

МИНИСТЕРСТВО ВЫСШЕГО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
вычислительной техники

Кафедра
«МОиПЭВМ»

Направление подготовки 09.03.04 «Программная инженерия»

БАКАЛАВРСКАЯ РАБОТА

на тему

**Программный сервис удаленного управления
медиасистемой**

Студент	<u>Бесчетвертов</u> (подпись, дата)	Бесчетвертов Павел Александрович (ФИО полностью)
Руководитель	<u>Кольчугина</u> 21.06.21 (подпись, дата)	Кольчугина Е.А. (фамилия, инициалы)
Нормоконтролёр	<u>В</u> 22.06.21 (подпись, дата)	Такташкин Д.В. (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой _____
(подпись) Макарычев П.П.
(фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК _____
(подпись) Попова Н.А.
(фамилия, инициалы)

Пенза, 2021

Федеральное государственное бюджетное образовательное учреждение
высшего образования

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
Вычислительной техники

Кафедра
«МОиПЭВМ»

«Утверждаю»
Заведующий кафедрой
П.П. Макарычев
«21» января 2021 г

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА

1. Студент Бесчетвертев Павел Александрович гр. 17ВПИ факультета ВТ
<Фамилия, имя, отчество полностью>
2. Тема работы Программный сервис удаленного управления медиасистемой

Тема утверждена приказом ПГУ № 0027 а/б от " 21 " января 2021

2. Руководитель работы д.т.н., профессор кафедры «МОиПЭВМ» Кольчугина Е.А.

3. Задание на работу (назначение разработки, исходные данные и т.п.)

Назначение: программный продукт реализует сервис для удаленного управления медиасистемой на базе плеера AIMP

Основные функции:

- 1) Аутентификация и авторизация пользователей
- 2) Управление состоянием медиасистемы
- 3) Управление списком воспроизведения

Технология разработки: ООП

Язык программирования: Java, JavaScript

Инструментальные средства разработки: IntelliJ IDEA 2021.1.1, WebStorm 2021.1.1

Среда исполнения: Windows

4. Перечень подлежащих разработке вопросов

1) Анализ предметной области и постановка задачи

2) План разработки и оценка бюджета

3) Проектирование программных средств

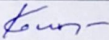
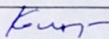
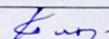
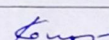
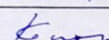
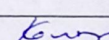
4) Реализация программных средств

5) Тестирование и анализ качества программного кода

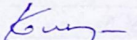
6) Рефакторинг программных средств

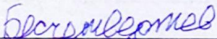
7) Оформление пояснительный записки

5. Календарный график выполнения работы

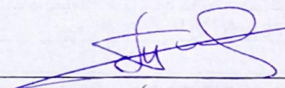
№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и постановка задачи	10	14.04.2021	
2	План разработки и оценка бюджета	5	21.04.2021	
3	Проектирование программных средств	30	25.04.2021	
4	Реализация программных средств	25	16.05.2021	
5	Тестирование и анализ качества программного кода	20	30.05.2021	
6	Оформление пояснительный записки	10	14.06.2021	

Дата выдачи задания 28.12.2020

Руководитель бакалаврской работы  28.12.2020
(подпись, дата)

Задание к исполнению принял студент  28.12.2020
(подпись, дата)

Работу к защите допустить
Декан факультета


(подпись, дата)

Реферат

Пояснительная записка содержит 111 листов, 34 рисунка, 18 таблиц, 15 использованных источников, 7 приложений.

WINDOWS, AIMP, УДАЛЕННОЕ УПРАВЛЕНИЯ, ВЕБ ПРИЛОЖЕНИЕ, ОБМЕН СООБЩЕНИЯМИ, БАЗЫ ДАННЫХ, PostgreSQL, ПРЕДМЕТНАЯ ОБЛАСТЬ.

Объект исследования – программные реализации технологий удаленного администрирования компьютерных систем.

Цель выпускной квалификационной работы – разработка программного сервиса удаленного управления медиасистемой

Технология разработки – ООП.

Среда исполнения – Windows.

В качестве базы данных программного сервиса выбрана PostgreSQL. На сегодняшний день это одна из самых популярных и востребованных баз данных и СУБД на рынке. К ее преимуществам можно отнести надежные механизмы транзакций, механизм репликации и легкую масштабируемость.

Приложение связывается с базой данных через набор RESTful сервисов, созданных с помощью технологий JavaEE на языке Java. Приложение серверной части разворачивается на сервере Apache Tomcat 8.0.4. Данная архитектура отличается легкостью разворачивания, администрирования, поддержки и расширяемости.

Клиентское приложение создано с использованием языка JavaScript на основе инструментов React в среде WebStorm.

ПГУ 09.03.04 – 08БР171.05 ПЗ

Изм.	Лист	№ докум.	Подп.	Дата	Программный сервис удаленного управления медиасистемой. Пояснительная записка.	Лит.	Лист	Листов
Разраб.		Бесчетвертев П.А.		21.06.21			3	111
Пров.		Кольчугина Е.А.		21.06.21		Группа 17ВП1		
Н. контр.		Такташкин Д.В.		21.06.21				
Утверд.		Макарычев П.П.						

Содержание

Введение	6
1 Анализ предметной области	8
1.1 Сервис удаленного управления компьютером	8
1.2 Классификация протоколов и технологий удаленного управления компьютером	9
1.3 Исследование программных продуктов-аналогов.....	11
1.3.1 Anyplace Control	11
1.3.2 Radmin.....	12
1.3.3 Hidden Administrator	13
1.3.4 Сравнительный анализ аналогичных программных средств.....	14
1.4 Постановка задачи на разработку программного сервиса удаленного управления медиасистемой	15
1.5 Анализ требований к программным средствам.....	15
1.5.1 Анализ функциональных требований	17
1.5.2 Моделирование бизнес-процессов предметной области	19
1.5.3 Разработка концептуальной модели базы данных	21
1.5.4 Анализ нефункциональных требований	22
1.6 Технологии и средства разработки.....	23
2 Проектирование и развертывание базы данных	24
2.1 Проектирование логической модели базы данных	24
2.2 Проектирование физической модели базы данных.....	25
2.3 Развертывание и заполнение базы данных тестовыми данными	26
2.3.1 Развертывание базы данных в СУБД PostgreSQL	26
2.3.2 Наполнение базы данных тестовыми данными	28
3 Реализация приложения	29

	5
3.1 Разработка базы данных.....	29
3.1.1 Разработка запросов к базе данных и их представлений для реализации аналитических бизнес-процессов	29
3.1.2 Разработка функций и хранимых процедур.....	34
3.1.3 Разработка триггеров	35
3.2 Разработка серверного приложения	38
3.3 Разработка клиентской части приложения	45
3.4 Тестирование программного сервиса уделенного управления.....	51
3.4.1 Выбор режима тестирования	51
3.4.2 Тестирование базы данных	53
3.4.3 Тестирование серверного приложения	55
3.5 Анализ качества кода	63
3.6 Планирование разработки и оценка бюджета.....	65
Заключение	69
Список использованных источников	71
Приложение А. Глоссарий.....	73
Приложение Б. Концептуальная модель базы данных.....	75
Приложение В. Скрипты для создания объектов базы данных.....	80
Приложение Г. Листинг хранимых процедур	90
Приложение Д. Листинг триггеров	96
Приложение Е. Результаты тестирования	98
Приложение Ж. Спецификация сервисов серверного приложения	106

Введение

Медиасистема – это компьютер, который снабжен программно-аппаратными средствами, реализующими технологию мультимедиа. В рамках данной работы в качестве медиасистемы будет выступать компьютер, на котором установлен медиаплеер AIMR.

Программы удаленного управления компьютерными системами – программы или функции операционной системы, включающие аппаратные интерфейсы, которые позволяют в режиме реального времени с помощью интернет-соединения, либо же в рамках локальной сети, получить доступ к управлению и администрированию удаленного компьютера [7].

Удаленное управление компьютерными системами – это перспективное направление развития технологий. В настоящее время количество компьютеров стремительно увеличивается, а также нарастает тенденция к упрощению и автоматизации повседневных задач. В обиход современного человека плотно входят технологии удаленного и бесконтактного управления. На основе технологии удаленного управления также формируются и новые рынки, например – рынок облачных вычислений, предоставляющий пользователям доступ к удаленным серверам, на которых и происходят вычисления [3].

Целью разработки сервиса удаленного управления медиасистемой является создание программного продукта, содержащего легкую в развертывании архитектуру, состоящую из серверной части, базы данных и клиентского web-приложения.

Для достижения данной цели необходимо решить следующие задачи:

- изучение практической и методической литературы;
- анализ предметной области разработки;
- анализ требований к программному обеспечению;
- проектирование программного обеспечения;
- реализация приложения;
- тестирование приложения.

Для проведения исследований в рамках этой работе был использован метод классификации.

Предметом исследования являются кроссплатформенность и безопасность протоколов и технологий удаленного администрирования компьютерных систем.

Объектом исследования являются программные реализации технологий удаленного администрирования компьютерных систем.

1 Анализ предметной области

1.1 Сервис удаленного управления компьютером

Под понятием «сервиса удаленного управления» понимаются архитектуры, протоколы, самые различные программы и сценарии, описывающие что и когда нужно делать для оптимального использования системы. Это касается не только того, как распознать ошибку и отреагировать на нее, но и повседневных процессов, таких, как установка нового программного обеспечения или добавление очередного пользователя. Чтобы все это осуществить централизованно и по возможности эффективно, нужна отработанная технология для организации удаленного доступа ко всем функциям компьютера [2].

Сейчас на рынке представлено немало программных продуктов, позволяющих получить удаленный доступ к компьютеру через интернет или локальную сеть. В большинстве случаев эти приложения объединяют два программных модуля – клиентский и серверный. Первый устанавливается на компьютер, с которого планируется осуществлять контроль за удаленным персональным компьютером, а второй – на компьютер, который требуется контролировать. Установить и настроить каждый из модулей не составляет проблемы при условии, что удаленный ПК имеет прямой выход в интернет и реальный IP-адрес. Если удаленный компьютер недоступен напрямую, то на маршрутизаторе, который имеет прямой доступ в интернет и одновременно доступ к удаленному компьютеру, необходимо настроить перенаправление портов либо воспользоваться утилитой для установления виртуального TCP/IP-соединения или туннеля между двумя компьютерами, не связанными напрямую. Что касается компьютеров с динамическим IP-адресом, то сложность в том, что совершенно неизвестно, какой IP-адрес окажется у удаленного компьютера в тот момент, когда к нему потребуются подключиться, ведь IP-адрес меняется при каждом следующем подключении. В этом случае не обойтись без подключения к сервису DynDNS или

аналогичному ему – такие сервисы отслеживают изменение динамического IP-адреса и заменяют его постоянным DNS-именем [8].

