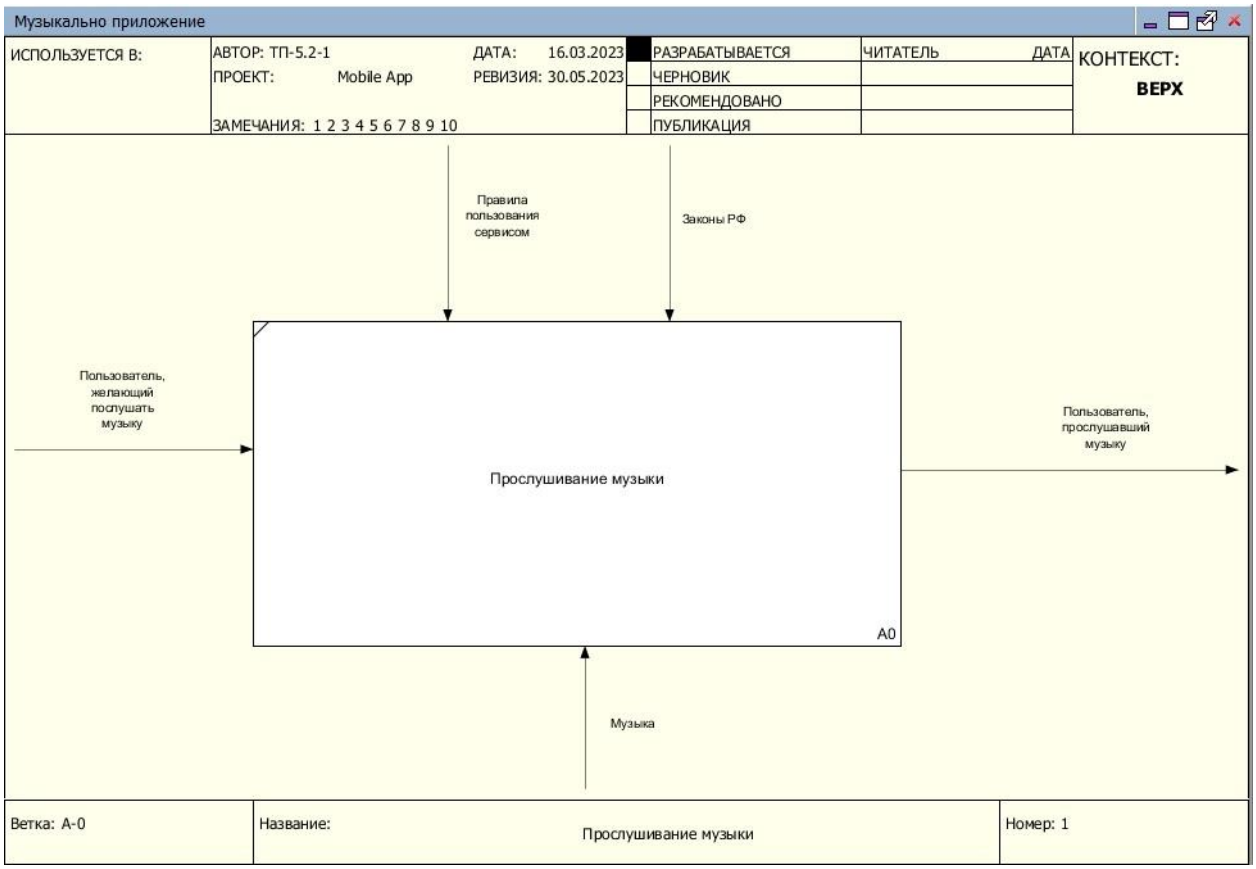


Рисунок 4 - Диаграмма в стиле методологии IDEF0

2.3.2 Диаграмма прецедентов

Рассмотрим полную диаграмму для использования приложения разными типами пользователей. В данном случае необходимость составления диаграммы прецедентов продиктована прежде всего тем, что use-case диаграмма — это инструмент для моделирования системы и понимания ее функциональности и потребностей пользователей. Они помогают в определении основных действий, которые пользователь должен совершить в системе, чтобы достичь определенных целей. Они также позволяют определить возможные риски и проблемы, которые могут возникнуть в ходе использования системы. Данная диаграмма представлена на рисунке 5.

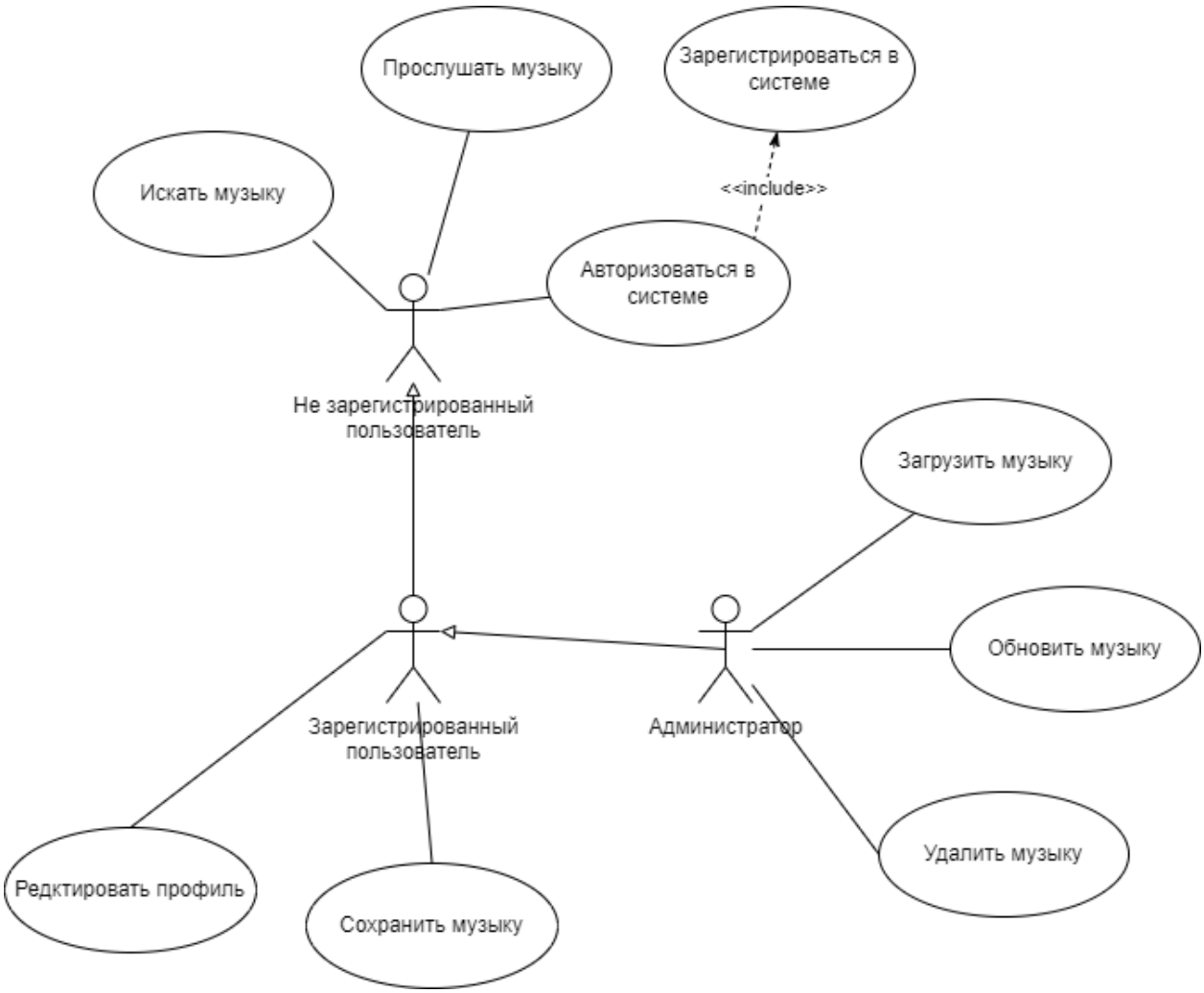


Рисунок 5 - Use-Case диаграмма пользования приложением

2.3.3 Диаграммы последовательности

Диаграмма последовательности является важным инструментом для проекта, который помогает более глубоко понимать процесс, улучшать его эффективность и упрощать взаимодействие.

Рассмотрим диаграмму последовательности администратора приложения. Данные диаграммы представлена на рисунках 6-8.