Несмотря на перечисленные выше сложности настройки, не стоит отказываться от организации удаленного доступа к ПК. Дело в том, что во многих приложениях для удаленного администрирования первой проблемы удастся легко избежать благодаря подключению не по IP-адресам, а по уникальным именам или идентификаторам ID – тогда наличие реального IP-адреса не требуется и автоматически обеспечивается обход брандмауэров, маршрутизаторов и пр. Это означает, что на практике настройка удаленного доступа к компьютеру потребует от пользователя минимум времени и усилий.

По окончании всех настроек можно приступать к удаленному управлению ПК. Обычно для этого бывает достаточно запустить клиентское приложение, выбрать интересующий компьютер и ввести пароль на получение доступа. После чего администратор получит доступ к управлению удаленным компьютером, а далее – можно запускать или останавливать приложения, загружать документы, копировать файлы с удаленного компьютера на свой ПК и обратно и т.д.

В ряде решений также предполагается выполнение таких операций, как перезагрузка и выключение удаленной машины, осуществление изменений в BIOS и др. При этом все перечисленные манипуляции производятся с помощью клавиатуры и мыши на компьютере администратора.

1.2 Классификация протоколов и технологий удаленного управления компьютером

Протоколы и технологии доступа имеют принципиальные отличия. Протоколы определяют, как структурируются относящиеся к управлению данные и осуществляется обмен ими. Они позволяют использовать стандартизованное программное обеспечение управления, однако не специфицируют, каким образом соответствующие данные физически добываются и обрабатываются. Это задача технологий доступа, именно они

позволяют дистанционно управлять устройством и удаленно запрашивать его параметры.

Принципиально такое становится возможным за счет дополнительных аппаратных средств или программного обеспечения, которое администратор устанавливает на контролируемом устройстве.

Аппаратные средства, позволяющие дистанционно управлять компьютером, можно классифицировать следующим образом [7]:

- удлинитель интерфейса;
- адаптер интерфейса;
- провайдер интерфейса.

К программным средствам относятся следующие технологии [7]:

- программа-клиент;
- программа-сервер.

Удлинитель интерфейсов – это прибор, преобразующий цифровые аудио и видеосигналы в формат для передачи по волоконно-оптическим, коаксиальным кабелям и проводам с витой парой. Или, наоборот, принимающий приходящие по таким каналам сигналы. В первом случае это передатчик, во втором – приемник.

Основное предназначение удлинителей интерфейсов – увеличивать дальность передачи сигнала в отличном разрешении без качественных потерь. Объединив кабелями несколько таких приборов в единую каскадную систему возможно без искажений осуществлять трансляции на большие расстояния. В таких системах репитеры также могут служить усилителями.

Адаптер интерфейса – это устройство, обеспечивающее сопряжение и взаимодействие двух или более технических средств с различными интерфейсами.

Провайдер интерфейса – расширяющий устройство интерфейс, который не находится в распоряжении данного устройства.

Серверные программные технологии представляют собой программы, которые используют коммуникационные интерфейсы контролируемого устройства и таким образом предлагают функциональность управления.

Клиенты – это пользовательские программы, взаимодействие которых осуществляется с помощью протокола с любой технологией доступа, программной или аппаратной, при условии, что они общаются посредством того же самого протокола.

1.3 Исследование программных продуктов-аналогов

В настоящее время существует множество приложений, предоставляющих возможность удаленного управления компьютером через локальную или глобальную сети. Рассмотрим некоторые из них.

1.3.1 Anyplace Control

Anyplace Control – программа, позволяющая получить удаленный доступ к компьютеру через Интернет или локальную сеть и в полной безопасности управлять удаленным компьютером в режиме реального времени [4]. Данная программа обладает следующими преимуществами и недостатками:

Преимущества:

- отображение рабочего стола удаленного компьютера в реальном времени;
- использование мыши и клавиатуры для управления удаленным компьютером;
- обмен файлами между компьютерами;
- копирование текста, графики или других данных из буфера обмена одного компьютера и перенос их на другой компьютер.

Недостатки:

- отсутствие поддержки мультиэкранных систем;
- невозможность проведения записи с экрана;
- неудачный алгоритм шифрования данных (128 бит вместо 256);
- режим работы через аккаунт-подключение, когда трафик пропускается через промежуточный веб-сервер, который принадлежит разработчикам Anyplace Control. С одной стороны, они гарантируют

сохранность подобной информации, с другой - пользователи становятся в своего рода зависимость от стабильности работы сервера вендора, который может, например, выйти на длительное время из строя.

Таким образом можно сделать вывод, что Anyplace Control является мощным программным средством для реализации удаленного управления компьютером, но является не самым надежным и небезопасным решением, так как весь обрабатываемый трафик проходит через сервера компании разработчика.

1.3.2 Radmin

Radmin – это программа удаленного администрирования для платформы Windows, которая позволяет полноценно работать сразу на нескольких удаленных компьютерах с помощью обычного графического интерфейса [5].

Преимущества:

- удаленное управление на аппаратном уровне с поддержкой технологии Intel® AMT;
- текстовый и голосовой чаты;
- безопасный обмен файлами с возможностью дозагрузки;
- возможность отображения экрана удалённого ПК в отдельном окне или в полноэкранном режиме с плавным изменением масштаба и сохранением пропорций;
- совместимость с ОС Linux;
- запись в лог файл имени пользователя и DNS расшифровки его адреса;
- поддержка нескольких одновременных соединений.

Недостатки:

- необходимость установки на компьютер клиента программного обеспечения, видео драйвера;
- невозможно вводить команды на русском языке;
- непрозрачная работа с буфером обмена.

Radmin представляет из себя отличный инструмент для управления удаленным компьютером. Данная программа содержит множество полезных функций, не передает трафик через сервера компании разработчика, но использовать как готовый вариант для удаленного управления медиасистемой нельзя: наличие специального клиентского приложения, которое не является кроссплатформенным. А также у данной разработки отсутствует поддержка русского языка.

1.3.3 Hidden Administrator

Программа Hidden Administrator предназначена для удаленного скрытого наблюдения или управления компьютерами локальной сети или рабочей группы [6].

Преимущества:

- отображение рабочего стола дистанционного ПК в реальном времени;
- управление удаленным компьютером при помощи мыши и клавиатуры;
- одновременное наблюдение за несколькими компьютерами с возможностью их администрирования;
- удаленный доступ в режиме командной строки;
- передача звука по сети в режиме реального времени.

Недостатки:

- необходимость установки программного обеспечения на компьютер клиента;
- сложная удаленная установка.

Таким образом, можно сделать вывод, что программ Hidden Administrator предоставляет возможность удаленного управления компьютером в удобной форме, но также, как и Radmin имеет клиентское приложение только для операционной системы Windows. Из этого следует, что данная разработка не подходит для обеспечения удаленного управления медиасистемой.

1.3.4 Сравнительный анализ аналогичных программных средств

Для того чтобы выполнить сравнительный анализ описанных выше средств удаленного управления компьютерами с собственной разработкой, необходимо определиться с критериями сравнения.

Как уже было отмечено, для сервиса удаленного управления медиасистемой необходимы следующие функциональные возможности:

- web-интерфейс для обеспечения доступа к возможностям сервиса;
- наличие русскоязычного интерфейса;
- отсутствие передачи трафика через сторонние сервера;
- возможность расширения функционала системы;
- легкая установка и настройка соединения.

Сравнение описанных программных средств по выбранным функциональным критериям приведено в таблице 1.

Таблица 1 – Сравнительный анализ программных средств

Критерии \ Системы	Anyplace Control	Radmin	Hidden Administrator
Отсутствие передачи трафика через сторонние сервера	нет	да	да
Русскоязычный интерфейс	да	нет	да
Возможность расширения функционала системы	нет	нет	нет
Легкая установка и настройка соединения	да	нет	нет
Web-версия системы	нет	нет	нет

Проведенный сравнительный анализ показал, что ни одна из систем не имеет необходимого для сервиса web-интерфейса. Также, большинство из перечисленных систем имеют и прочие недостатки. Можно сделать вывод, что необходима разработка собственного программного средства.

1.4 Постановка задачи на разработку программного сервиса удаленного управления медиасистемой

В рамках данной работы необходимо разработать сервис для удаленного управления медиасистемой.

Медиасистема состоит из компьютера, под управлением операционной системы Windows 10, на котором запущено приложение медиаплеера AIMP и развернута серверная часть разрабатываемого сервиса, включая сервер базы данных. А также из множества клиентов, подключенных к серверному приложению. Каждый из клиентов должен пройти авторизацию, чтобы получить доступ к удаленному управлению медиасистемой. Всем авторизованным пользователям должна быть доступна информация о текущем состоянии системы, а также детальная информация по воспроизводимому треку. Пользователь может запустить, а также остановить воспроизведение трека. Переключить воспроизведение к следующему, либо же к предыдущему треку. Пользователь с правами администратора дополнительно имеет возможность изменить список воспроизведения. Также пользователь может изменять значение громкости воспроизводимого трека.

Данные признаки определяют границы предметной области разрабатываемого приложения.

Созданное программное обеспечение должно быть отлажено и протестировано на наборах данных предметной области.

1.5 Анализ требований к программным средствам

Согласно поставленной задаче, в рамках данной работы необходимо разработать программный сервис, способный выполнять следующие информационные задачи:

- 1) обеспечение регистрации и авторизации пользователей;
- 2) создание пользователей;
- 3) загрузка/удаление песен;
- 4) управление плеером;
- 5) управление списком воспроизведения;

- 6) управление пользователями;
- 7) просмотр списка воспроизведения (аналитическая задача);
- 8) просмотр списка пользователей (аналитическая задача);
- 9) возможность пользователю поставить оценку с отзывом о песне.

Категории пользователей программного сервиса удаленного управления медиасистемой и соответствующие им информационные задачи представлены в таблице 2.

Таблица 2 – Выделенные категории пользователей

Категория пользователей	Фрагмент структуры предметной области	Информационные задачи
Гость	Обеспечение авторизации и регистрации в системе.	1
Пользователь	Управление плеером и возможность редактирования своего профиля.	3, 4, 7, 9
Администратор	Управление пользователями и списком воспроизведения.	2, 5, 6, 8

При этом администратор помимо своих функциональных возможностей наследует функционал от пользователя и может выполнять все информационные задачи пользователя.

Исходя из всего описанного выше, целью разработки программного сервиса удаленного управления медиасистемой является создание работоспособного программного сервиса, обладающего функционалом для решения всех описанных выше информационных задач в рамках предметной области удаленного управления медиасистемой и поддерживающей все выделенные категории пользователей.

1.5.1 Анализ функциональных требований

Согласно поставленным информационным задачам, были определены следующие функциональные требования для сервиса удаленного управления медиасистемой:

- авторизироваться в системе;
- запросить роль в системе;
- выйти из системы;
- отправить команду серверу;
- остановить\возобновить воспроизведение;
- изменить позицию воспроизведения;
- изменить плейлист воспроизведения;
- изменить громкость воспроизведения;
- переключить трек;
- получить информацию о треке.

На рисунке 1 представлена диаграмма вариантов использования.

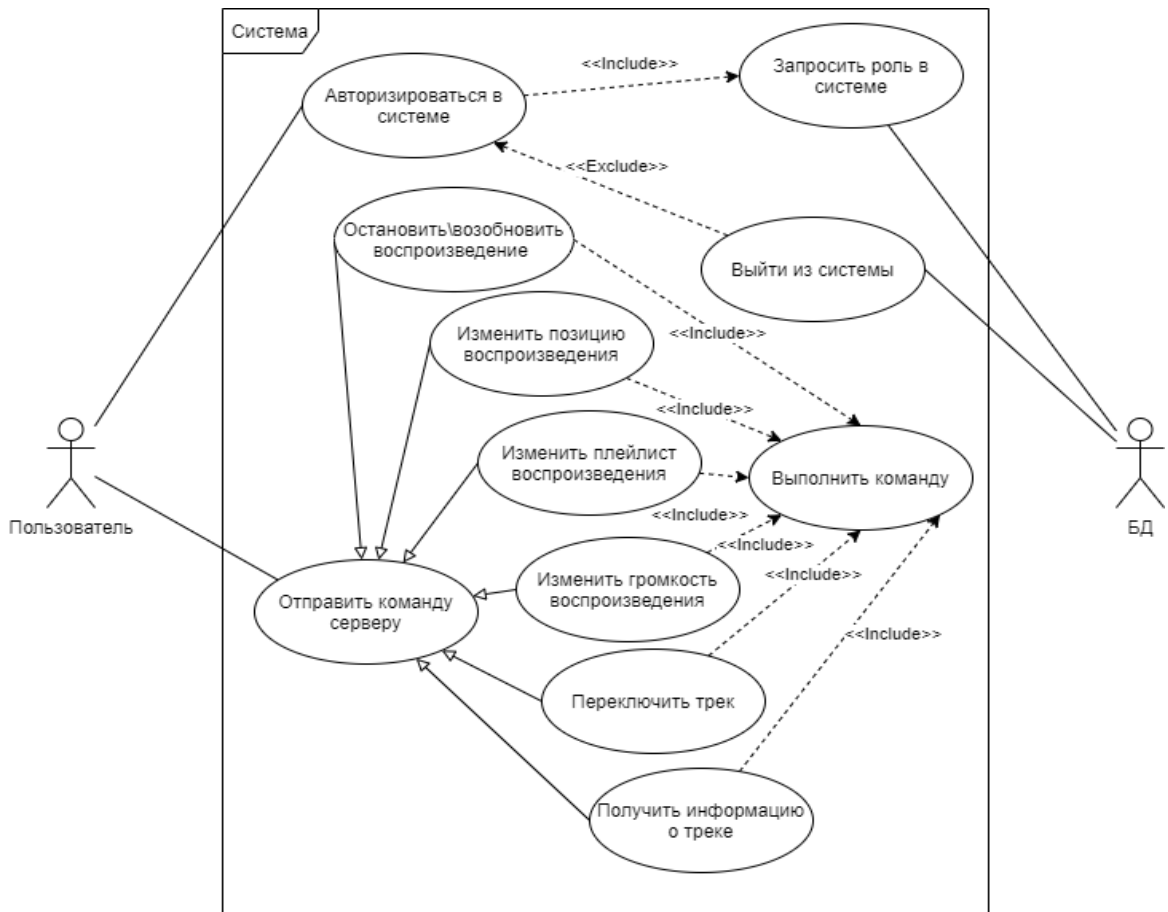


Рисунок 1 – Диаграмма вариантов использования

На диаграмме вариантов использования представлены функциональные требования к разрабатываемому программному сервису. Была составлена спецификация для основных прецедентов диаграммы вариантов использования (рисунок 2, 3).

Прецедент: авторизоваться в системе
ID: 1
<p>Краткое описание:</p> <p>Пользователь авторизуется в системе с помощью email и пароля. Получает авторизационный токен, содержащий информацию о роле пользователя в системе.</p>
<p>Главные актёры:</p> <p>Пользователь.</p>
<p>Предусловие:</p> <p>Пользователь не авторизован в системе.</p>
<p>Основной поток:</p> <ol style="list-style-type: none"> 1. Пользователь открывает страницу приложения 2. Пользователь вводит email 3. Система проверяет введенный email на валидность и доступность среди списка пользователей 4. Пользователь вводит пароль 5. Пользователь отправляет запрос на авторизацию 6. Система возвращает пользователю его роль в системе и токен <p>Альтернативный поток:</p> <ol style="list-style-type: none"> 5.a. Система возвращает сообщение об ошибке, авторизационные данные некорректны
<p>Постусловие:</p> <p>Пользователь авторизован в системе. Пользователь получает доступ к функциям системы, разрешенным для его текущей роли.</p>

Рисунок 2 – Спецификация прецедента «Авторизоваться в системе»

Прецедент: отправить команду серверу
ID: 2
Краткое описание: Проверяет, имеются ли у пользователя права для выполнения команды, в случае их наличия отправляет серверу ID и параметры выполняемой команды
Главные актёры: Пользователь.
Предусловия: пользователь авторизировался в системе
Основной поток: 1. Пользователь выбирает команду для сервера и устанавливает ее параметры 2. Система проверяет права доступа пользователя к выполняемой команде 3. Система выполняет команду Альтернативный поток: 3.а. Система возвращает сообщение об ошибке, если у пользователя недостаточно прав для исполняемой команды
Постусловие: Запрошенная команда выполнена

Рисунок 3 – Спецификация прецедента «Отправить команду серверу»

В результате анализа функциональных требований были выделены варианты использования программы. Для основных вариантов использования была приведена спецификация, которая позволяет понять последовательность действий, которые выполняет система при использовании функций пользователем [8].

1.5.2 Моделирование бизнес-процессов предметной области

Для разрабатываемого сервиса удаленного управления медиасистемой были выделены бизнес-процессы, представленные в таблице 3.

Таблица 3 – Выделенные бизнес-процессы

№	Бизнес-процесс	Исполнитель	Входные данные		Выходные данные		Ограничения
			Поставщик	Содержание	Потребитель	Содержание	
1	Регистрация	Гость	Гость	Email, логин, пароль	Приложение	Email, пароль.	- Необходимо ограничение дублирования записей пользователей - Использовать уникальный идентификатор записей
2	Авторизация	Гость	Гость	Email, пароль	Клиентское приложение	Результат авторизации	- Использовать алгоритм асинхронного шифрования пароля
3	Формирование списка воспроизведения	Пользователь, администратор	Пользователь, администратор	Запрос	Пользователь, администратор	Список воспроизведения	- Быть авторизованным
4	Формирование списка пользователей	Администратор	Администратор	Запрос	Администратор	Список зарегистрированных пользователей	- Быть авторизованным
5	Управление списком воспроизведения	Администратор	Администратор	Идентификатор песни, выполняемая над песней команда, параметры команды	Клиентское приложение	Обновленный список воспроизведения	- Быть авторизованным
6	Загрузить песню в систему	Пользователь, администратор	Пользователь, администратор	Загружаемая песня, описание загружаемой песни, данные пользователя	Пользователь, администратор	Загруженная песня	- Быть авторизованным
6	Просмотр подробной информации о песне	Пользователь, администратор	Пользователь, администратор	Идентификатор песни	Пользователь, администратор	Данные о песне	- Быть авторизованным
7	Управлять плеером	Пользователь, администратор	Пользователь, администратор	Тип выполняемой команды, параметры выполняемой команды	Клиентское приложение	Результат выполнения команды плеером	- Быть авторизованным, плеер запущен
8	Оставить оценку с отзывом о песне	Пользователь, администратор	Пользователь, администратор	Оценка, отзыв	Пользователь, администратор	Оценка и отзыв о песне	- Быть авторизованным

Среди выделенных бизнес-процессов присутствуют аналитические бизнес-процессы, требующие использования представлений и триггеров.

1.5.3 Разработка концептуальной модели базы данных

На основе анализа предметной области и выделенных функциональных требований построена концептуальная модель базы данных, которая представлена в приложении Б.

В концептуальной модели выделены следующие сущности:

- пользователь;
- роль пользователя;
- песня;
- оценка.

Подробное описание сущностей представлено в таблице Б.1. Связи между сущностями описаны в таблице Б.2

После проведения структурирования информации в предметной области без учета деталей возможной реализации базы данных была составлена концептуальная модель связей между сущностями в нотации «сущность-связь». Данная диаграмма представлена на рисунке 4.

Последним этапом разработки концептуальной модели было проведено проектирование атрибутов сущностей. На данном уровне моделирования были спроектированы лишь атрибуты, несущие информационную нагрузку для каждой сущности, то есть, не рассматривая фиктивные атрибуты и внешние ключи. Атрибуты сущностей представлены в таблицах Б.3 – Б.6.