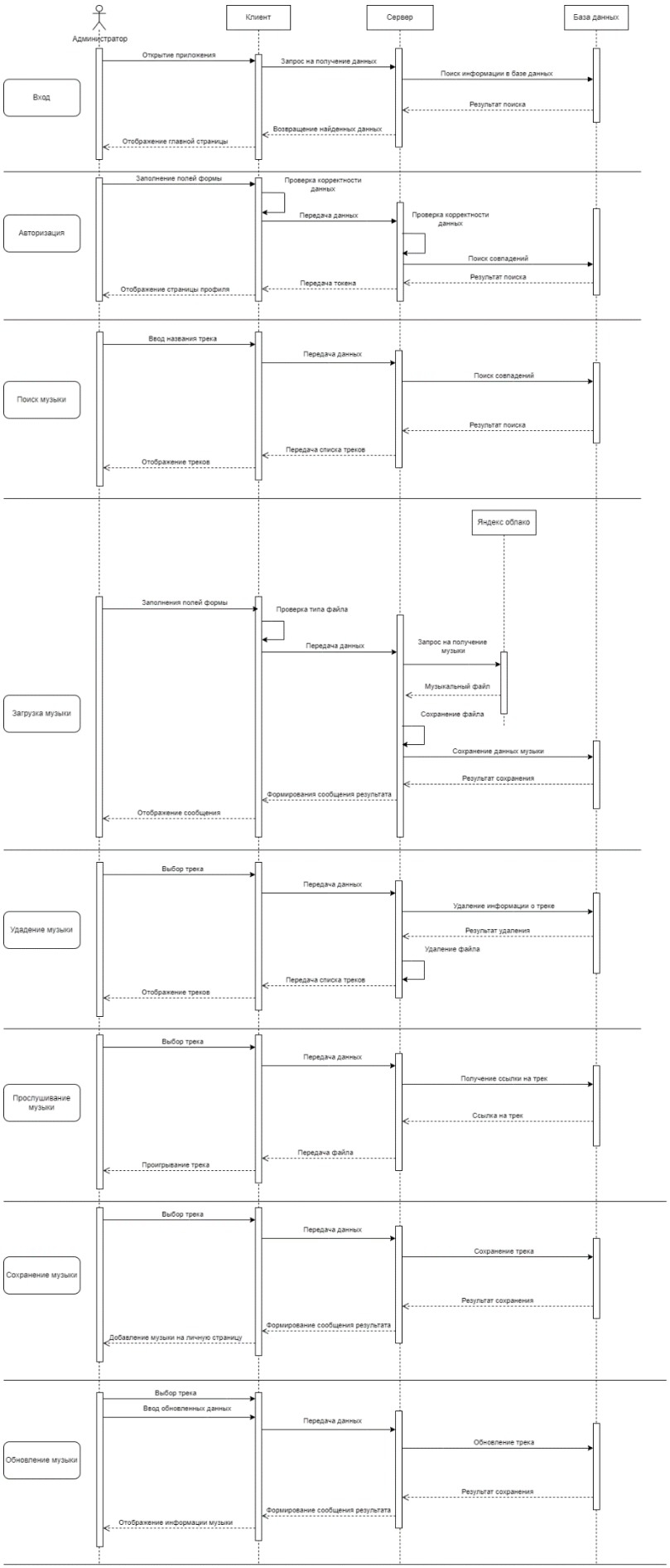


Рисунок 6 - Диаграмма последовательности администратора

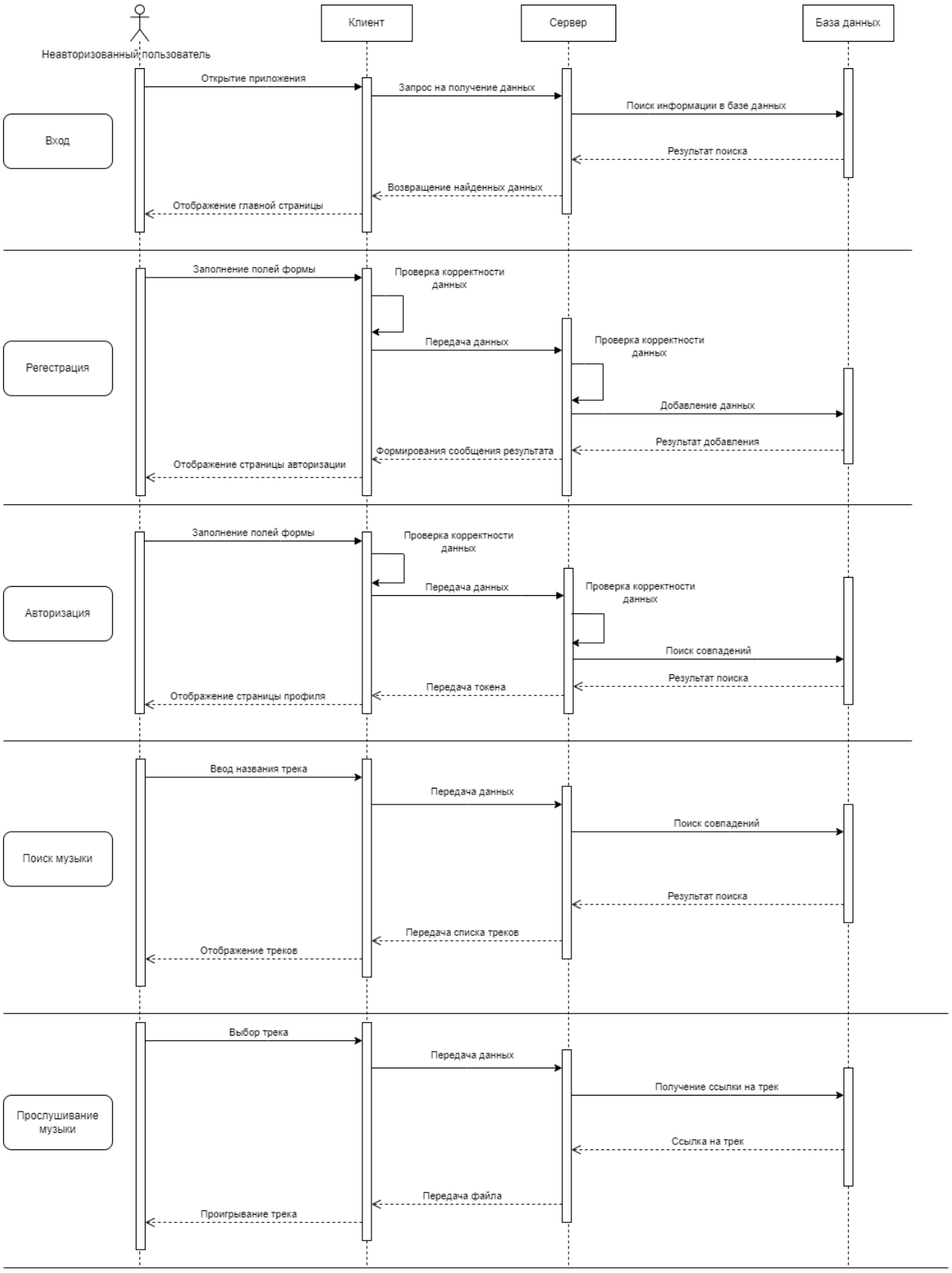


Рисунок 7 - Диаграмма последовательности неавторизированного пользователя

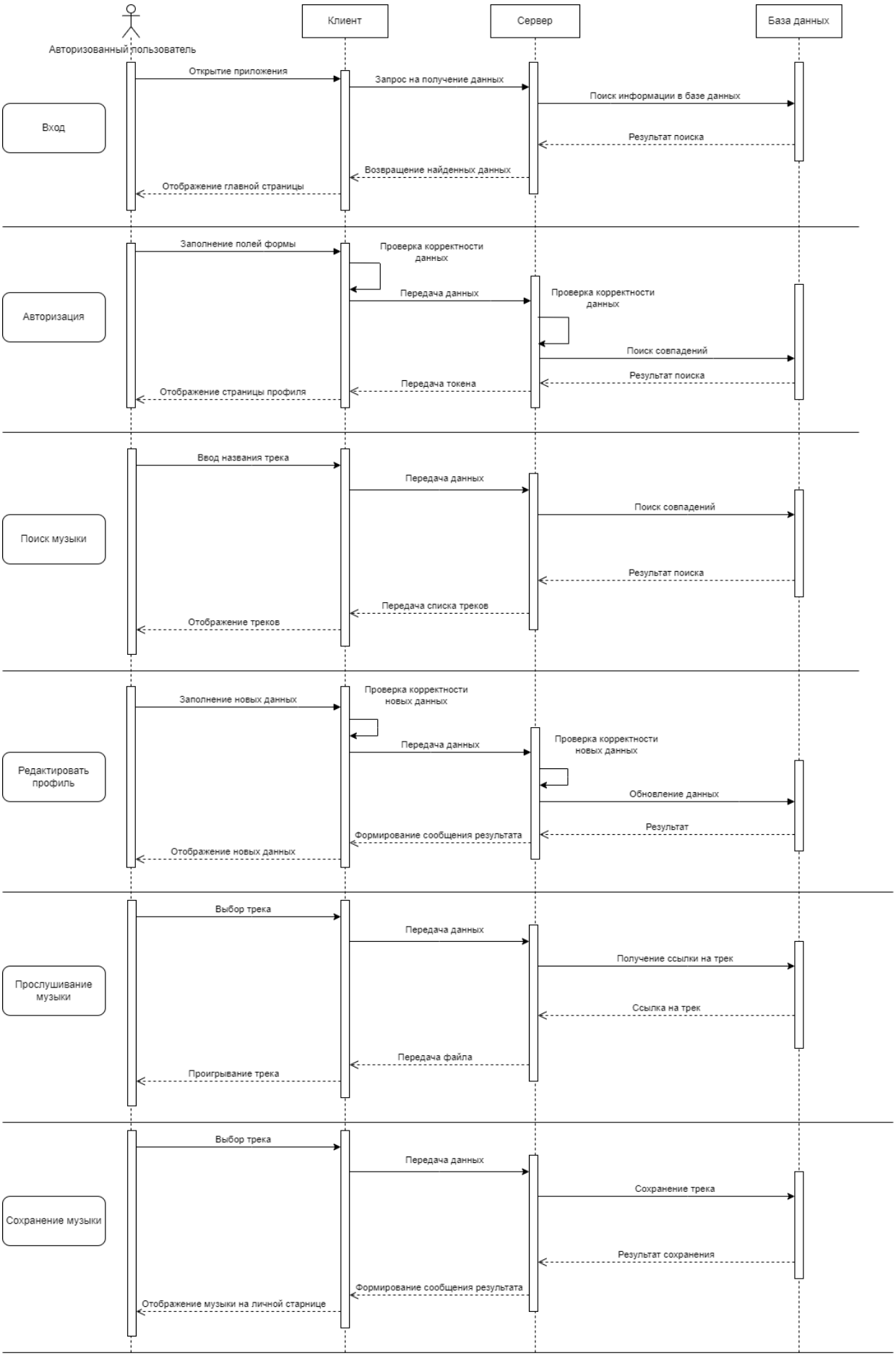


Рисунок 8 - Диаграмма последовательности авторизованного пользователя

2.3.4 Диаграмма развертывания

Диаграмма развертывания позволяет определить требования к аппаратному обеспечению, планировать установку и настройку компонентов системы, а также оценивать ее производительность и масштабируемость. Данная диаграмма представлена на рисунке 9.

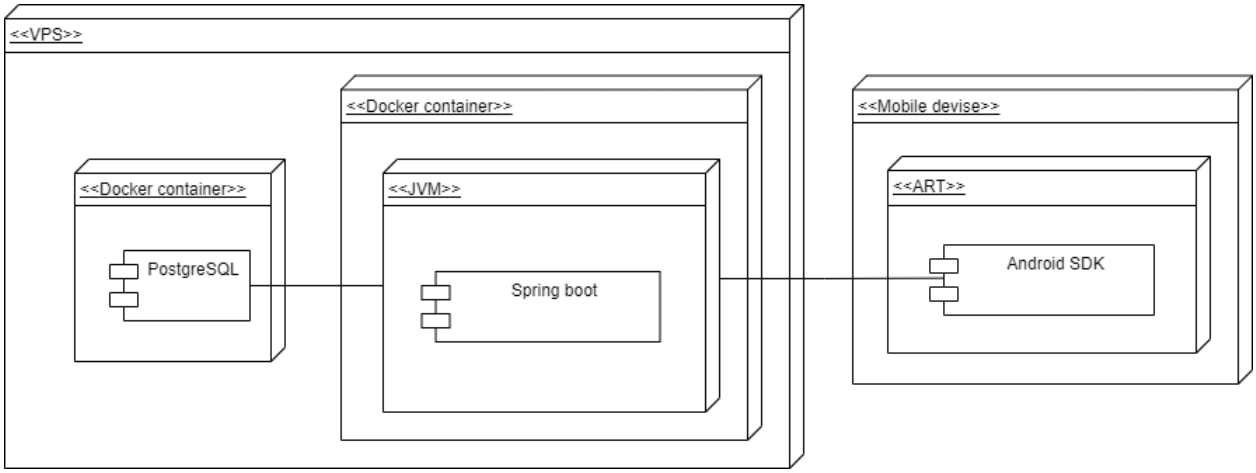


Рисунок 9 - Диаграмма развертывания приложения

2.3.5 Диаграммы состояния

Диаграмма состояния позволяет определить возможные сценарии поведения системы, выделить ключевые состояния и переходы между ними, а также оценить ее надежность и устойчивость к ошибкам. Для нашего проекта были спроектированы 3 диаграммы для состояний администратора, зарегистрированного пользователя и незарегистрированного пользователя. Данные диаграммы представлены на рисунках 10-12.

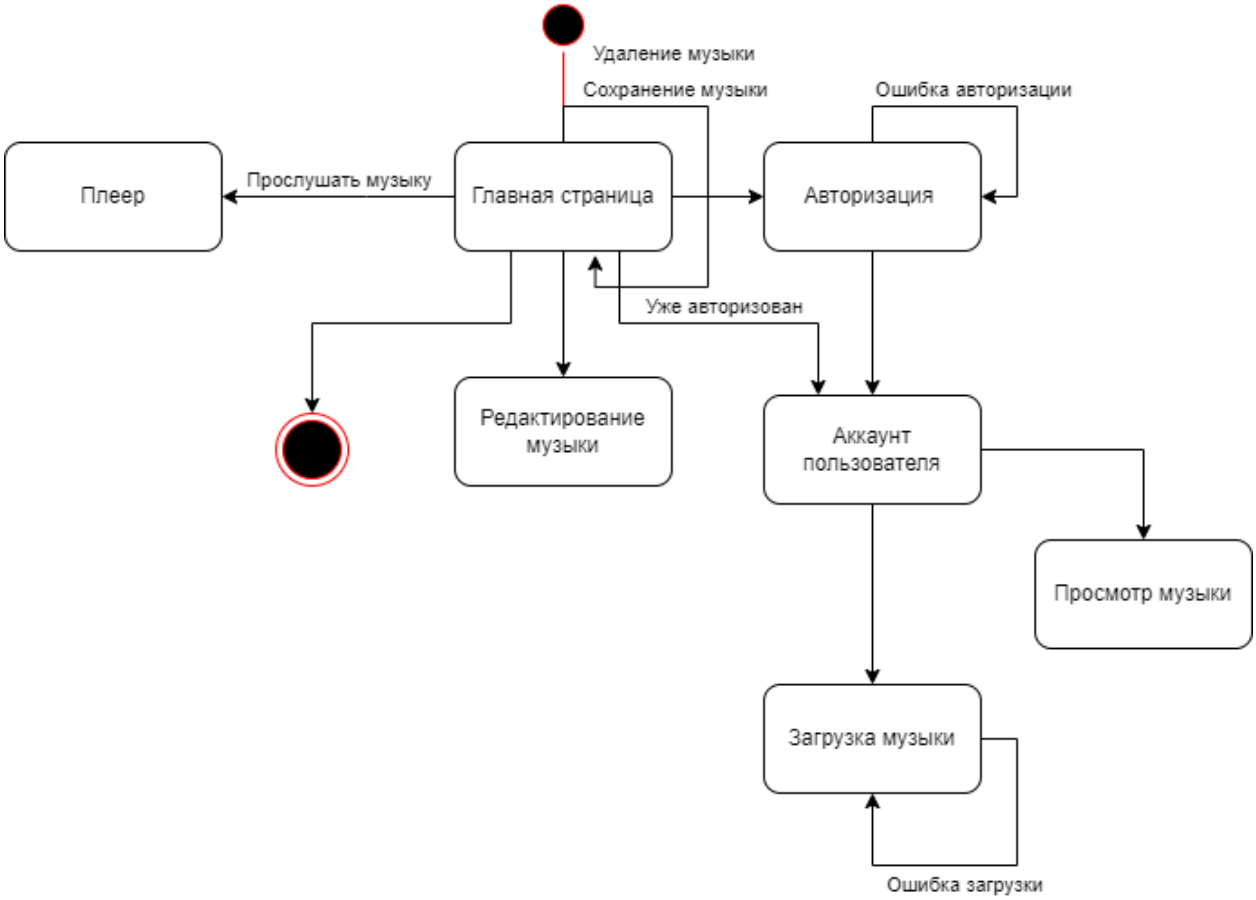


Рисунок 10 - Диаграмма состояния администратора

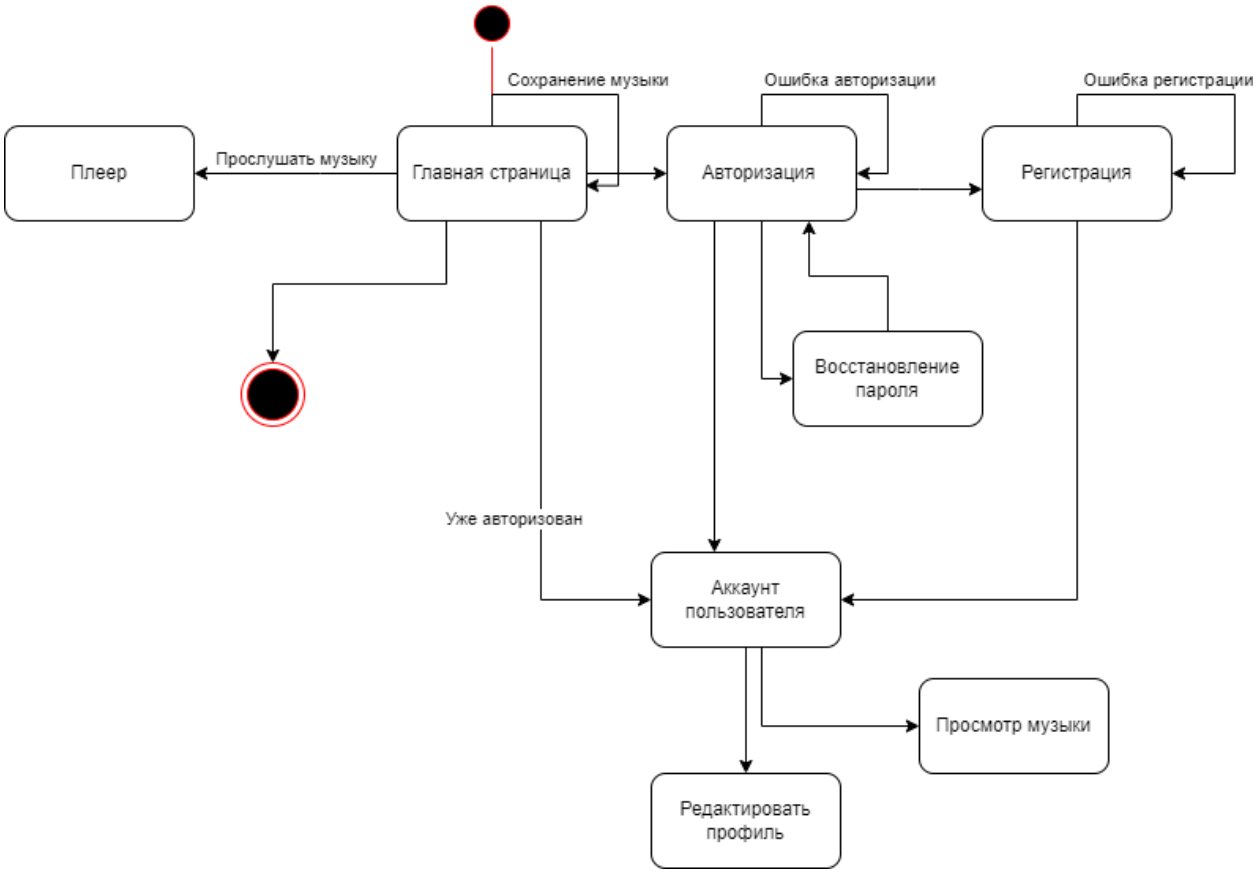


Рисунок 11 - Диаграмма состояния неавторизованного пользователя

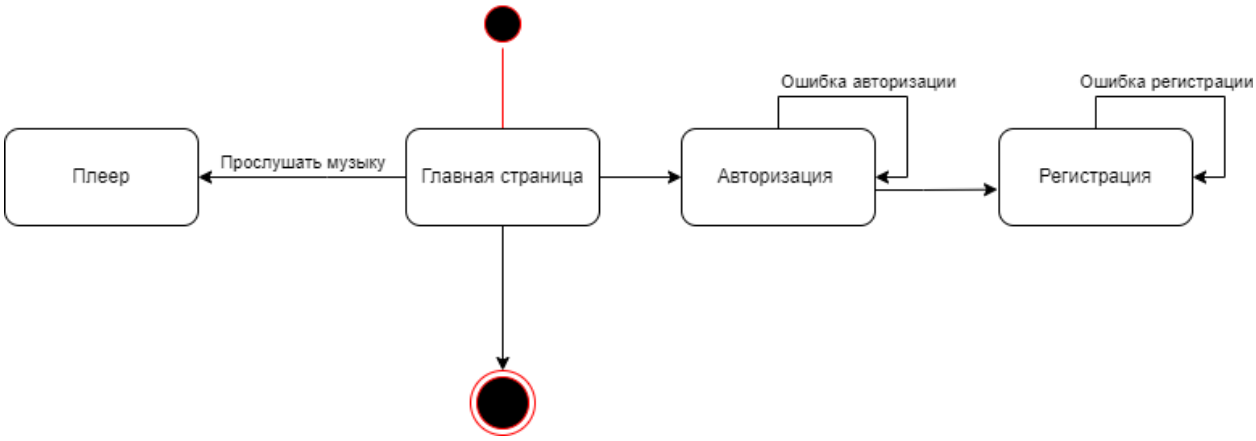


Рисунок 12 - Диаграмма состояния авторизованного пользователя

2.3.6 Диаграмма объектов

Диаграмма объектов позволяет определить классы объектов, их атрибуты и методы, а также взаимодействие между ними. Она помогает разработчикам лучше понимать структуру системы и проектировать ее более эффективно. Данная диаграмма представлена на рисунке 13.