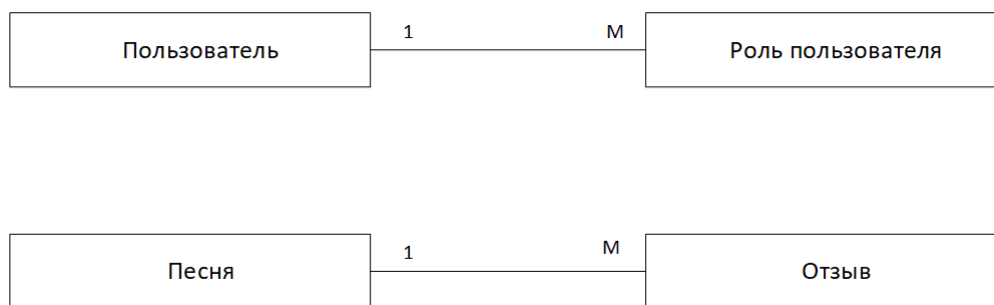


Рисунок 4 – Концептуальная модель базы данных в нотации «сущность-связь»

Как предполагает стандарт нотации «сущность-связь», в концептуальной модели отображены мощности связей между сущностями предметной области. Во всех случаях, использован тип связи «один-ко-многим». При использовании данной связи несколько экземпляров зависимой сущности могут принадлежать одному экземпляру родительской сущности. Например, один пользователь может обладать несколькими ролями в рамках системы.

Таким, образом, созданная концептуальная модель предоставляет исходную формализованную информацию о сущностях и семантики предметной области сервиса удаленного управления медиасистемой.

1.5.4 Анализ нефункциональных требований

Одним из основных требований к программе является то, что все подключенные к серверному приложению пользователи должны иметь возможность одновременно производить управление системой. Кроме того, каждый из пользователей должен получать актуальную информацию о состоянии системы.

Исходя из вышеописанных требований можно сделать вывод, что система должна осуществлять параллельные вычисления: сам плеер AIMR и серверное приложение должны работать параллельно для реализации удаленного управления плеером.

Кроме того, для сервиса удаленного управления желательно обеспечить независимость его работоспособности от работоспособности плеера.

Таким образом в будущей программе необходимо реализовать достаточно много параллельных вычислений. Одной из важных задач при проектировании такой программы является необходимость обеспечить правильную последовательность взаимодействий между различными вычислительными процессами, а также координацию ресурсов, разделяемых между процессами. Для решения данной задачи сформулировано еще одно нефункциональное требование – «способ обмена информацией между параллельными процессами – обмен сообщениями».

1.6 Технологии и средства разработки

Разработка серверного приложения сервиса удаленного управления медиасистемой проводилась с использованием языка программирования Java 1.8. Для сборки программы использовался кроссплатформенный фреймворк Java maven. Выбор данного фреймворка для сборки проекта обусловлен простотой подключения необходимых библиотек к проекту, а также большим набором предоставляемых библиотек в рамках репозитория maven.

В качестве среды разработки было решено использовать IntelliJ IDEA. Данная среда разработки включает в себя возможности работы с данными технологиями, а также она предоставляет возможности подсветки синтаксиса, интеллектуального анализа кода, средства для генерации технической документации и автоматического рефакторинга для Java 1.8. Кроме того, IntelliJ IDEA является кроссплатформенной средой разработки, которая может работать под управлением следующих операционных систем: Windows, Linux и MacOS.

Разработка клиентского приложения в среде разработки WebStorm. В качестве языка программирования использовался JavaScript с набором инструментов, предоставляемым библиотекой React 17. В качестве менеджера пакетов использовался npm. Данные инструменты позволяют достаточно легко подключать новые модули к проекту. А также удобны при построении интерфейса, так как React позволяет все приложение разбивать на переиспользуемые компоненты с ограниченным набором логики, необходимым для конкретной задачи.

2 Проектирование и развертывание базы данных

При построении программного сервиса удаленного управления медиасистемой для хранения учетных данных пользователей, их ролей в системе сервиса, а также данных о загруженных пользователями в систему песнях необходимо будет хранить в базе данных. База данных может размещаться на одном сервере с сервисом удаленного управления медиасистемой, либо же на удаленном сервере. В случае, если сервер базы данных будет развернут на удаленном сервере, потребуется дополнительная конфигурация СУБД для обеспечения доступа к базе данным со сторонних IP адресов.

Для доступа к сервису удобнее всего будет использовать веб-интерфейс в качестве клиентского приложения. Это позволит обеспечить доступ к сервису из любой точки планеты, где есть доступ к интернет-соединению. Также использование веб приложения позволяет исключить отдельные реализации для каждой новой платформы, что позволяет без дополнительных затрат значительно увеличить охват аудитории.

2.1 Проектирование логической модели базы данных

Используя концептуальную модель, разработанную ранее, была создана логическая модель базы данных с помощью программы проектирования и документирования баз данных Erwin Data Modeler [1]. Полученная логическая модель в виде IDEF1X-диаграммы продемонстрирована на рисунке 5.

Созданная с помощью автоматизированных средств документации Erwin Data Modeler таблица отношений логической модели базы данных представлена как таблица 4.

Таблица 4 – Описание отношений логической модели

Name	Type	Definition
Пользователь	Independent	Записи пользователей сервиса
Роль пользователя	Independent	Роли пользователей в рамках сервиса
Песня	Independent	Сущность, описывающая объект песни
Оценка	Independent	Оценка и отзыв пользователя о песне

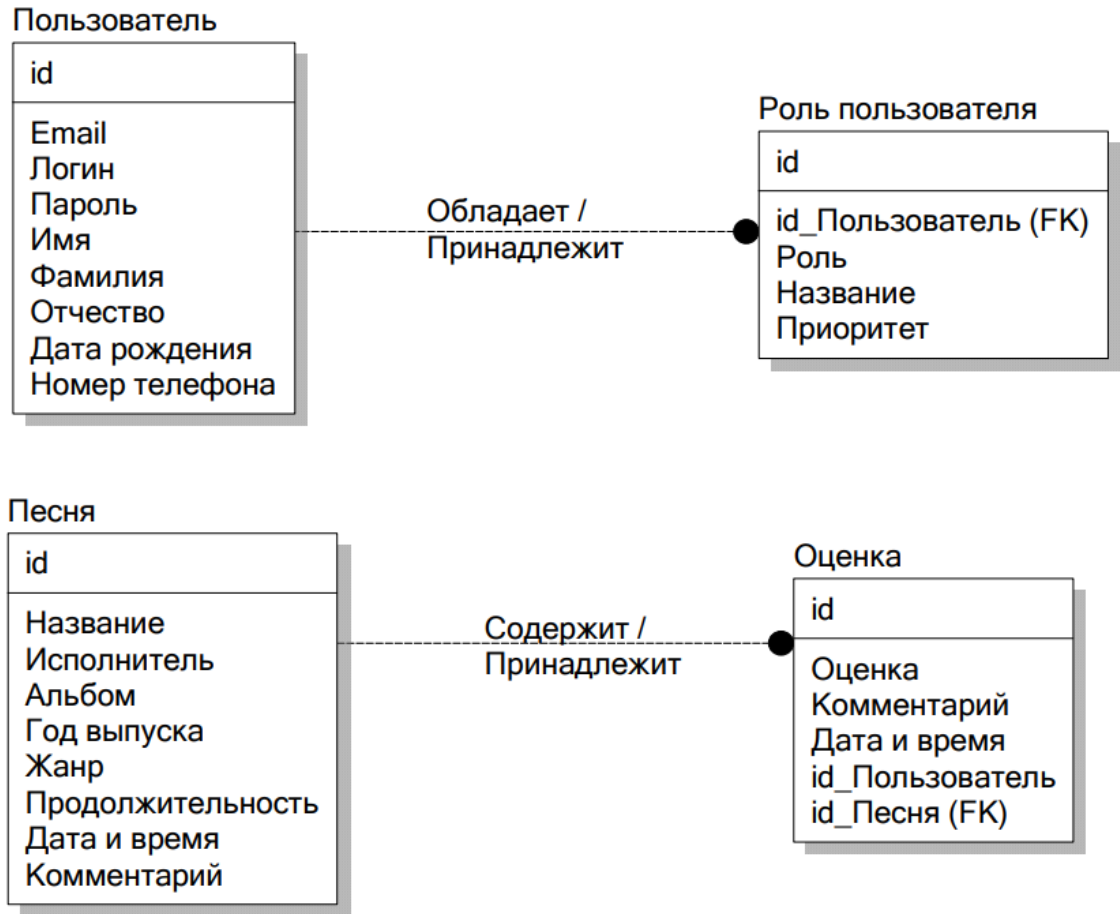


Рисунок 5 – Логическая модель базы данных в виде IDEF1X- диаграмм

Созданная логическая модель является начальным прототипом проектируемой базы данных и описывает отношения, связи между отношениями и атрибуты.

2.2 Проектирование физической модели базы данных

Используя логическую модель базы данных, была создана физическая модель базы данных с помощью программы проектирования и документирования баз данных Erwin Data Modeler. Полученная физическая модель в виде IDEF1X-диаграммы продемонстрирована на рисунке 6.

Созданное с помощью автоматизированных средств документации Erwin Data Modeler описание таблиц физической модели базы данных представлена как таблица 5.