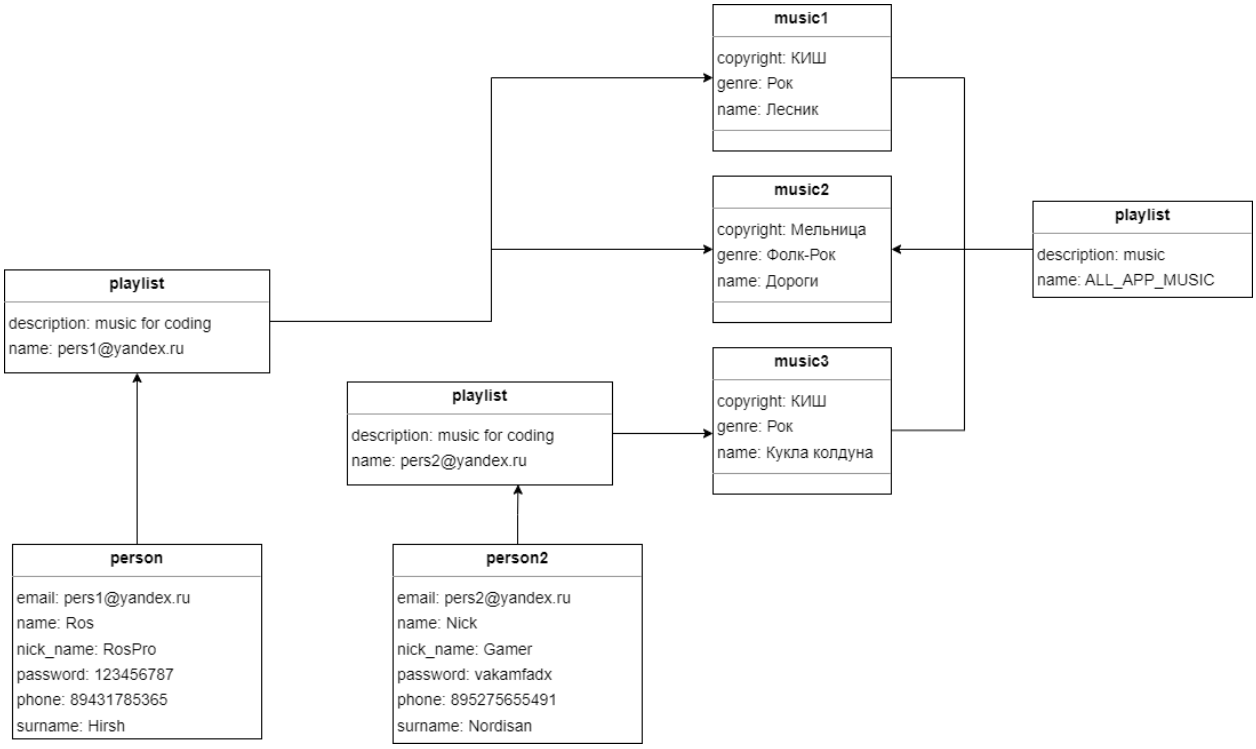


Рисунок 13 - Диаграмма объектов

2.3.7 Диаграммы активности

Диаграмма активности помогает разработчикам лучше понимать процессы в системе, выявлять узкие места и оптимизировать их. Она также может использоваться для описания бизнес-процессов и управления

проектами. Для данного проекта были спроектированы 3 диаграммы активности для администратора, зарегистрированного пользователя и незарегистрированного пользователя. Данные диаграммы представлены на рисунках 14-16.

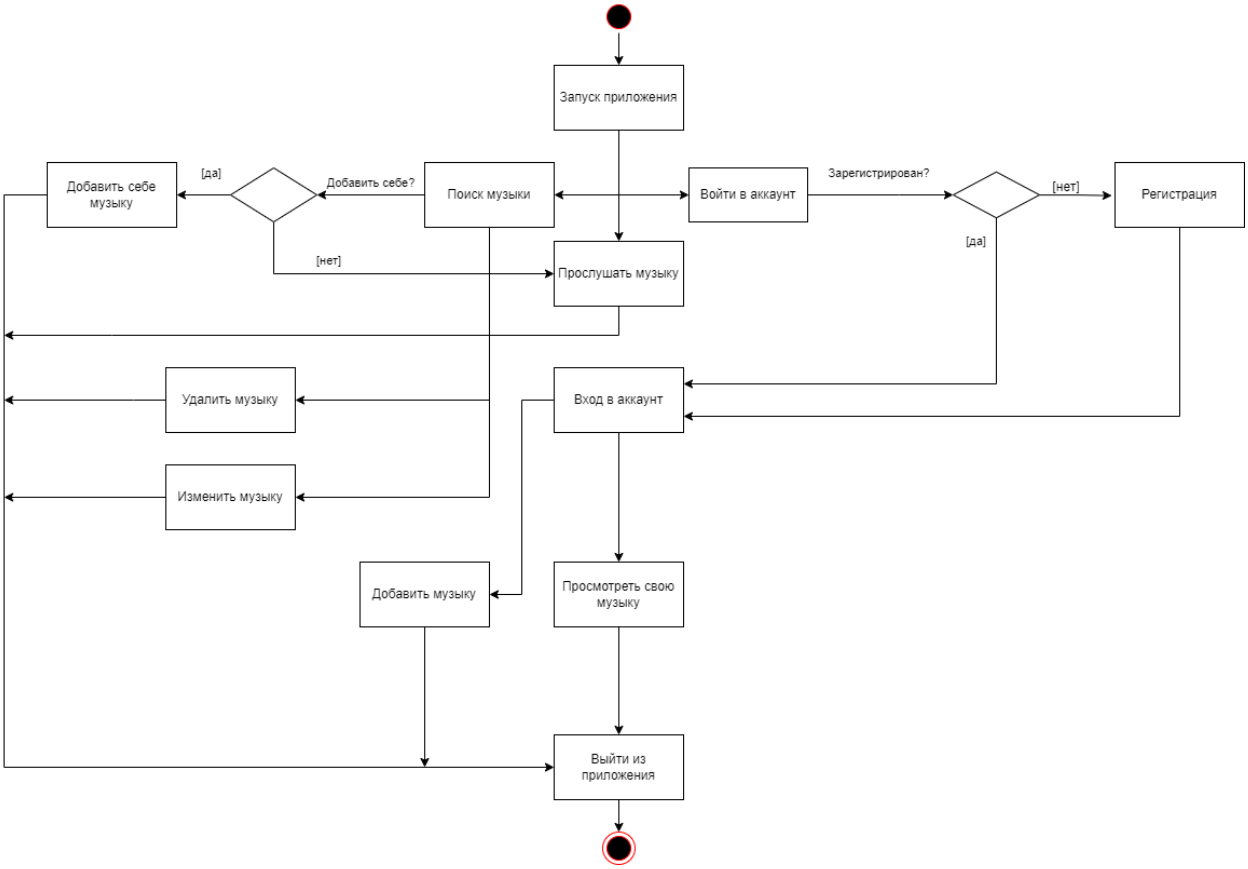


Рисунок 14 - Диаграмма активности администратора

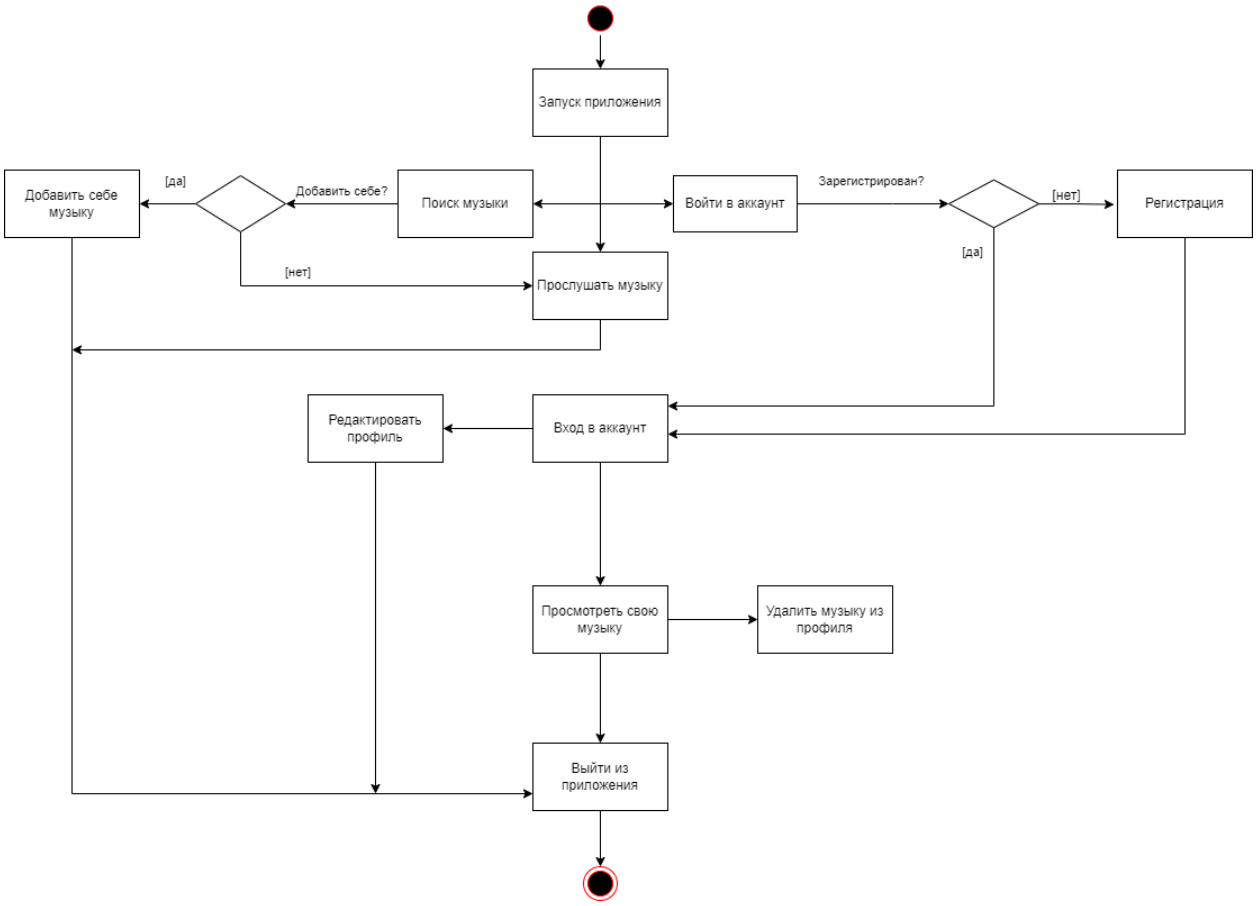


Рисунок 15 - Диаграмма активности авторизованного пользователя

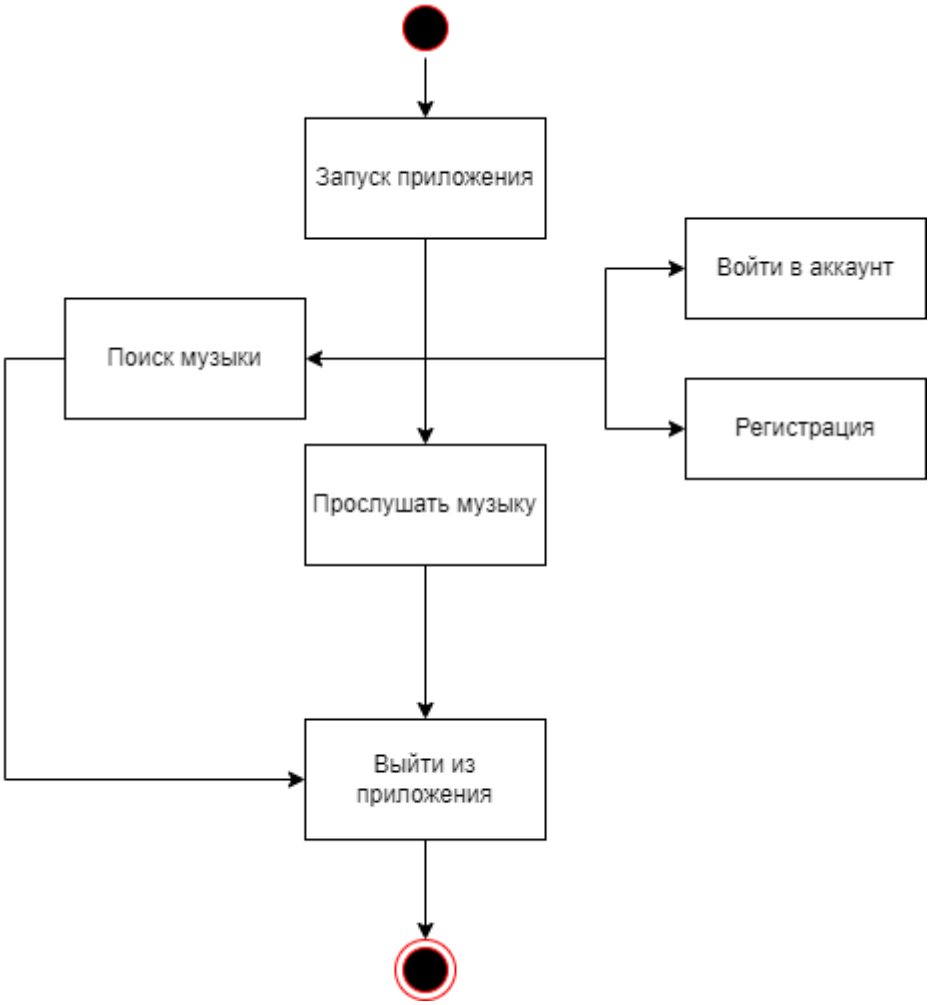


Рисунок 16 - Диаграмма активности неавторизованного пользователя

2.3.8 Диаграмма сотрудничества

Основная цель диаграммы сотрудничества состоит в том, чтобы визуализировать и описать взаимодействие объектов в рамках системы или компонента. Она позволяет понять, как объекты обмениваются информацией, вызывают методы друг у друга и работают вместе для выполнения задач. Данные диаграммы представлены на рисунках 17-25.

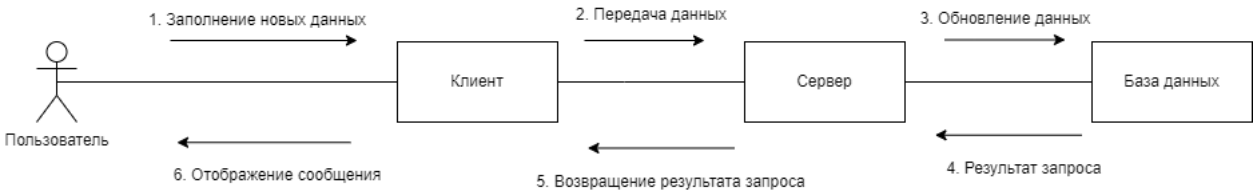


Рисунок 17 - Диаграмма сотрудничества обновления трека

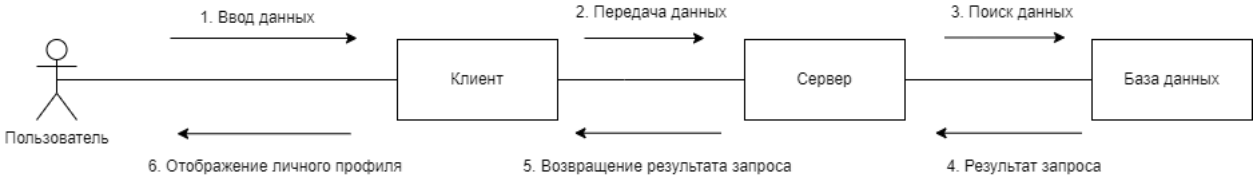


Рисунок 18 - Диаграмма сотрудничества авторизации

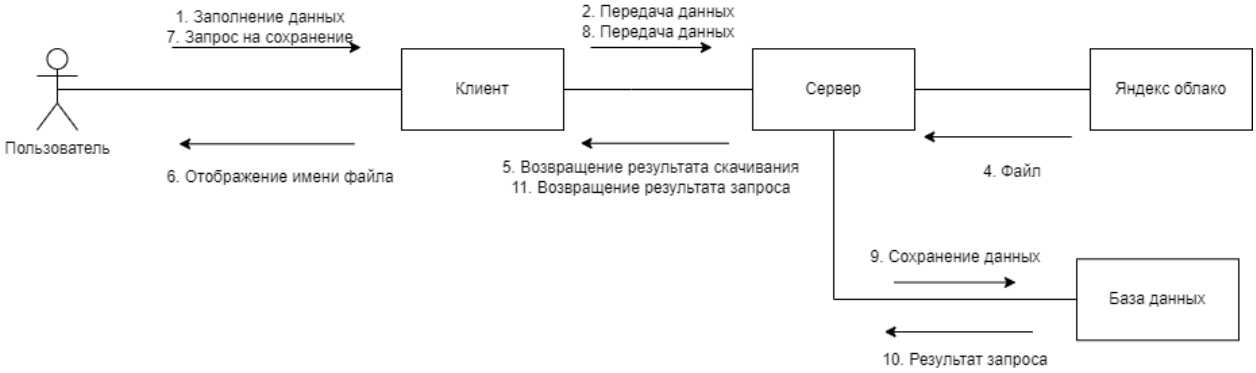


Рисунок 19 - Диаграмма сотрудничества загрузки трека

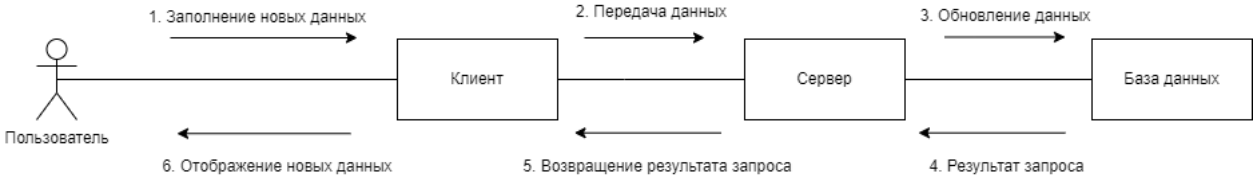


Рисунок 20 - Диаграмма сотрудничества обновления аккаунта

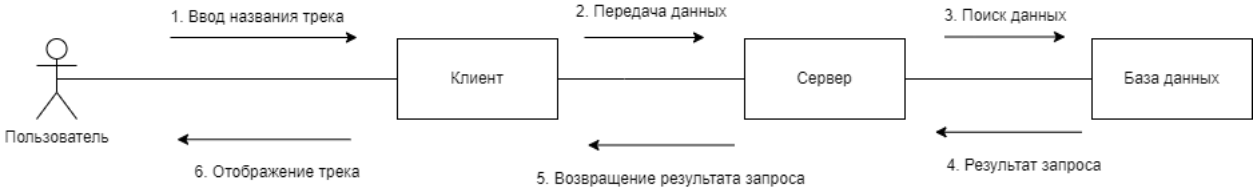


Рисунок 21 - Диаграмма сотрудничества поиска трека

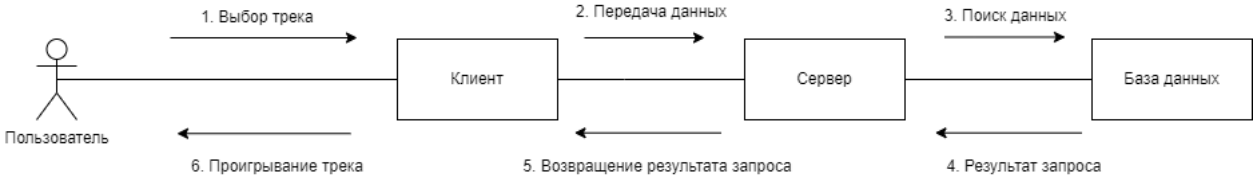


Рисунок 22 - Диаграмма сотрудничества прослушивания трека

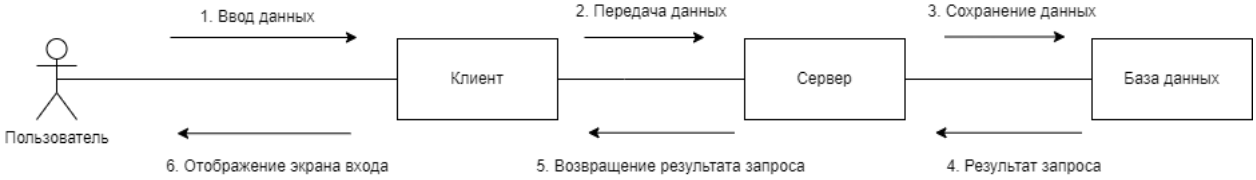


Рисунок 23 - Диаграмма сотрудничества регистрации

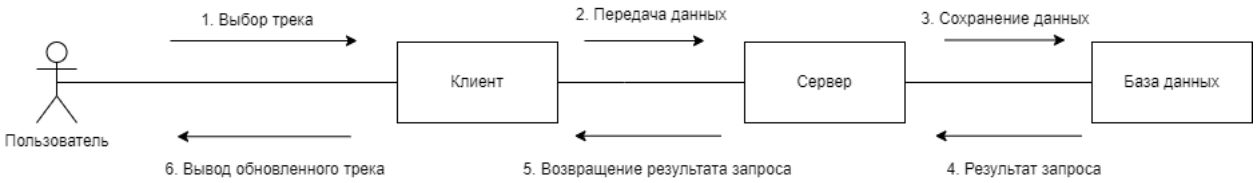


Рисунок 24 - Диаграмма сотрудичества сохранения трека

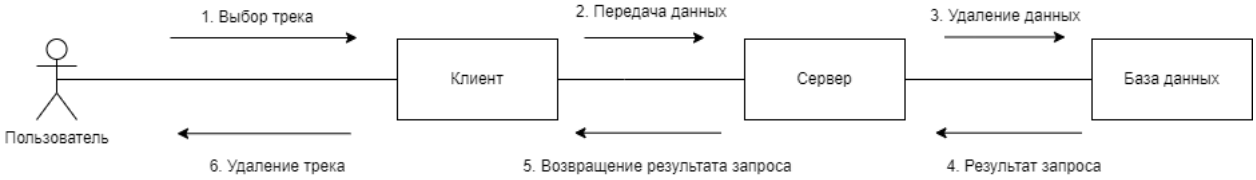


Рисунок 25 - Диаграмма сотрудничества удаления трека

3 Реализация

3.1 Средства реализации

Ниже приведен перечень используемых технологий.

Backend

— Java;

— SpringBootFramework;

— PostgreSQL;

— FlyWay;

— Docker.

Frontend:

— Android SDK;

— Java.

Инструменты для ведения документации:

— Miro;

— Swagger;

— Draw.io;

— Ramus;

— Figma.

Дополнительный инструментарий:

— Git;

— GitHub;

— Trello.

3.2 Реализация базы данных

Для хранения данных была выбрана база данных PostgreSQL. Она является продуктом с открытым исходным кодом, который поддерживается многими серверами. PostgreSQL поддерживает множественные типы данных, такие как числа разной точности, тексты с различными кодировками, изображения, звуки, видео, XML-документы, JSON-объекты и многие другие.

3.2.1 ER-диаграмма

ER-диаграмма используется для описания структуры базы данных, проектирования новых баз данных и анализа существующих баз данных. Она позволяет разработчикам и аналитикам лучше понять логические связи и зависимости между различными сущностями и их атрибутами. Данная диаграмма представлена на рисунке 26.

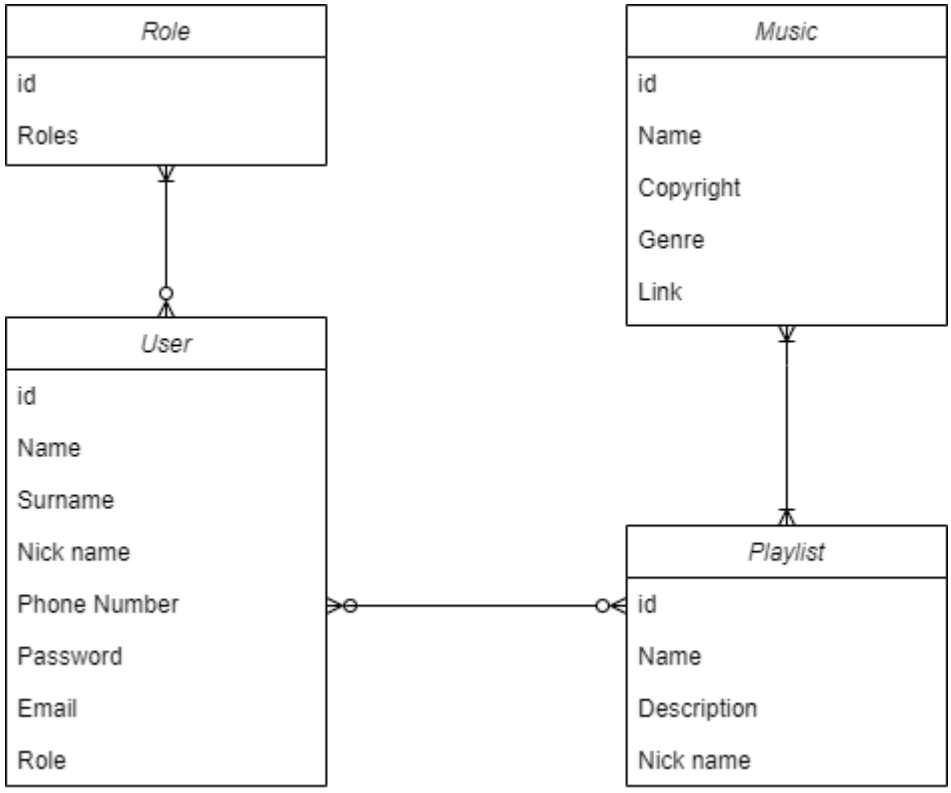


Рисунок 26 - ER-диаграмма базы данных

3.2.2 Физическая модель базы данных

Физическая модель базы данных используется для описания способа хранения и организации данных в реальной системе базы данных. Она определяет конкретные технические детали, такие как типы данных, таблицы, связи и другие аспекты, связанные с физической реализацией базы данных. Данная диаграмма представлена на рисунке 27.