Таблица 5 – Описание таблиц физической модели

Name	Definition
user	Записи пользователей сервиса
user_role	Роли пользователей в рамках сервиса
soundtrack	Сущность, описывающая объект песни

Продолжение таблицы 5

Name	Definition
feedback	Оценка и отзыв пользователя о песне

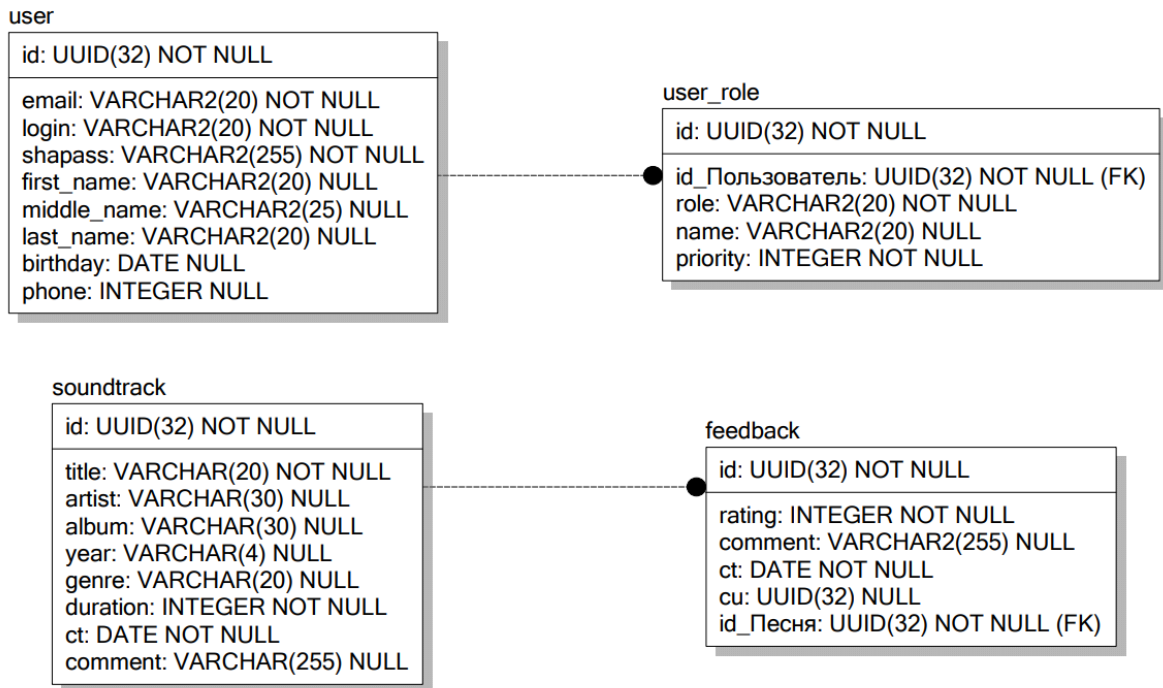


Рисунок 6 – Физическая модель базы данных в виде IDEF1X-диаграммы

Созданная физическая модель базы данных отображает детали физического уровня организации проектируемой базы данных, включая определение типов данных и режимов доступа к данным в специфике выбранной базы данных.

2.3 Развертывание и заполнение базы данных тестовыми данными

2.3.1 Развертывание базы данных в СУБД PostgreSQL

В качестве СУБД информационной системы выбрана PostgreSQL. На сегодняшний день это одна из самых популярных и востребованных баз данных и СУБД на рынке. К ее преимуществам можно отнести надежные механизмы транзакций, механизм репликации и легкую масштабируемость. Кроме того, установочный пакет СУБД PostgreSQL содержит набор востребованных утилит. Например, утилиты `pg_restore` и `pg_backup` позволяют

легко производить резервное копирование базы данных и воспроизведение базы данных из сохраненных копий.

Для работы с базой данных через СУБД PostgreSQL использована программа DBeaver, содержащая удобный редактор SQL скриптов, редактор таблиц баз данных с возможностью генерации диаграммы данных.

Используя созданную в среде CA Erwin Data Modeler физическую модель, был создан скрипт создания базы данных на языке PL/pgSQL, который представлен в приложении В.

Результат выполнения данного скрипта в среде DBeaver представлен на рисунке 7.

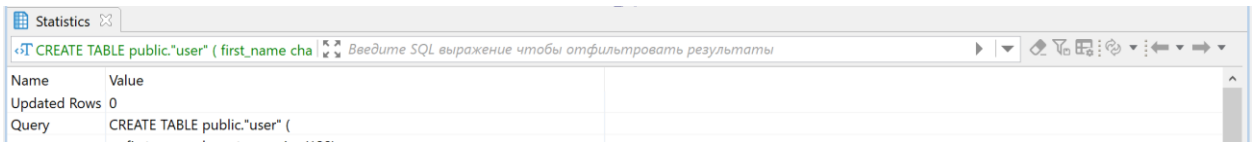


Рисунок 7 – Результат выполнения скрипта создания базы данных

На рисунке 8 представлена диаграмма данных развернутой базы данных, построенная в среде DBeaver.

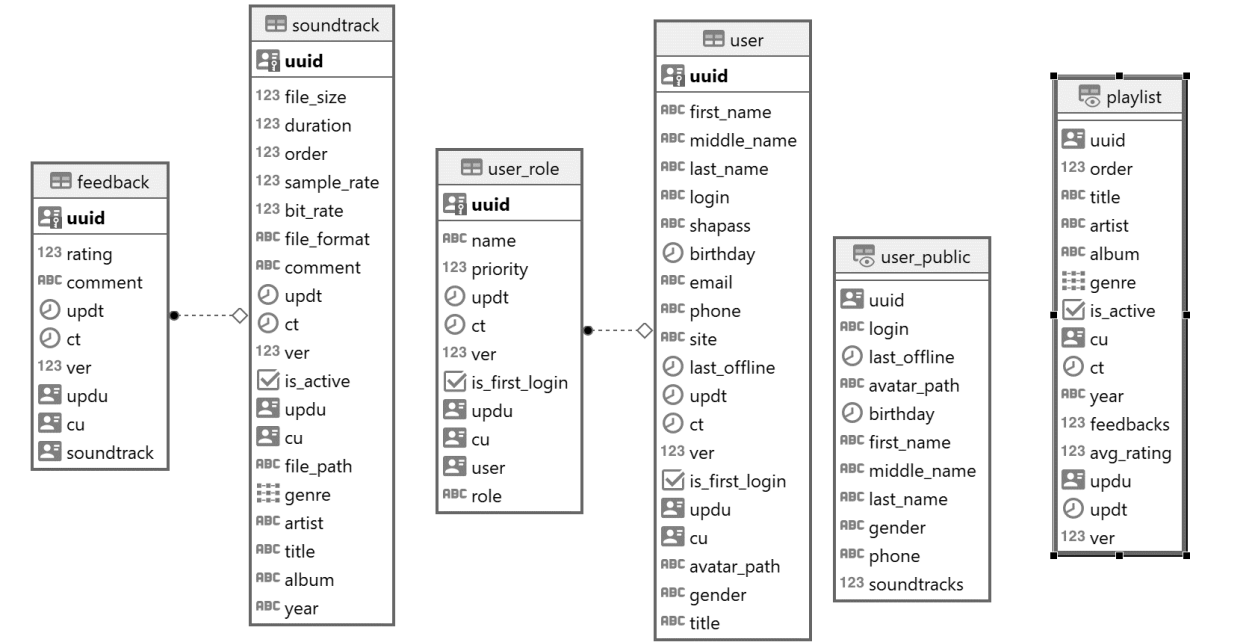


Рисунок 8 – Диаграмма данных развернутой базы данных

Построенная в среде DBeaver диаграмма данных отражает состояние базы данных после выполнения скрипта генерации данной базы данных и так же отражает результат создания базы данных.

Диаграмма отражает таблицы и атрибуты с указанием использованных типов данных.

2.3.2 Наполнение базы данных тестовыми данными

Наполнение базы данных происходило с использованием команд INSERT языка SQL, которые были выполнены через СУБД. Пример скрипта представлен в листинге 1.

Листинг 1 – Пример скрипта наполнения базы данных

```
INSERT INTO public."user"
(first_name,middle_name,last_name,login,shapass,birthday,email,p
hone,site,last_offline,updt,ct,ver,is_first_login,uuid,updu,cu,a
vatar_path,gender,title) VALUES
('test','test','test','test','937e8d5fbb48bd4949536cd65b8d35c426
b80d2f830c5c308e2cdec422ae2244',NULL,'test@test.test','test','te
st','2021-06-02 11:42:35.498','2021-06-02 11:42:35.498','2021-
05-19 12:42:19.936',5,true,'814d08b6-b886-11eb-9a34-
7f0ea57a76dd','814d08b6-b886-11eb-9a34-
7f0ea57a76dd',NULL,NULL,'Male',NULL);
```

Полный код, использованный для наполнения базы данных тестовыми данными, представлен в приложении В. Результат наполнения представлен на рисунках 9 и 10. Скриншоты сделаны в среде DBeaver.

	ABC first_name	ABC middle_name	ABC last_name	ABC login	ABC shapass	birthday	ABC email
1	test	test	test	test	937e8d5fbb48bd4949536cd65b8d35c426b80d2f830c5c	[NULL]	test@test.test
2	[NULL]	[NULL]	[NULL]	test2	937e8d5fbb48bd4949536cd65b8d35c426b80d2f830c5c	[NULL]	test@test2.test
3	Павел	[NULL]	Бесчетвертев	aVoid	dff3f9f8bce8f7c9de2a373d233020d36c10d265275d7e2	[NULL]	avoidpnz@gmail.com
4	[NULL]	[NULL]	[NULL]	test	9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822	[NULL]	test.ui@test.ru
5	[NULL]	[NULL]	[NULL]	test	9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822	[NULL]	test.json@test.ru

Рисунок 9 – Наполненная таблица пользователей

	123 file_size	123 duration	123 order	123 sample_rate	123 bit_rate	ABC file_format	ABC comment	updt	ct
1	6 657 530	165	2	44 100	320 000	mp3	[NULL]	2021-05-31 10:56:44	2021-05-24 21:50:33
2	9 050 317	225	3	44 100	320 000	mp3	[NULL]	2021-05-31 10:57:27	2021-05-24 00:21:15
3	5 339 890	132	4	44 100	320 000	mp3	[NULL]	2021-05-31 10:55:26	2021-05-24 19:54:07
4	9 984 515	248	5	44 100	320 000	mp3	[NULL]	2021-05-25 21:01:28	2021-05-24 21:51:51
5	6 314 790	156	6	44 100	320 000	mp3	[NULL]	2021-05-25 14:35:58	2021-05-24 19:59:49
6	6 685 731	166	7	44 100	320 000	mp3	[NULL]	2021-05-24 21:57:21	2021-05-24 20:03:06
7	7 801 700	194	8	44 100	320 000	mp3	ppp	2021-05-24 21:35:29	2021-05-24 17:36:13
8	6 057 729	150	9	44 100	320 000	mp3	[NULL]	2021-05-27 11:32:34	2021-05-24 22:14:48
9	8 206 097	204	10	44 100	320 000	mp3	[NULL]	2021-05-25 14:35:52	2021-05-24 00:24:40
10	6 805 888	169	11	44 100	320 000	mp3	1234	2021-05-31 18:54:16	2021-05-31 18:54:16
11	3 841 500	95	12	44 100	320 000	mp3	1234567	2021-05-27 14:06:59	2021-05-27 14:06:34
12	6 362 899	158	1	44 100	320 000	mp3	[NULL]	2021-06-02 11:56:52	2021-05-24 00:22:57

Рисунок 10 – Наполненная таблица песен

Подобным способом тестовыми данными были заполнены все таблицы базы данных. Это необходимо для облегчения последующих этапов разработки. На тестовых данных удобно отлаживать запросы, представления, процедуры и работу клиентского приложения.