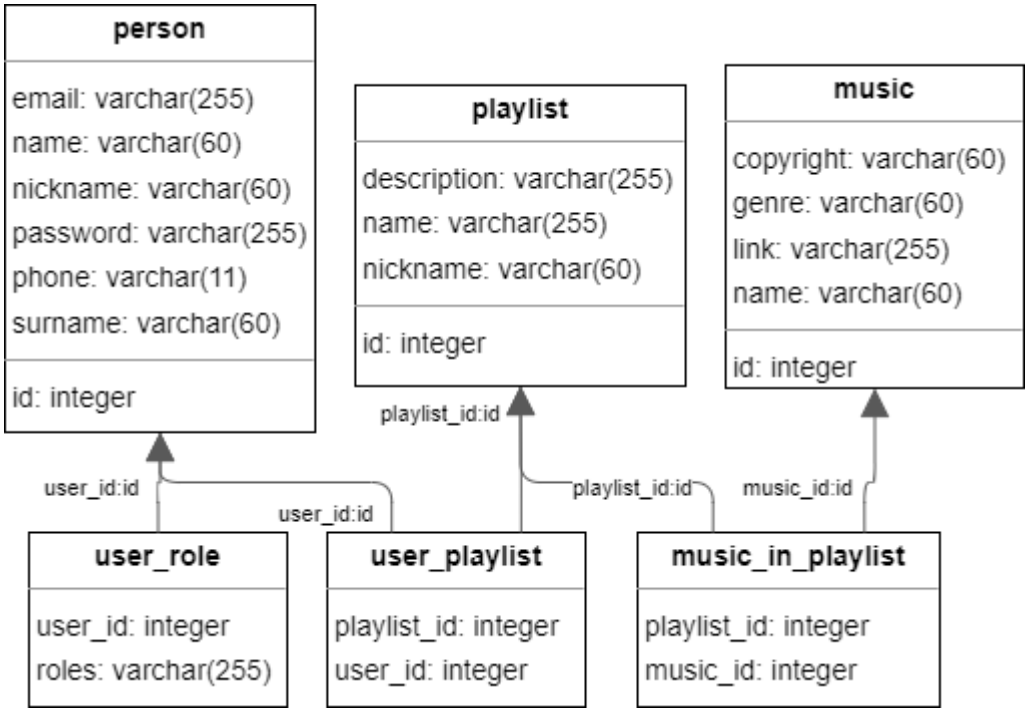


Рисунок 27 - Физическая модель базы данных

3.3 Реализация клиентской части

Для реализации клиентской части приложения было выбрано сочетание Android SDK и языка Java. Этот набор обеспечивает разработчикам множество возможностей для создания мобильных приложений под Android. Они могут использовать SDK для доступа к аппаратным возможностям устройства, взаимодействия с различными сервисами и API, разработки пользовательского интерфейса и многого другого. Java, в свою очередь, предоставляет надежный и мощный язык программирования, который позволяет разработчикам создавать сложные и высокопроизводительные приложения.

Клиентская часть имеет архитектуру, основанную на модели MVC. Контроллеры обрабатывают входящие запросы и отправляют ответы клиенту. Они являются интерфейсом между клиентом и бизнес-логикой

приложения. Сервисы предоставляют бизнес-логику приложения. Они представлены в виде классов активностей и фрагментов приложения. Модели представлены отдельными классами для пользователей и музыки.

Все эти компоненты взаимодействуют друг с другом, чтобы обеспечить функциональность приложения. Контроллеры получают запросы от клиента, направляют их в сервисы, которые используют репозитории для доступа к данным в базе данных, а затем возвращают ответы в контроллеры, которые отправляют их клиенту.

3.3.1 Форма для поиска музыки

Пользователь (авторизованный и не авторизованный) имеет возможность найти музыку для прослушивания по названию, автору или жанру. В качестве ответа на его запрос будет выведен список музыки. Пользователь может нажать на трек, чтобы прослушать его.

Зарегистрированный пользователь имеет возможность добавить найденную музыку себе в профиль.

Администратор имеет возможность добавить найденную музыку себе в профиль, а также редактировать или удалить музыку.

Снизу расположена панель с кнопками, при нажатии на соответствующую кнопку пользователь переходит на страницы: поиск/главная, прослушивание музыки, личная страница. Форма данного экрана представлена на рисунке 28.

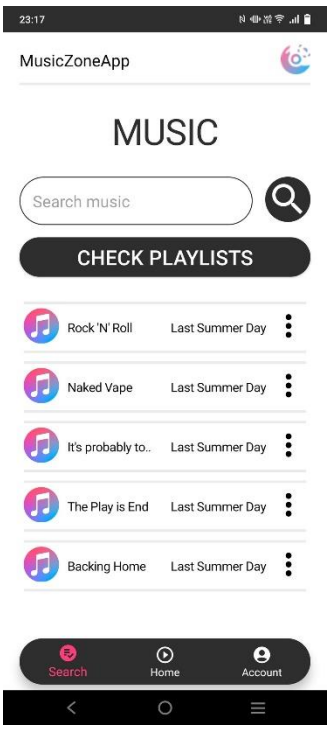


Рисунок 28 - Форма экрана поиска музыки

3.3.2 Форма экрана плеера

На данном экране расположена панель с названием трека и его описанием, слайдер, отображающий текущее время трека и кнопки:

— перемотка трека;

— стоп или играть;

— следующий трек.

Снизу расположена панель с кнопками, при нажатии на соответствующую кнопку пользователь переходит на страницы: поиск/главная, прослушивание музыки, личная страница. Форма данного экрана представлена на рисунке 29.

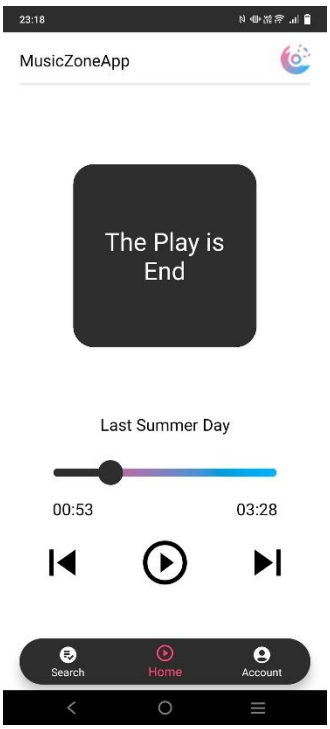


Рисунок 29 - Форма экрана плеера

3.3.3 Форма экрана авторизации

На данном экране отображены поля с вводом логина и пароля к аккаунту, кнопка войти в аккаунт, кнопка зарегистрироваться, а также кнопка восстановление пароля.

Снизу расположена панель с кнопками, при нажатии на соответствующую кнопку пользователь переходит на страницы: поиск/главная, прослушивание музыки, личная страница. Форма данного экрана представлена на рисунке 30.

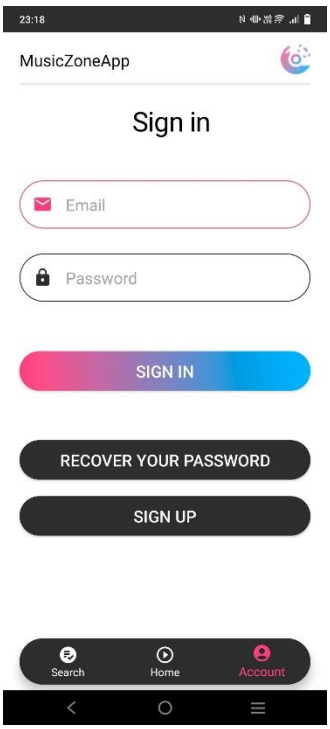


Рисунок 30 - Форма экрана авторизации

3.3.4 Форма экрана регистрации

На данном экране отображены поля для ввода имени, фамилии, псевдонима, почты, номера телефона, пароля. Ниже расположена кнопка регистрации, также есть кнопка для выхода из данной формы. Форма данного экрана представлена на рисунке 31.

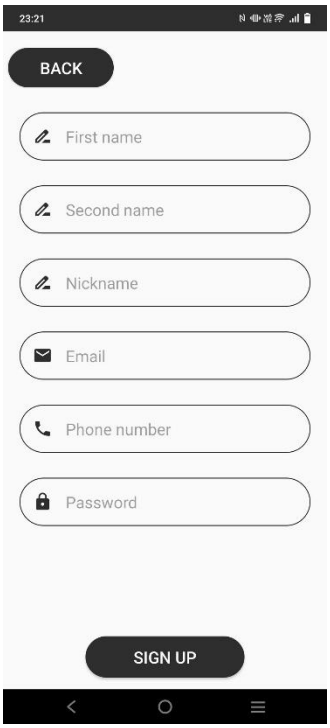


Рисунок 31 - Форма экрана регистрации

3.3.5 Форма экрана восстановления пароля

На данном экране отображены поля для ввода адреса электронной почты и кнопка отправки кода. Также есть форма для ввода полученного кода, и кнопка для подтверждения кода. При успешной проверке кода, появляется форма для заполнения нового пароля, и кнопка подтверждения нового пароля. Формы данных экранов представлены на рисунках 32-33.

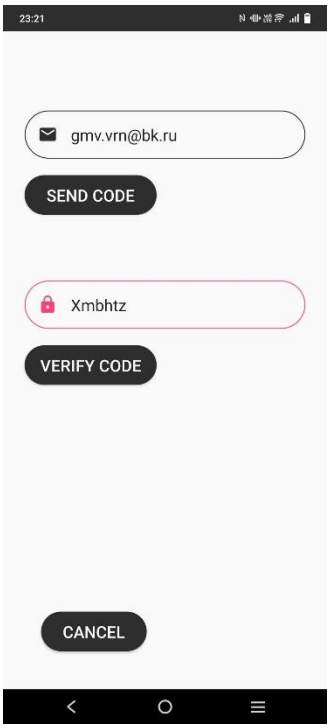


Рисунок 32 - Форма экрана восстановления пароля

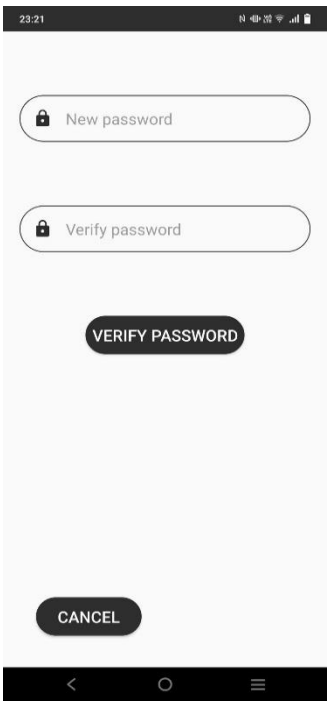


Рисунок 33 - Форма экрана ввода нового пароля

3.3.6 Форма экрана личной страницы

На данном экране указана информация о пользователе (имя, фамилия, псевдоним, почта и номер телефона), рядом расположена кнопки редактирования пользователя и выхода из аккаунта. Ниже находится кнопка для просмотра добавленной музыки.

Снизу расположена панель с кнопками, при нажатии на соответствующую кнопку пользователь переходит на страницы: поиск(главная), прослушивание музыки, личная страница.

Если пользователь является администратором, то на экране появляется кнопка загрузки музыки. Формы данных экранов представлены на рисунках 34-35.

Рисунок 34 - Форма экрана личной страницы

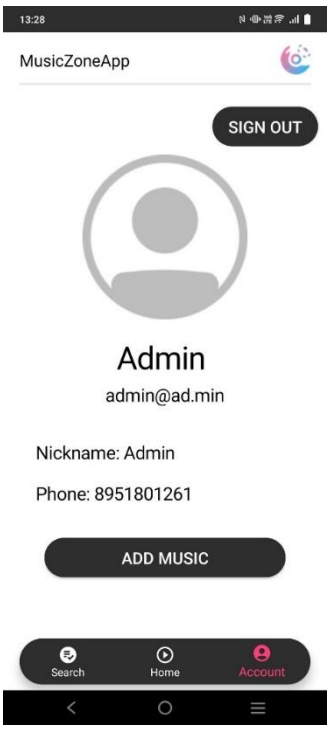


Рисунок 35 - Форма экрана личной страницы администратора

3.3.7 Форма экрана редактирования профиля

На данном экране отображены поля для ввода имени, фамилии, псевдонима, почты, номера телефона с уже введенными соответствующими данными. Ниже расположена кнопка для сохранения данных. Также есть кнопка для выхода из данного экрана. Форма данного экрана представлена на рисунке 36.

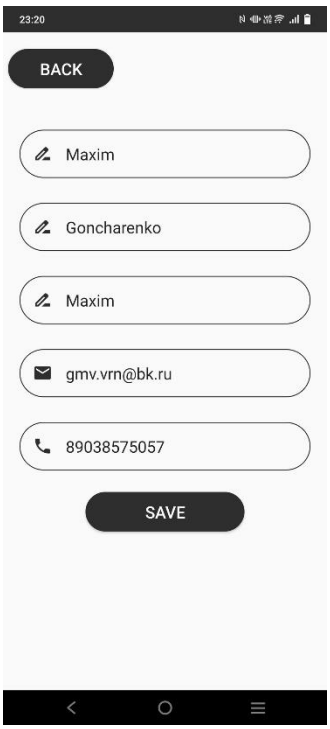


Рисунок 36 - Форма экрана редактирования профиля

3.3.8 Форма экрана загруженной музыки

На данном экране расположены список добавленной музыки со стороны администратора приложения, и кнопка для добавления музыки.

Снизу расположена панель с кнопками, при нажатии на соответствующую кнопку пользователь переходит на страницы: поиск/главная, прослушивание музыки, личная страница. Форма данного экрана представлена на рисунке 37.

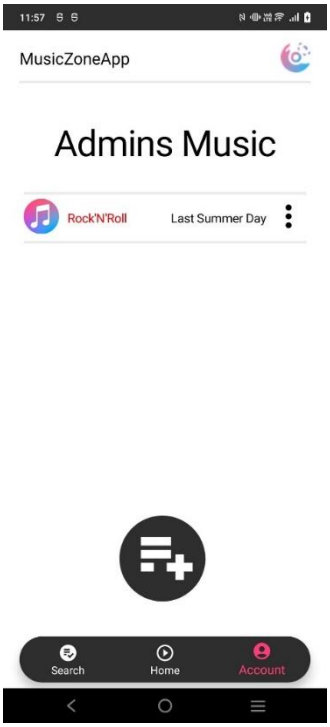


Рисунок 37 - Форма экрана загрузки музыки

3.3.9 Макет экрана добавленной музыки

На этой странице отображается добавленная музыка пользователем. Снизу расположена панель с кнопками, при нажатии на соответствующую кнопку пользователь переходит на страницы: поиск/главная, прослушивание музыки, личная страница. Форма данного экрана представлена на рисунке 38.

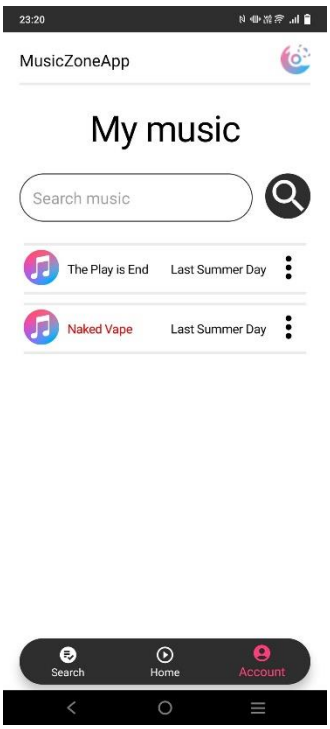


Рисунок 38 - Форма экрана пользовательской музыки

3.3.10 Форма экрана добавления музыки

На данном экране расположены поля для заполнения названия музыки, имя автора, жанра музыки, названия трека, кнопки для загрузки музыкального файла, сохранения и выхода. Форма данного экрана представлена на рисунке 39.

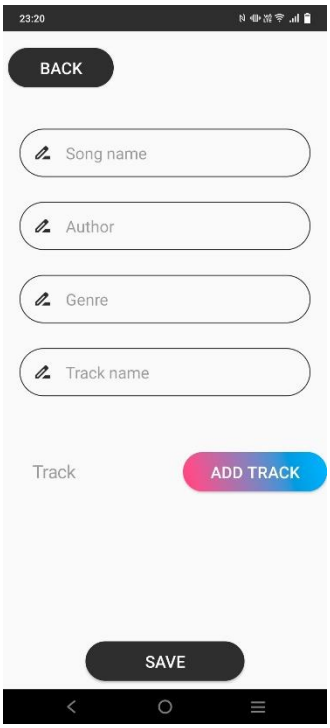


Рисунок 39 - Форма экрана добавления музыки

3.3.11 Форма экрана редактирования музыки

На экране расположены поля уже заполненными соответствующими данными для редактирования (название музыки, автор, жанр музыки) и загруженный музыкальный файл. В нижней части находятся кнопки для сохранения, загрузки музыкального файла и выхода. Форма данного экрана представлена на рисунке 40.

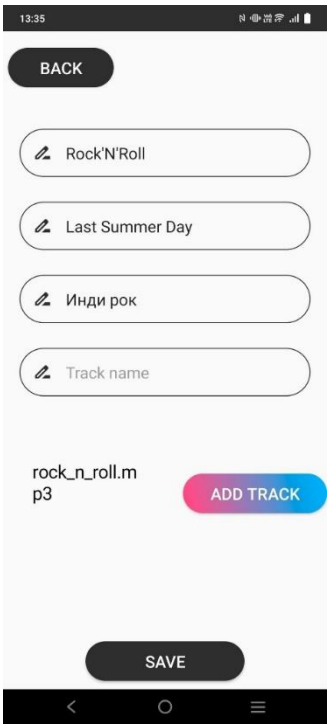


Рисунок 40 - Макет экрана редактирования музыки

3.4 Серверная часть

В качестве языка был выбран строго типизированный объектно-ориентированный язык программирования Java. Он остается очень популярным языком программирования в этой мобильной разработке благодаря своим мощным возможностям и широкому спектру инструментов для разработки. К тому же существует огромное количество фреймворков и библиотек, написанных на Java, которые в перспективе можно легко интегрировать в проект.

Также применяется программная платформа Docker для тестирования и развертывания приложений.

3.4.1 Архитектура серверной части приложения

Серверная часть приложения реализована соответственно трехслойной архитектуре веб-приложения с API Rest с использованием фреймворка Spring boot. Данный фреймворк предоставляет возможности для работы с базами данных, а также механизм внедрения зависимостей, который позволяет сделать компоненты программы слабосвязанными, а всю программу в целом более гибкой, адаптируемой и расширяемой.

3.4.2 Диаграмма классов контроллеров

Контроллеры в Spring Boot отвечают за обработку входящих HTTP-запросов и управление потоком выполнения приложения. Данные классы образуют так называемы первый микросервисный слой. Контроллеры определяют различные методы для обработки запросов. В нашем случае они представлены классами на рисунке 41.

Рисунок 41 - Диаграмма классов контроллеров

3.4.3 Диаграмма классов сервисов

Сервисы в Spring Boot представляют второй слой бизнес-логики приложения. Они содержат бизнес-логику и выполняют операции, которые требуются для обработки запросов контроллеров. Сервисы могут использовать другие компоненты, такие как репозитории, для доступа к данным. Классы сервисы представлены на рисунке 42.

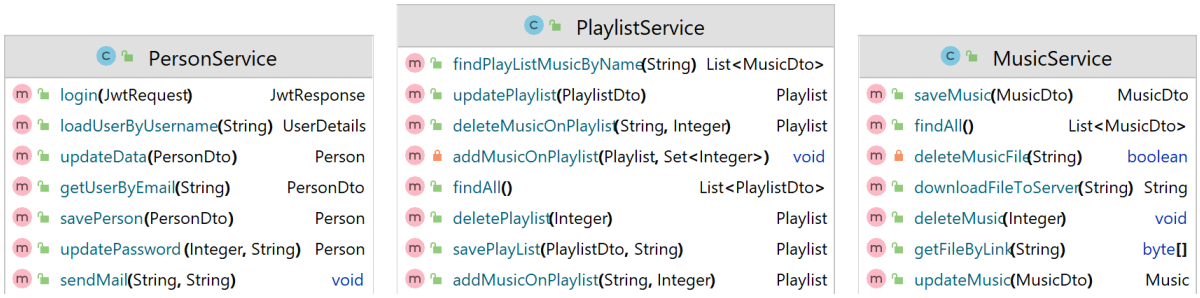


Рисунок 42 - Диаграмма классов сервисов

3.4.4 Диаграмма классов репозиториев

Репозитории в Spring Boot обеспечивают доступ к данным, таким как базы данных или внешние сервисы. Данные классы образуют третий микросервисный слой. Они используются для выполнения операций чтения и записи данных. Репозитории используют специфические аннотации, такие как query, для определения пользовательских запросов к базе данных. Классы репозиториев представлены на рисунке 43.

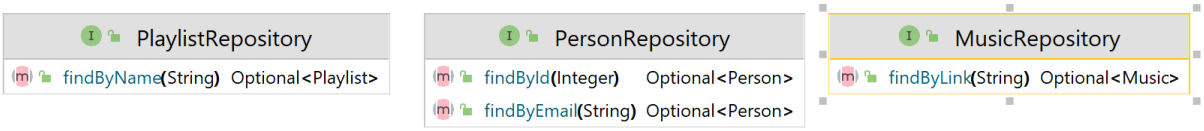


Рисунок 43 - Диаграмма классов репозиториев

3.4.5 Диаграмма классов моделей

Модели представляют данные, с которыми работает приложение. Они являются простыми Java-классами, содержащими поля, геттеры и сеттеры для доступа к данным. Эти классы образуют последний микросервисный слой. Модели используются для передачи данных между контроллерами, сервисами и репозиториями. Классы моделей представлены на рисунке 44.

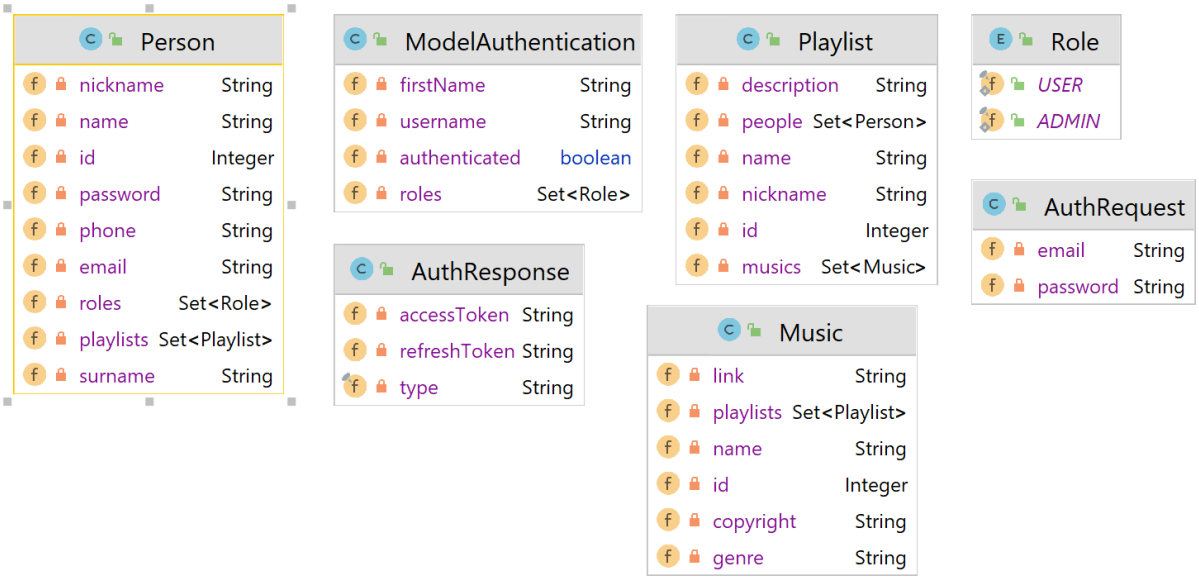


Рисунок 44 - Диаграмма классов моделей

3.4.6 Диаграмма классов DTO

DTO представляют объекты, используемые для передачи данных между слоями приложения или между приложением и клиентом. Они содержат только необходимые данные и не содержат бизнес-логики. В нашем случае DTO используются в RESTful API для передачи данных между клиентом и сервером. Диаграмма классов DTO представлена на рисунке 45.

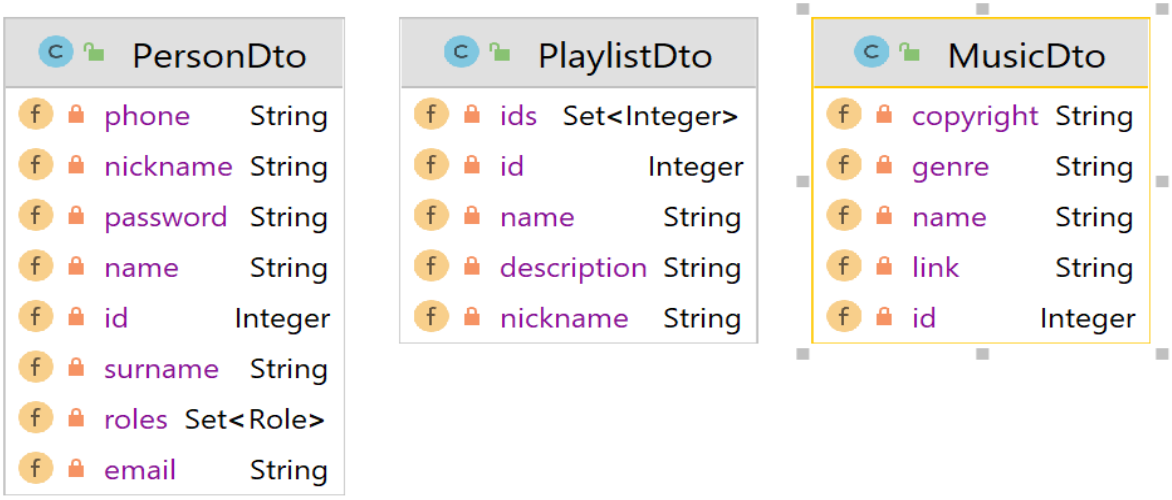


Рисунок 45 - Диаграмма классов DTO

3.4.7 Диаграмма классов исключений

Исключения в Java используются для обработки ошибок и необычных ситуаций, которые могут возникнуть во время выполнения программы. Они позволяют предусмотреть возможные проблемы и определить, как программа должна реагировать на них. Например, если программа пытается найти пользователя из базы данных которого не существует, может быть сгенерировано исключение, и для обработки этой ситуации вынесется сообщения об ошибке. Классы исключений представлены на рисунке 46.

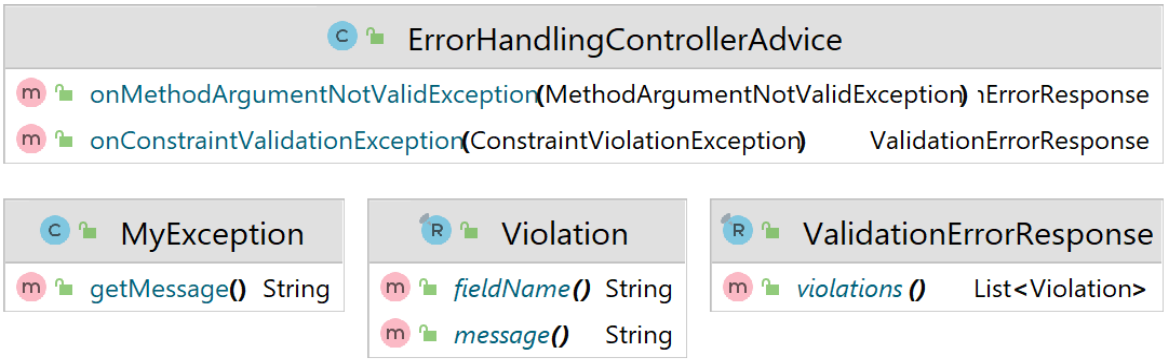


Рисунок 46 - Диаграмма классов исключений

3.4.8 Диаграмма классов конфигураций

Конфигурации в Spring Boot позволяют настраивать ваше приложение. Данные классы конфигурации, определяют бины и другие настройки приложения. Конфигурации используют аннотации bean, value, autowired, для настройки зависимостей и значений свойств. Классы конфигураций представлены на рисунке 47.

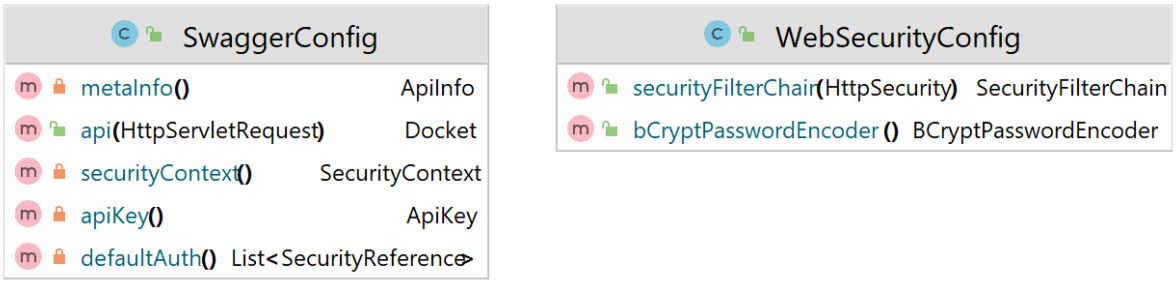


Рисунок 47 - Диаграмма классов конфигураций

3.4.9 Диаграмма классов сериализаторов

Классы сериализаторы, или в нашем случае мапперы, используются для преобразования данных между различными типами или моделями. Сериализация применяется для представления модели основного объекта в DTO. В Spring Boot для этого используется библиотека MapStruct, которая предоставляет функционал для генерации кода сериализации и десереализации. Данные классы представлены на рисунке 48.

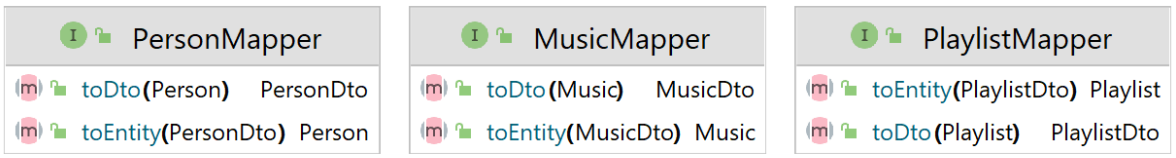


Рисунок 48 - Диаграмма классов сериализаторов

3.4.10 Диаграмма классов JWT

JWT токены используются для авторизации доступа к ресурсам веб-приложения. В токене можно включить информацию о ролях или разрешениях пользователя. Сервер при получении запроса сервер проверяет токен, чтобы определить, имеет ли пользователь необходимые разрешения

для доступа к запрашиваемым ресурсам. Классы, реализующие данный функционал представлены на рисунке 49.