3 Реализация приложения

3.1 Разработка базы данных

3.1.1 Разработка запросов к базе данных и их представлений для реализации аналитических бизнес-процессов

Для аналитических бизнес-процессов сервиса были разработаны запросы и представления.

Для аналитического бизнес-процесса «Формирование списка воспроизведения» создан запрос. Код запроса представлен в листинге 2.

Листинг 2 – Запрос для аналитического бизнес-процесса «Формирование списка воспроизведения»

```
select
    soundtrack.uuid,
    soundtrack."order",
    soundtrack.title,
    soundtrack.text,
    soundtrack.artist,
    soundtrack.album,
    soundtrack.genre,
    soundtrack.is_active,
    soundtrack.cu,
    soundtrack.ct,
    soundtrack.year,
    count (feedback.soundtrack)
as feedbacks,
    avg (feedback.rating)
as avg_rating,
    soundtrack.updu,
    soundtrack.updt,
    soundtrack.ver
from
    soundtrack
left join feedback on
```

```

        soundtrack.uuid = feedback.soundtrack
group by
        soundtrack.uuid
order by
        avg_rating desc
LIMIT 200 OFFSET 0;

```

Результат работы запроса показан на рисунке 11.

	uuid	123 order	ABC title	ABC artist	ABC album	genre
1	5d7f40c9-ed63-4398-9165-ad1b21d97d68	9	Аромат	Ramil'	Громкие новинки Декабря 2	{Pop,Dance,Other}
2	df92a5a8-cec0-4585-9010-7a9f7b33cb8e	5	Losing My Religion	Dj Antonio, Bertie Scott	Громкие новинки Декабря 2	{Pop,Dance,Other}
3	8059141c-de34-449f-a8ad-958c35cbf1e7	7	Че там у хохлов	YARMAK	Громкие новинки Декабря 2	{Pop,Dance,Other}
4	cd5616a8-02cb-4517-81f6-6ce3b838829e	8	Тренировочный день	КУОК, Markul	Громкие новинки Декабря 2	{Pop,Dance,Other}
5	cb361d80-c3ae-43c0-b8da-49550ec05e8e	6	СТАТУС ВЛЮБЛЕНА	ХАННА	Громкие новинки Декабря 2	{Pop,Dance,Other}
6	bb3c1176-01bd-4327-9253-add309364574	2	Друг помоги мне	HENSY, Slame	Громкие новинки Декабря 2	{Pop,Dance,Other}
7	61aea7e0-b94c-413f-8468-303b31493e49	4	Big Money	Rakhim	Громкие новинки Декабря 2	{Pop,Dance,Other}
8	87d5575d-3320-4542-b717-4f3973f6a6a6	3	Высоко	ANIVAR	Громкие новинки Декабря 2	{Pop,Dance,Other}
9	f8ac10e9-3ee9-46c1-979a-6025ed8562bc	10	Dreams	David Guetta, MORTEN, Lanie Gardner	Громкие новинки Декабря 2	{Pop,Dance,Other}
10	934a2f8e-955c-4797-a172-ef6bd68f84e0	11	Меланхолия	Elvira T	Громкие новинки Декабря 2	{Pop,Dance,Other}
11	2c1f097a-9b0f-4a49-bbb3-11efa28c4dc0	1	Reminds Me Of You	Juice WRLD, The Kid LAROI	Громкие новинки Декабря 2	{Pop,Dance,Other}
12	f83e4381-1617-4151-83bb-11abd44c8a23	12	АУФ	Mia Boyka	Громкие новинки Декабря 2	{Pop,Dance,Other}

Рисунок 11 – Результат выполнения запроса

Разработано соответствующее этому запросу представление. Код представления находится в листинге 3.

Листинг 3 – Код представления для аналитического бизнес-процесса «Формирование списка воспроизведения»

```

CREATE OR REPLACE VIEW public.playlist
AS SELECT soundtrack.uuid,
        soundtrack."order",
        soundtrack.title,
        soundtrack.artist,
        soundtrack.album,
        soundtrack.genre,
        soundtrack.is_active,
        soundtrack.cu,
        soundtrack.ct,
        soundtrack.year,
        count(feedback.soundtrack) AS feedbacks,
        avg(feedback.rating) AS avg_rating,
        soundtrack.updu,
        soundtrack.updt,

```

```

soundtrack.ver
FROM soundtrack
LEFT JOIN feedback ON soundtrack.uuid = feedback.soundtrack
GROUP BY soundtrack.uuid
order by
avg_rating desc;

```

Результат обращения к данному представлению показан на рисунке 12.

uuid	order	title	artist	album	genre
61aea7e0-b94c-413f-8468-303b31493e49	4	Big Money	Rakhim	Громкие новинки Декабря 2	{Pop,Dance,Other}
df92a5a8-cec0-4585-9010-7a9f7b33cb8e	6	Losing My Religion	Dj Antonio, Bertie Scott	Громкие новинки Декабря 2	{Pop,Dance,Other}
8059141c-de34-449f-a8ad-958c35cbf1e7	8	Чё там у хохлов	YARMAK	Громкие новинки Декабря 2	{Pop,Dance,Other}
cd5616a8-02cb-4517-81f6-6ce3b838829e	9	Тренировочный день	KYOK, Markul	Громкие новинки Декабря 2	{Pop,Dance,Other}
cb361d80-c3ae-43c0-b8da-49550ec05e8e	7	СТАТУС ВЛЮБЛЕНА	ХАННА	Громкие новинки Декабря 2	{Pop,Dance,Other}
f83e4381-1617-4151-83bb-11abd44c8a23	13	АУФ	Mia Boyka	Громкие новинки Декабря 2	{Pop,Dance,Other}
87d5575d-3320-4542-b717-4f3973f6a6a6	3	Высоко	ANIVAR	Громкие новинки Декабря 2	{Pop,Dance,Other}
f8ac10e9-3ee9-46c1-979a-6025ed8562bc	11	Dreams	David Guetta, MORTEN, Lanie Gardner	Громкие новинки Декабря 2	{Pop,Dance,Other}
2c1f097a-9b0f-4a49-bbb3-11efa28c4dc0	1	Reminds Me Of You	Juice WRLD, The Kid LAROI	Громкие новинки Декабря 2	{Pop,Dance,Other}
9c671f00-c2b9-11eb-a5c8-fb3579c7300c	5	test	[NULL]	[NULL]	NULL
934a2f8e-955c-4797-a172-ef6bd68f84e0	12	Меланхолия	Elvira T	Громкие новинки Декабря 2	{Pop,Dance,Other}
bb3c1176-01bd-4327-9253-add309364574	2	Друг помоги мне	HENSY, Slame	Громкие новинки Декабря 2	{Pop,Dance,Other}
5d7f40c9-ed63-4398-9165-ad1b21d97d68	10	Аромат	Ramil'	Громкие новинки Декабря 2	{Pop,Dance,Other}

Рисунок 12 – Результат обращения к представлению playlist

Для аналитического бизнес-процесса «Формирование списка пользователей» создан запрос. Код запроса представлен в листинге 4.

Листинг 4 – Запрос для аналитического бизнес-процесса «Формирование списка пользователей»

```

SELECT "user".uuid,
       "user".login,
       "user".last_offline,
       "user".avatar_path,
       "user".birthday,
       "user".first_name,
       "user".middle_name,
       "user".last_name,
       "user".gender,
       "user".phone,
       count(soundtrack.*) AS soundtracks
FROM "user"
LEFT JOIN soundtrack ON soundtrack.cu = "user".uuid
GROUP BY "user".uuid
ORDER BY soundtracks DESC

```

LIMIT 200 OFFSET 0;

Результат работы запроса показан на рисунке 13.

1	9a8db856-16b3-469e-85d0-8f408b2441b7	aVoid	2021-06-03 13:17:29	/9a8db856-16b3-469e-85d0-8f408b2441b7/dokument	[NULL]	Павел
2	9c26e931-d6dd-4a13-b59b-71871479cc16	test	[NULL]	[NULL]	[NULL]	[NULL]
3	22ae0112-3900-4ceb-b74a-8f6dc4c0a04a	test	[NULL]	[NULL]	[NULL]	[NULL]
4	814d08b6-b886-11eb-9a34-7f0ea57a76dd	test	2021-06-02 11:42:35	[NULL]	[NULL]	test
5	f7eae3c-3643-4b2f-850b-6bf268652b8e	test2	2021-05-23 14:38:04	[NULL]	[NULL]	[NULL]

Рисунок 13 – Результат выполнения запроса

Разработано соответствующее данному запросу представление. Код представления находится в листинге 5.

Листинг 5 – Код представления для аналитического бизнес-процесса «Формирование списка пользователей»

```
CREATE OR REPLACE VIEW public.user_public
AS SELECT "user".uuid,
        "user".login,
        "user".last_offline,
        "user".avatar_path,
        "user".birthday,
        "user".first_name,
        "user".middle_name,
        "user".last_name,
        "user".gender,
        "user".phone,
        count(soundtrack.*) AS soundtracks
FROM "user"
        LEFT JOIN soundtrack ON soundtrack.cu = "user".uuid
ORDER BY soundtracks DESC
GROUP BY "user".uuid;
```

Результат обращения к данному представлению показан на рисунке 14.

	uuid	login	last_offline	avatar_path	birthday	first_name
1	9c26e931-d6dd-4a13-b59b-71871479cc16	test	[NULL]	[NULL]	[NULL]	[NULL]
2	22ae0112-3900-4ceb-b74a-8f6dc4c0a04a	test	[NULL]	[NULL]	[NULL]	[NULL]
3	814d08b6-b886-11eb-9a34-7f0ea57a76dd	test	2021-06-02 11:42:35	[NULL]	[NULL]	test
4	f7eae3c-3643-4b2f-850b-6bf268652b8e	test2	2021-05-23 14:38:04	[NULL]	[NULL]	[NULL]
5	9a8db856-16b3-469e-85d0-8f408b2441b7	aVoid	2021-06-03 13:17:29	/9a8db856-16b3-469e-85d0-8f408b2441b7/dokument	[NULL]	Павел

Рисунок 14 – Результат обращения к представлению user_public

В таблице 6 представлена спецификация созданных запросов и параметры группировки и выборки для данных запросов. Таблица 7 содержит спецификацию созданных представлений, их описание и параметры группировки. Все представления основаны на запросах, описанных ранее в таблице 6.