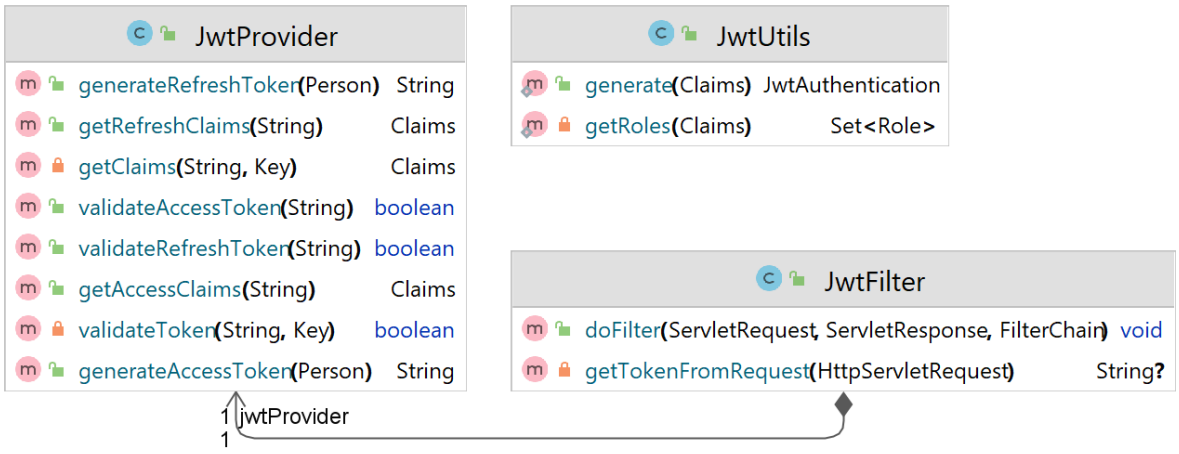


Рисунок 49 - Диаграмма классов JWT

4 Тестирование

4.1 UI тестирование

UI тестирование (User Interface testing) - это процесс проверки пользовательского интерфейса программного обеспечения на соответствие требованиям и ожиданиям пользователей. Оно включает в себя проверку внешнего вида, удобства использования, доступности и функциональности интерфейса.

UI тестирование необходимо для обеспечения высокого качества программного обеспечения и удовлетворения потребностей пользователей. Оно помогает выявлять ошибки и недочеты в интерфейсе, которые могут привести к негативному опыту использования приложения. Тестирование также позволяет оптимизировать интерфейс и улучшить его функциональность, что повышает удобство использования и привлекательность приложения для пользователей. Результаты UI тестирования данного приложения представлено на рисунках 50-59.

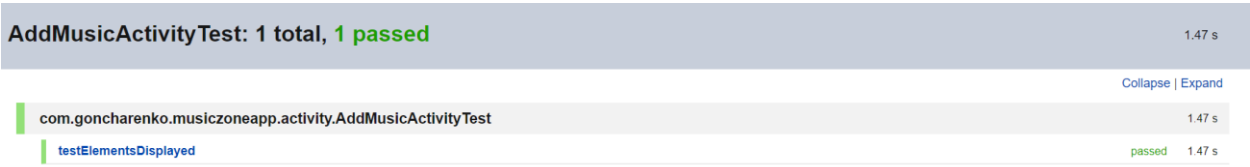


Рисунок 50 - Результат тестирования формы для добавления музыки

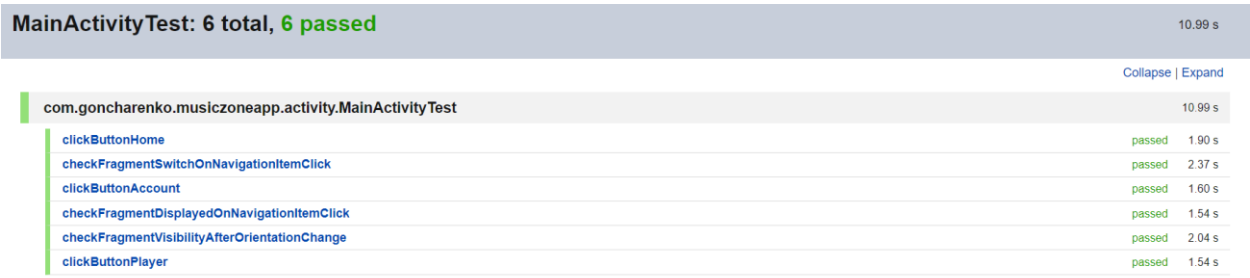


Рисунок 51 - Результат тестирования главной активности

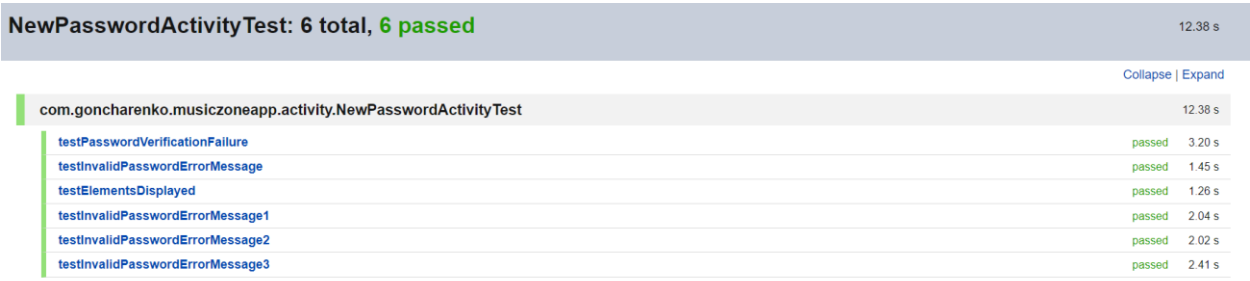
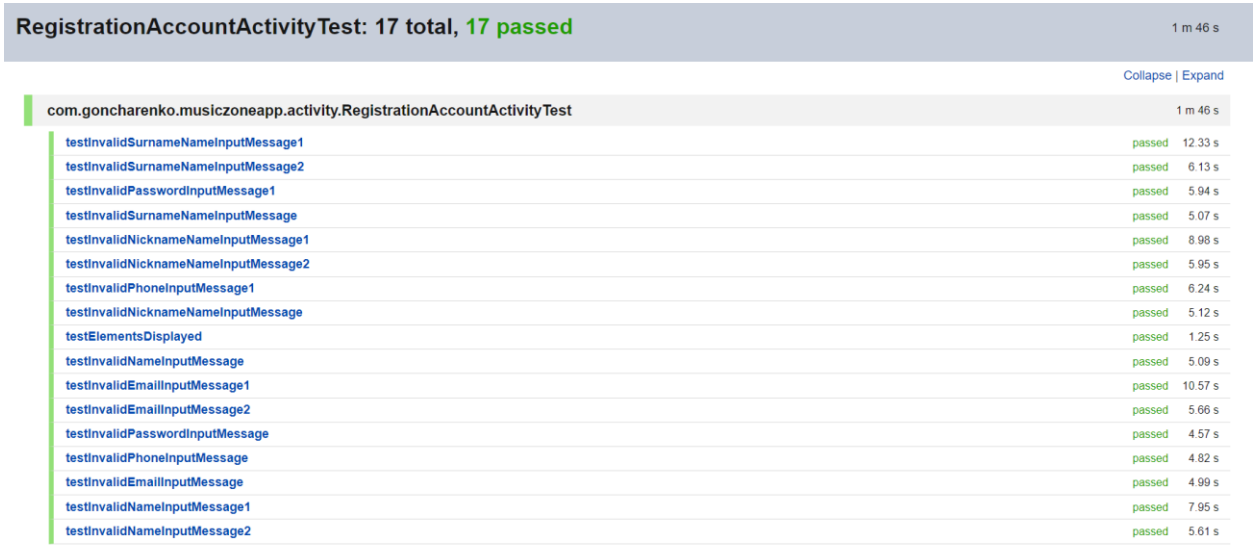


Рисунок 52 - Результат тестирования формы для изменения пароля



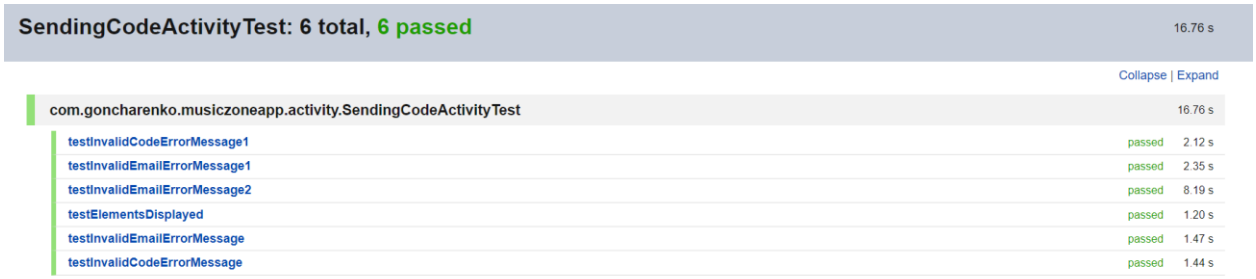
RegistrationAccountActivityTest: 17 total, 17 passed — 1 m 46 s

Collapse | Expand

com.goncharenko.musiczoneapp.activity.RegistrationAccountActivityTest — 1 m 46 s

testInvalidSurnameNameInputMessage1	passed	12.33 s
testInvalidSurnameNameInputMessage2	passed	6.13 s
testInvalidPasswordInputMessage1	passed	5.94 s
testInvalidSurnameNameInputMessage	passed	5.07 s
testInvalidNicknameNameInputMessage1	passed	8.98 s
testInvalidNicknameNameInputMessage2	passed	5.95 s
testInvalidPhoneInputMessage1	passed	6.24 s
testInvalidNicknameNameInputMessage	passed	5.12 s
testElementsDisplayed	passed	1.25 s
testInvalidNameInputMessage	passed	5.09 s
testInvalidEmailInputMessage1	passed	10.57 s
testInvalidEmailInputMessage2	passed	5.66 s
testInvalidPasswordInputMessage	passed	4.57 s
testInvalidPhoneInputMessage	passed	4.82 s
testInvalidEmailInputMessage	passed	4.99 s
testInvalidNameInputMessage1	passed	7.95 s
testInvalidNameInputMessage2	passed	5.61 s

Рисунок 53 - Результат тестирования формы регистрации пользователя



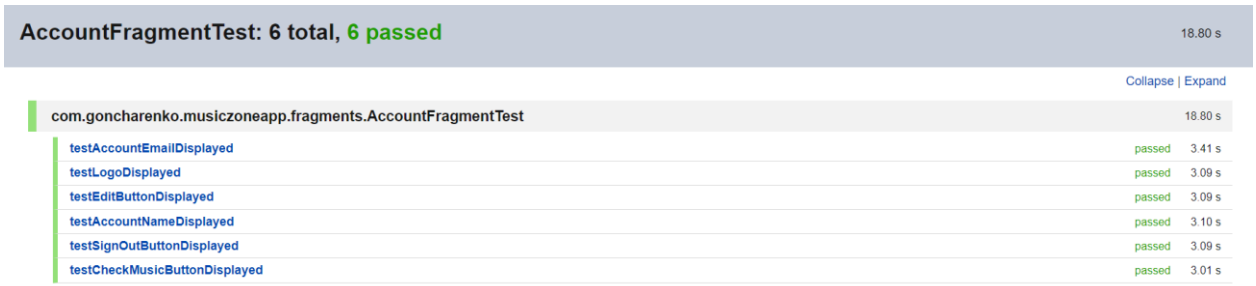
SendingCodeActivityTest: 6 total, 6 passed — 16.76 s

Collapse | Expand

com.goncharenko.musiczoneapp.activity.SendingCodeActivityTest — 16.76 s

testInvalidCodeErrorMessage1	passed	2.12 s
testInvalidEmailErrorMessage1	passed	2.35 s
testInvalidEmailErrorMessage2	passed	8.19 s
testElementsDisplayed	passed	1.20 s
testInvalidEmailErrorMessage	passed	1.47 s
testInvalidCodeErrorMessage	passed	1.44 s

Рисунок 54 - Результат тестирования формы для отправки кода на почту



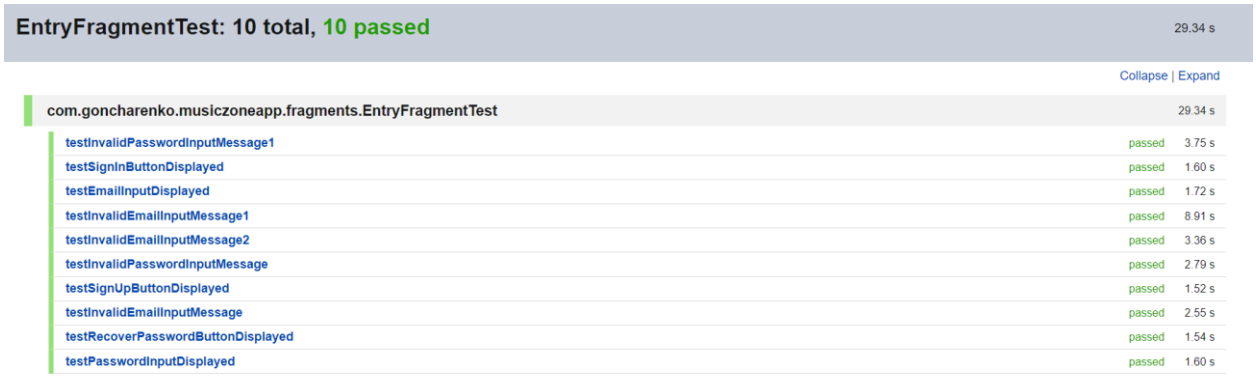
AccountFragmentTest: 6 total, 6 passed — 18.80 s

Collapse | Expand

com.goncharenko.musiczoneapp.fragments.AccountFragmentTest — 18.80 s

testAccountEmailDisplayed	passed	3.41 s
testLogoDisplayed	passed	3.09 s
testEditButtonDisplayed	passed	3.09 s
testAccountNameDisplayed	passed	3.10 s
testSignOutButtonDisplayed	passed	3.09 s
testCheckMusicButtonDisplayed	passed	3.01 s

Рисунок 55 - Результат тестирования формы личной страницы



EntryFragmentTest: 10 total, 10 passed — 29.34 s

Collapse | Expand

com.goncharenko.musiczoneapp.fragments.EntryFragmentTest — 29.34 s

testInvalidPasswordInputMessage1	passed	3.75 s
testSignInButtonDisplayed	passed	1.60 s
testEmailInputDisplayed	passed	1.72 s
testInvalidEmailInputMessage1	passed	8.91 s
testInvalidEmailInputMessage2	passed	3.36 s
testInvalidPasswordInputMessage	passed	2.79 s
testSignUpButtonDisplayed	passed	1.52 s
testInvalidEmailInputMessage	passed	2.55 s
testRecoverPasswordButtonDisplayed	passed	1.54 s
testPasswordInputDisplayed	passed	1.60 s

Рисунок 56 - Результат тестирования формы для входа в личную страницу

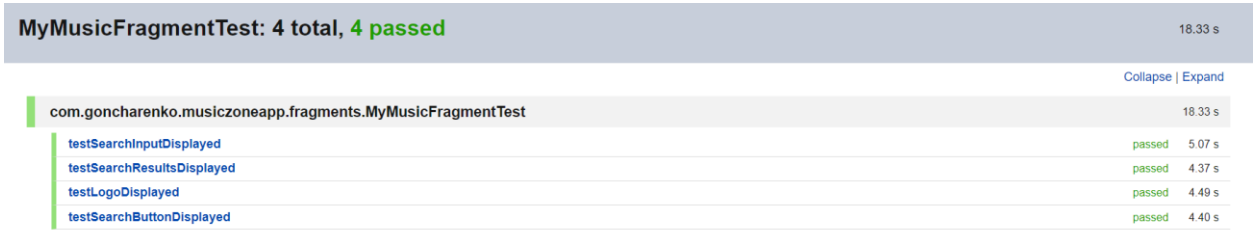


Рисунок 57 - Результат тестирования формы личной музыки пользователя

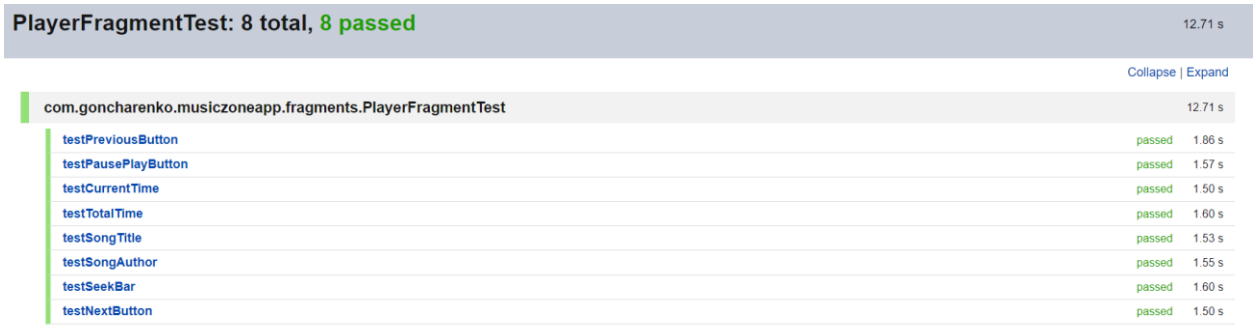


Рисунок 58 - Результат тестирования формы плеера

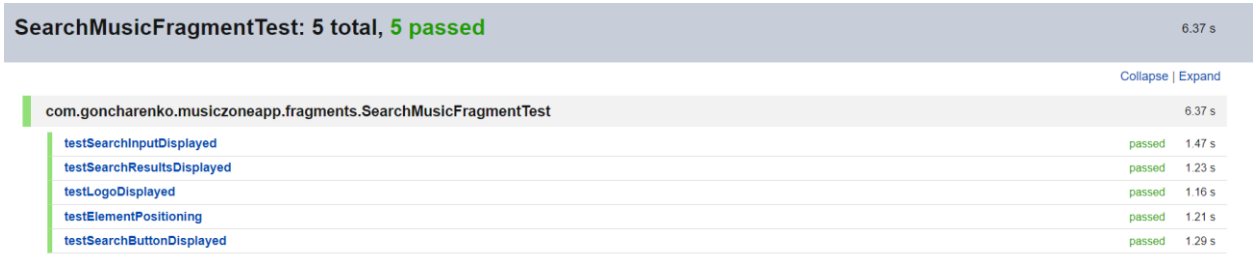


Рисунок 59 - Результат тестирования формы для поиска музыки

4.2 Интеграционное тестирование

Интеграционные тесты направлены на проверку правильности передачи данных, согласованности интерфейсов, взаимодействия компонентов и обнаружения возможных проблем, возникающих при интеграции различных модулей. Они помогают выявить ошибки, которые могут возникнуть при взаимодействии компонентов, такие как неправильная передача данных, некорректное обращение к интерфейсам, проблемы совместной работы разных модулей и другие аномалии. Результаты интеграционного тестирования представлены на рисунке 60-62

Рисунок 60 - Результат тестирования бизнес логики для пользователя

Рисунок 61 - Результат тестирования бизнес логики для музыки

Рисунок 62 - Результат тестирования бизнес логики плейлистов

5 Аналитика

Для сбора метрик для приложения было выбрано AppMetrica от Яндекс. Это система для сбора данных об использовании приложения пользователями. Она отличается высокой скоростью и удобством настройки метрик для мобильных приложений. Кроме того, она предлагает интуитивный интерфейс и понятное руководство по использованию.

Были составлены четыре воронки конверсии:

— общая статистика;

— музыка;

— аккаунт;

— музыка пользователя.

Воронка «Общая статистика» нужна для просмотра информации о количестве скачиваний приложения и количества запуска приложения. Изображение данной воронки на рисунке 60.

Рисунок 63 - Воронка «Общая статистика»

Воронка «Музыка» нужна для просмотра статистики по количеству прослушавания музыки, а также перемотки музыки и её пролистывания вперед. Изображение данной воронки на рисунке 61.

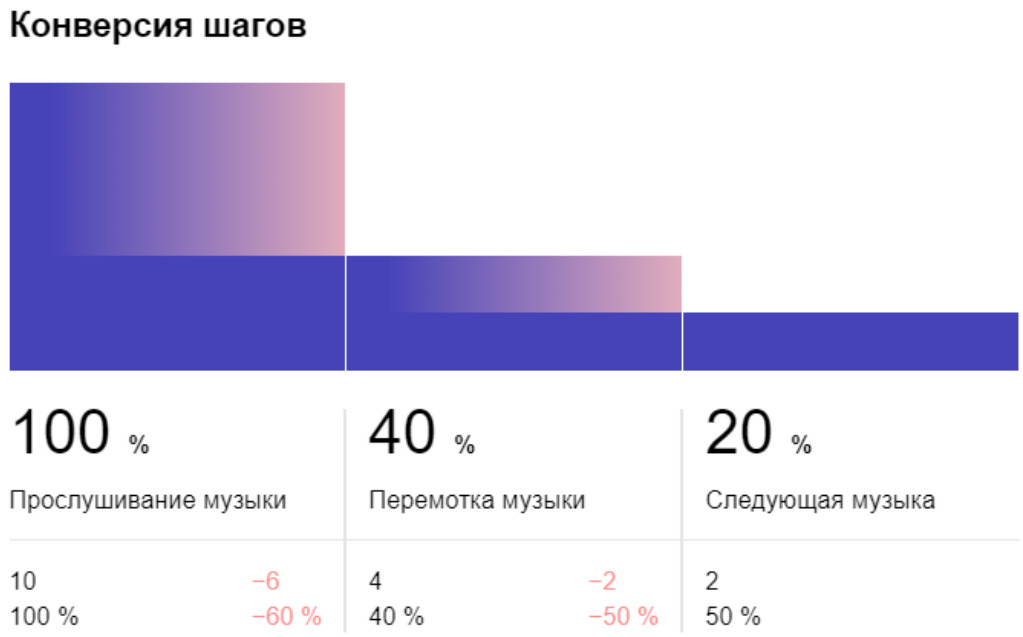


Рисунок 64 - Воронка «Музыка»

Воронка «Аккаунт» нужна для просмотра статистики по количеству входа в аккаунт и выхода из него, а также регистрации и изменении аккаунта. Изображение данной воронки на рисунке 62.

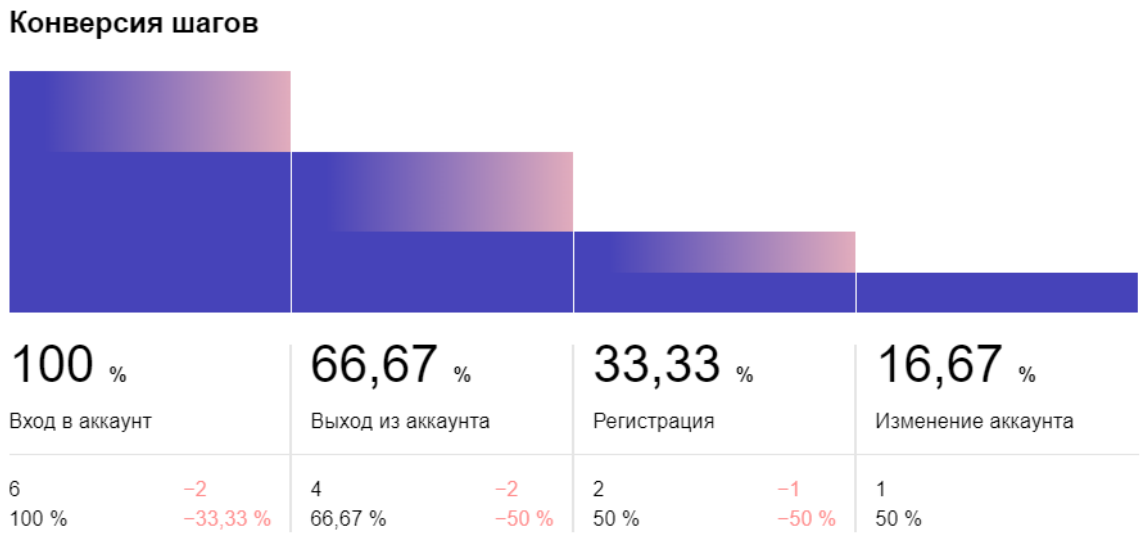


Рисунок 65 - Воронка «Аккаунт»

Воронка «Музыка пользователя» нужна для просмотра статистики по количеству добавления и удаления музыки со стороны обычных пользователей. Изображение данной воронки на рисунке 63.

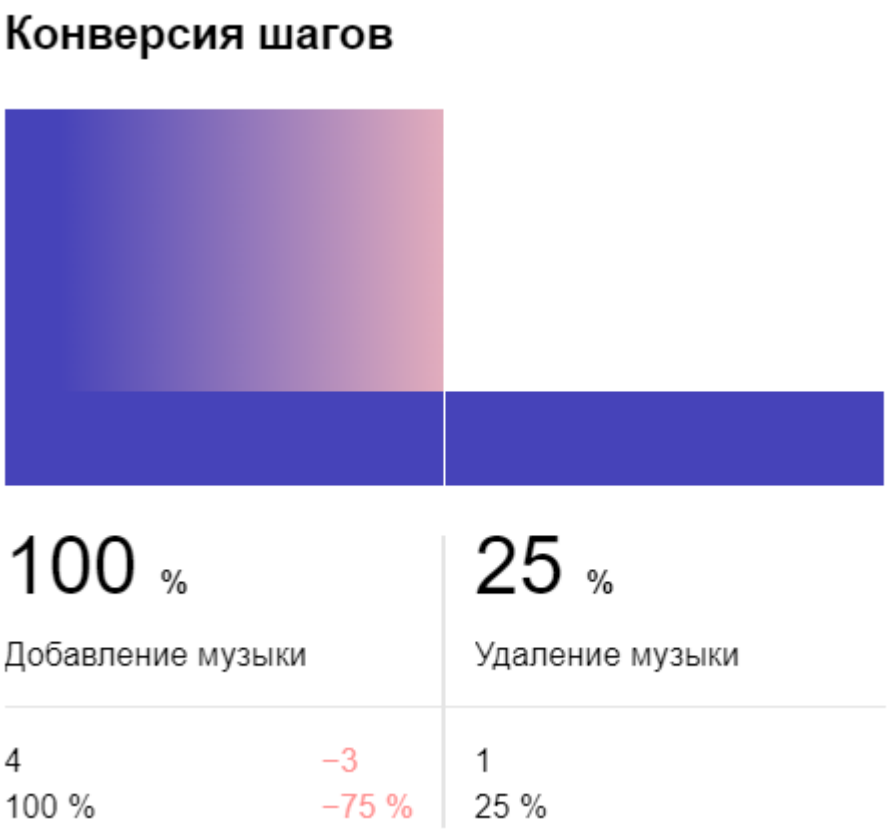


Рисунок 66 - Воронка «Музыка пользователя»

Заключение

В ходе выполнения данного курсового проекта был выполнен анализ предметной области и аналогов разрабатываемого приложения.

Для разработки приложения были разработаны макеты интерфейса, выбрана платформа приложения, построены UML диаграммы.

Для контроля версий был создан репозиторий GitHub.

При разработке приложения были реализованы следующие задачи:

— поиск музыки;

— прослушивание найденной музыки;

— добавление музыки на личную страницу;

— удаление музыки из личной страницы;

— загрузка музыки администратором приложения;

— редактирование музыки администратором приложения;

— удаление музыки администратором приложения;

Backend часть приложения и база данных были размещены в контейнере Dosker compose на хостинге.

Разработанное приложение удовлетворяет поставленным требованиям.

Все поставленные задачи были выполнены.

Список использованных источников

1. Android Programming: The Big Nerd Ranch Guide (Big Nerd Ranch Guides) / K. Marsicano, B. Gardner, B. Phillips, C Stewart. – New York: Big Nerd Ranch Guides, 2019. – 1036 с.
2. Effective Java / J. Bloch. – Reading: Addison-Wesley Professional, 2017. – 416 с.
3. Android Application Development Cookbook: 93 Recipes for Building Winning Apps / L. Wei-Meng. – New York: Wrox, 2013. – 408 с.
4. Официальная документация Yandex Metrica [Электронный ресурс]. – Режим доступа: https://metrica.yandex.com/about? – Заглавие с экрана. – (Дата обращения: 01.06.2023).
5. Официальная документация Espresso [Электронный ресурс]. – Режим доступа: https://developer.android.com/training/testing/espresso – Заглавие с экрана. – (Дата обращения: 01.06.2023).
6. Официальный сайт YouTube Music Espresso [Электронный ресурс]. – Режим доступа: https://music.youtube.com/ – Заглавие с экрана. – (дата обращения: 04.06.2023).
7. Официальный сайт Spotify Espresso [Электронный ресурс]. – Режим доступа: https://open.spotify.com/? – Заглавие с экрана. – (дата обращения: 04.06.2023).
8. Официальный сайт Яндекс Музыка Espresso [Электронный ресурс]. – Режим доступа: https://music.yandex.ru/home?= – Заглавие с экрана. – (дата обращения: 04.06.2023).