Таблица 6 – Спецификации запросов к базе данных

№	Содержание	Используемые таблицы	Параметры группировки	Параметры выбора данных
1	Список песен плейлиста, их средняя пользовательская оценка и количество оценок каждой песни.	soundtrack (Песня), feedback (Оценка с отзывом о песне)	Сортировка по средней пользовательской оценке.	Ограничение количества записей. Смещение выборки.
2	Список пользователей и количество загруженных ими песен.	user (Пользователь), feedback (Оценка с отзывом о песне)	Сортировка по количеству загруженных в систему песен.	Ограничение количества записей. Смещение выборки.

Таблица 7 – Спецификации представлений

№	Имя представления	Назначение	Используемые таблицы	Параметры группировки
1	playlist	Вывод упорядоченного списка всех песен плейлиста с добавлением столбцов средней пользовательской оценки и количества пользовательских оценок	soundtrack (Песня), feedback (Оценка с отзывом о песне)	Сортировка по средней пользовательской оценке.
2	user_public	Вывод упорядоченного списка всех пользователей с добавлением столбца количества загруженных пользователями песен	user (Пользователь), feedback (Оценка с отзывом о песне)	Сортировка по количеству загруженных в систему песен.

Созданные запросы и представления используются для реализации аналитических бизнес-процессов в рамках сервиса удаленного управления медиасистемой. Данные аналитические бизнес-процессы позволяют клиентам сервиса загружать актуальный плейлист системы, а также список пользователей, упорядоченный по определенным критериям и содержащие

дополнительную аналитическую информацию по загруженным пользователями песням.

3.1.2 Разработка функций и хранимых процедур

Для бизнес-процесса «Управлять плейлистом» созданы функции `insert_soundtrack`, `update_soundtrack`, `delete_soundtrack` и `update_playlist`. Код функций представлен в приложении Г.

Спецификация всех созданных для сервиса хранимых процедур и функций представлена в таблице 8. Данные функции будут использоваться в дальнейшем при реализации триггеров.

Таблица 8 – Спецификации функций и хранимых процедур

№	Имя функции (процедуры)	Назначение	Описание логики работы
1	<code>insert_soundtrack</code>	Добавление записи в таблицу <code>soundtrack</code>	Выполняет добавление записи в таблицу <code>soundtrack</code> . Используется для поддержания целостности значений атрибута <code>order</code> . Отменяет операцию при выходе значения <code>order</code> за границы допустимых значений. В случае совпадения значения <code>order</code> с существующей записью приоритет отдается создаваемой записи, записи с большим значением атрибута <code>order</code> сдвигаются.
2	<code>update_soundtrack</code>	Обновление записи в таблице <code>soundtrack</code>	Выполняет обновление записи в таблице <code>soundtrack</code> . Используется для поддержания целостности значений атрибута <code>order</code> . Отменяет операцию при выходе значения <code>order</code> за границы допустимых значений. В случае совпадения значения <code>order</code> с существующей записью приоритет отдается обновляемой записи, записи с большим значением атрибута <code>order</code> сдвигаются.
3	<code>delete_soundtrack</code>	Удаление записи из таблицы <code>soundtrack</code>	Выполняет удаление записи из таблицы <code>soundtrack</code> . Используется для поддержания целостности значений атрибута <code>order</code> . После удаления записи смещает значение <code>order</code> записей, следующих за удаленной, для сохранения упорядоченности.

Продолжение таблицы 8

№	Имя функции (процедуры)	Назначение	Описание логики работы
4	update_playlist	Обновление записи представления playlist	Выполняет обновление записи представления playlist. Используется для поддержания целостности значений атрибута order таблицы soundtrack. Отменяет операцию при выходе значения order за границы допустимых значений. В случае совпадения значения order с существующей записью приоритет отдается обновляемой записи, записи с большим значением атрибута order сдвигаются.

Полученные процедуры и функции представляют бизнес-логику, реализация которой необходима на уровне базы данных.

3.1.3 Разработка триггеров

Для бизнес-процесса «Управлять плейлистом» созданы триггеры insert_soundtrack_trigger, update_soundtrack_trigger, delete_soundtrack_trigger и update_playlist_trigger. Данные триггеры будут поддерживать логическую целостность атрибута order таблицы soundtrack, отвечающего за упорядоченность песен в плейлисте.

Таким образом, при попытке выполнения действие над песней, которое нарушит целостность атрибута, база данных заблокирует транзакцию [2].

Код триггеров представлен в приложении Д.

Триггер update_playlist_trigger используется для работы с представлением playlist.

Результат запроса представления показан на рисунке 15.

1	61aea7e0-b94c-413f-8468-303b31493e49	4	Big Money	Rakhim	Громкие новинки Декабря 2	{Pop,Dance,Other}
2	df92a5a8-cec0-4585-9010-7a9f7b33cb8e	5	Losing My Religion	Dj Antonio, Bertie Scott	Громкие новинки Декабря 2	{Pop,Dance,Other}
3	8059141c-de34-449f-a8ad-958c35cbf1e7	7	Чё там у хохлов	YARMAK	Громкие новинки Декабря 2	{Pop,Dance,Other}
4	cd5616a8-02cb-4517-81f6-6ce3b838829e	8	Тренировочный день	KYOK, Markul	Громкие новинки Декабря 2	{Pop,Dance,Other}
5	cb361d80-c3ae-43c0-b8da-49550ec05e8e	6	СТАТУС ВЛЮБЛЕНА	ХАННА	Громкие новинки Декабря 2	{Pop,Dance,Other}
6	f83e4381-1617-4151-83bb-11abd44c8a23	12	АУФ	Mia Boyka	Громкие новинки Декабря 2	{Pop,Dance,Other}
7	87d5575d-3320-4542-b717-4f3973f6a6a6	3	Высоко	ANIVAR	Громкие новинки Декабря 2	{Pop,Dance,Other}
8	f8ac10e9-3ee9-46c1-979a-6025ed8562bc	10	Dreams	David Guetta, MORTEN, Lanie Gardner	Громкие новинки Декабря 2	{Pop,Dance,Other}
9	2c1f097a-9b0f-4a49-bbb3-11efa28c4dc0	1	Reminds Me Of You	Juice WRLD, The Kid LAROI	Громкие новинки Декабря 2	{Pop,Dance,Other}
10	934a2f8e-955c-4797-a172-ef6bd68f84e0	11	Меланхолия	Elvira T	Громкие новинки Декабря 2	{Pop,Dance,Other}
11	bb3c1176-01bd-4327-9253-add309364574	2	Друг помоги мне	HENSY, Slame	Громкие новинки Декабря 2	{Pop,Dance,Other}
12	5d7f40c9-ed63-4398-9165-ad1b21d97d68	9	Аромат	Ramil'	Громкие новинки Декабря 2	{Pop,Dance,Other}

Рисунок 15 – Результат запроса к представлению playlist

Данное представление кроме полей сущности песни предоставляет возможность использовать информацию о количестве отзывов пользователей и рассчитывает среднюю пользовательскую оценку песни.

Для возможности через данное приложение работать напрямую с таблицей soundtrack был создан триггер update_playlist_trigger. Он срабатывает при обновлении данных в представлении. Триггер перенаправляет вышеуказанные транзакции в таблицу песен проводя дополнительные действия. Так, при обновлении песни через представление для обновляемой записи пользователя автоматически будет вызвана хранимая процедура обновления плейлиста.

Для демонстрации работы триггера выполним операцию обновления над представлением.

Список записей таблицы soundtrack перед выполнением операции обновления представления playlist представлен на рисунке 16.

	123 duration	123 order	123 sample_rate	123 bit_rate	abc file_format	abc comment	updt	ct	123 ver
1	165	2	44 100	320 000	mp3	[NULL]	2021-05-31 10:56:44	2021-05-24 21:50:33	47
2	225	3	44 100	320 000	mp3	[NULL]	2021-05-31 10:57:27	2021-05-24 00:21:15	30
3	132	4	44 100	320 000	mp3	[NULL]	2021-05-31 10:55:26	2021-05-24 19:54:07	69
4	248	5	44 100	320 000	mp3	[NULL]	2021-05-25 21:01:28	2021-05-24 21:51:51	35
5	156	6	44 100	320 000	mp3	[NULL]	2021-05-25 14:35:58	2021-05-24 19:59:49	35
6	166	7	44 100	320 000	mp3	[NULL]	2021-05-24 21:57:21	2021-05-24 20:03:06	41
7	194	8	44 100	320 000	mp3	ppp	2021-05-24 21:35:29	2021-05-24 17:36:13	28
8	150	9	44 100	320 000	mp3	[NULL]	2021-05-27 11:32:34	2021-05-24 22:14:48	16
9	204	10	44 100	320 000	mp3	[NULL]	2021-05-25 14:35:52	2021-05-24 00:24:40	34
10	169	11	44 100	320 000	mp3	1234	2021-05-31 18:54:16	2021-05-31 18:54:16	2
11	95	12	44 100	320 000	mp3	1234567	2021-05-27 14:06:59	2021-05-27 14:06:34	9
12	158	1	44 100	320 000	mp3	[NULL]	2021-06-02 11:56:52	2021-05-24 00:22:57	71

Рисунок 16 – Записи таблицы soundtrack перед обновлением

Обновление данных в представлении. Код выполняемого запроса представлен на рисунке 17.

Name	Value
Updated Rows	1
Query	update playlist set "order" = 2 where uuid = '934a2f8e-955c-4797-a172-ef6bd68f84e0'
Finish time	Thu Jun 03 22:12:08 MSK 2021

Рисунок 17 – Запрос обновления представления playlist

В результате будут изменены данные в таблице soundtrack, а также в самом представлении playlist. Результат показан на рисунках 18, 19.

	123 order	123 sample_rate	123 bit_rate	abc file_format	abc comment	updt	ct	123 ver	uuid
1	11	44 100	320 000	mp3	[NULL]	2021-05-25 14:35:52	2021-05-24 00:24:40	35	f8ac10e9-3ee9-46c1-9
2	10	44 100	320 000	mp3	[NULL]	2021-05-27 11:32:34	2021-05-24 22:14:48	17	5d7f40c9-ed63-4398-5
3	9	44 100	320 000	mp3	ppp	2021-05-24 21:35:29	2021-05-24 17:36:13	29	cd5616a8-02cb-4517-8
4	8	44 100	320 000	mp3	[NULL]	2021-05-24 21:57:21	2021-05-24 20:03:06	42	8059141c-de34-449f-a
5	7	44 100	320 000	mp3	[NULL]	2021-05-25 14:35:58	2021-05-24 19:59:49	36	cb361d80-c3ae-43c0-b
6	6	44 100	320 000	mp3	[NULL]	2021-05-25 21:01:28	2021-05-24 21:51:51	36	df92a5a8-cec0-4585-9
7	5	44 100	320 000	mp3	[NULL]	2021-05-31 10:55:26	2021-05-24 19:54:07	70	61aea7e0-b94c-4131-8
8	12	44 100	320 000	mp3	1234567	2021-05-27 14:06:59	2021-05-27 14:06:34	9	f83e4381-1617-4151-8
9	4	44 100	320 000	mp3	[NULL]	2021-05-31 10:57:27	2021-05-24 00:21:15	31	87d5575d-3320-4542-
10	3	44 100	320 000	mp3	[NULL]	2021-05-31 10:56:44	2021-05-24 21:50:33	48	bb3c1176-01bd-4327-
11	2	44 100	320 000	mp3	1234	2021-05-31 18:54:16	2021-05-31 18:54:16	3	934a2f8e-955c-4797-a
12	1	44 100	320 000	mp3	[NULL]	2021-06-02 11:56:52	2021-05-24 00:22:57	71	2c1f097a-9b0f-4a49-b

Рисунок 18 – Запись в таблице soundtrack изменена

	uuid	123 order	abc title	abc artist	abc album	genre
1	61aea7e0-b94c-413f-8468-303b31493e49	5	Big Money	Rakhim	Громкие новинки Декабря 2	(Pop,Dance,Other)
2	df92a5a8-cec0-4585-9010-7a9f7b33cb8e	6	Losing My Religion	Dj Antonio, Bertie Scott	Громкие новинки Декабря 2	(Pop,Dance,Other)
3	8059141c-de34-449f-a8ad-958c35cbf1e7	8	Че там у хохлов	YARMAK	Громкие новинки Декабря 2	(Pop,Dance,Other)
4	cd5616a8-02cb-4517-81f6-6ce3b838829e	9	Тренировочный день	КУОК, Markul	Громкие новинки Декабря 2	(Pop,Dance,Other)
5	cb361d80-c3ae-43c0-b8da-49550ec05e8e	7	СТАТУС ВЛЮБЛЕНА	ХАННА	Громкие новинки Декабря 2	(Pop,Dance,Other)
6	f83e4381-1617-4151-83bb-11abd44c8a23	12	АУФ	Mia Boyka	Громкие новинки Декабря 2	(Pop,Dance,Other)
7	87d5575d-3320-4542-b717-4f3973f6a6a6	4	Высоко	ANIVAR	Громкие новинки Декабря 2	(Pop,Dance,Other)
8	f8ac10e9-3ee9-46c1-979a-6025ed8562bc	11	Dreams	David Guetta, MORTEN, Lanie Gardner	Громкие новинки Декабря 2	(Pop,Dance,Other)
9	2c1f097a-9b0f-4a49-bbb3-11efa28c4dc0	1	Reminds Me Of You	Juice WRLD, The Kid LAROI	Громкие новинки Декабря 2	(Pop,Dance,Other)
10	934a2f8e-955c-4797-a172-e6bd68f84e0	2	Меланхолия	Elvira T	Громкие новинки Декабря 2	(Pop,Dance,Other)
11	bb3c1176-01bd-4327-9253-add309364574	3	Друг помоги мне	HENSY, Slame	Громкие новинки Декабря 2	(Pop,Dance,Other)
12	5d7f40c9-ed63-4398-9165-ad1b21d97d68	10	Аромат	Ramil'	Громкие новинки Декабря 2	(Pop,Dance,Other)

Рисунок 19 – Состояние представления playlist после обновления

Спецификация триггеров приведена в таблице 9.

Таблица 9 – Спецификации триггеров

№	Назначение триггера	Вид триггера	Событие триггера	Таблица триггера	Описание работы триггера
1	Триггер insert_soundtrack_trigger предназначен для контроля целостности атрибута order.	BEFORE	INSERT	soundtrack	Данный триггер активируется перед выполнением операции INSERT для таблицы soundtrack. Проверяет, удовлетворяет ли условиям целостности значение атрибута order у добавляемой записи. В случае выявления нарушения целостности отменяет транзакцию. Если значение атрибута order уже существует в таблице, приоритет отдается создаваемой записи, остальные записи сдвигаются.
2	Триггер update_soundtrack_trigger предназначен для контроля целостности атрибута order.	BEFORE	UPDATE	soundtrack	Данный триггер активируется перед выполнением операции UPDATE для таблицы soundtrack. Проверяет, удовлетворяет ли условиям целостности значение атрибута order у добавляемой записи. В случае выявления нарушения целостности отменяет транзакцию. Если значение атрибута order уже существует в таблице, приоритет отдается обновляемой записи, остальные записи сдвигаются.

Продолжение таблицы 9

№	Назначение триггера	Вид триггера	Событие триггера	Таблица триггера	Описание работы триггера
3	Триггер delete_soundtrack_trigger предназначен для контроля целостности атрибута order.	AFTER	DELETE	soundtrack	Триггер срабатывает после выполнения операции DELETE для таблицы soundtrack. Восстанавливает целостность атрибута order, смещая значения этого атрибута у следующих за удаленной записей.
4	Триггер update_playlist_trigger предназначен для работы с представлением playlist. Триггер позволяет через работу с представлением выполнять операции обновления с таблицей soundtrack.	INSTEAD OF	UPDATE	playlist (Представление)	Данный триггер переопределяет операцию UPDATE над представлением playlist. При операции обновления данных в представлении обработчик триггера перенаправляет параметры с соответствующий запрос к таблице soundtrack.

Полученные триггеры выполняют автоматизацию определенных аспектов бизнес-процессов на уровне базы данных.

3.2 Разработка серверного приложения

В архитектуре данной системы серверное приложение играет роль посредника между клиентским web приложением и базой данных, а также серверное приложение осуществляет управление плеером AIMP. Серверное приложение выполнено на основе RESTful технологий Java Web Services. Веб сервисы предлагают удобный интерфейс передачи данных через прикладной протокол прикладного уровня HTTP с использованием формата данных JSON.

Кроме этого, технологии Java Web Services предоставляют инструменты для преобразования данных из формата JSON в объекты – модели. Преобразованием можно управлять по средствам классов-адаптеров.

Серверная программа представляет собой набор классов – сервисов, автоматически определяемых технологией Java Web Service. Каждый такой сервис служит входной точкой в программу.

Исходя из выбранных технологий и требований, для серверной программы выбрана следующая архитектура. Все классы приложения делятся на три слоя:

Слой сервисов. Определяет точки входа в программу по средствам обращения к ним через HTTP запрос. Метод запроса, URL путь и возвращаемые данные определяются в сервисах через нотации из библиотеки javax.ws.rs.

Пример использования нотаций для объявления веб сервиса представлен в листинге 6.

Листинг 6 – Пример объявления веб сервиса

```
@GET
@Path("playpause")
@Authorized({Roles.Registered})
@Produces({MediaType.APPLICATION_JSON})
public Response playpause() {
    try {
        remoteApi.executePlaypause();
        AIMPPPropertyDto res = remoteApi.getPlayerState();

        return Response.ok(res).build();
    } catch (Exception e) {
        logger.log(Level.SEVERE, e.getMessage(), e);
        return Response.serverError().entity(new
StatusMessage(e)).build();
    }
}
```

Слой фасадов. Предоставляет методы доступа к хранилищам данных, например, к базе данных или к файлам. Для запросов к базе данных использована библиотека Java Persistence. Библиотека включает в себя Object-Relational Mapping (ORM) технологию и предоставляет три возможных варианта доступа к базе данных:

- прямой запрос в базу данных, используя язык SQL;
- запрос на языке Java Persistence Query Language (JPQL);
- запрос через цепочку вызова методов ORM-библиотеки.

Язык JPQL является расширением языка SQL, поставляемым вместе с библиотекой Java Persistence. Данный язык позволяет использовать в запросе сущности из контекста Java-программы.

Пример запроса на языке JPQL с использованием Java Persistence представлен в листинге 7.

Листинг 7 – Пример запроса на языке JPQL

```
getEntityManager()
.createQuery("SELECT u FROM User u WHERE u.email = :email and u.shapass
= :password", User.class)
.setParameter("email", email.toLowerCase())
.setParameter("password", password)
.getResultList();
```

Слой моделей. Выполняет сразу несколько функций: является моделями данных предметной области, управляет отражением сущностей базы данных в Java-объектах, управляет преобразованием объектов в JSON-строку и обратно.

Пример модели представлен в листинге 8.

Листинг 8 – Пример объявления модели данных

```
@Entity
@Table(name = "playlist")
@XmlRootElement
@Converter(name = "uuidConverter", converterClass =
UuidPgConverter.class)
@Struct(name = "playlist")
public class Playlist extends AbstractEntity<PlaylistDto> {
    @Column(name = "\"order\"")
    private Integer order;
    @Column(name = "is_active")
    private Boolean isActive;
    @Column(name = "title")
    private String title;
    @Column(name = "artist")
    private String artist;
```

```

@Column(name = "album")
private String album;
@ElementCollection
@Array(databaseType = "varchar")
@Column(name = "genre")
private List<String> genre;
@Column(name = "year")
private String year;
@Column(name = "feedbacks")
private Integer feedbacks;
@Column(name = "avg_rating")
private Integer avgRating;

```

Как можно видеть, в модели использовано множество атрибутов.

Исходный код серверного приложения сервиса представлен в репозитории [13]. На рисунках 20 – 22 представлены диаграммы классов соответствующие различным слоям системы.

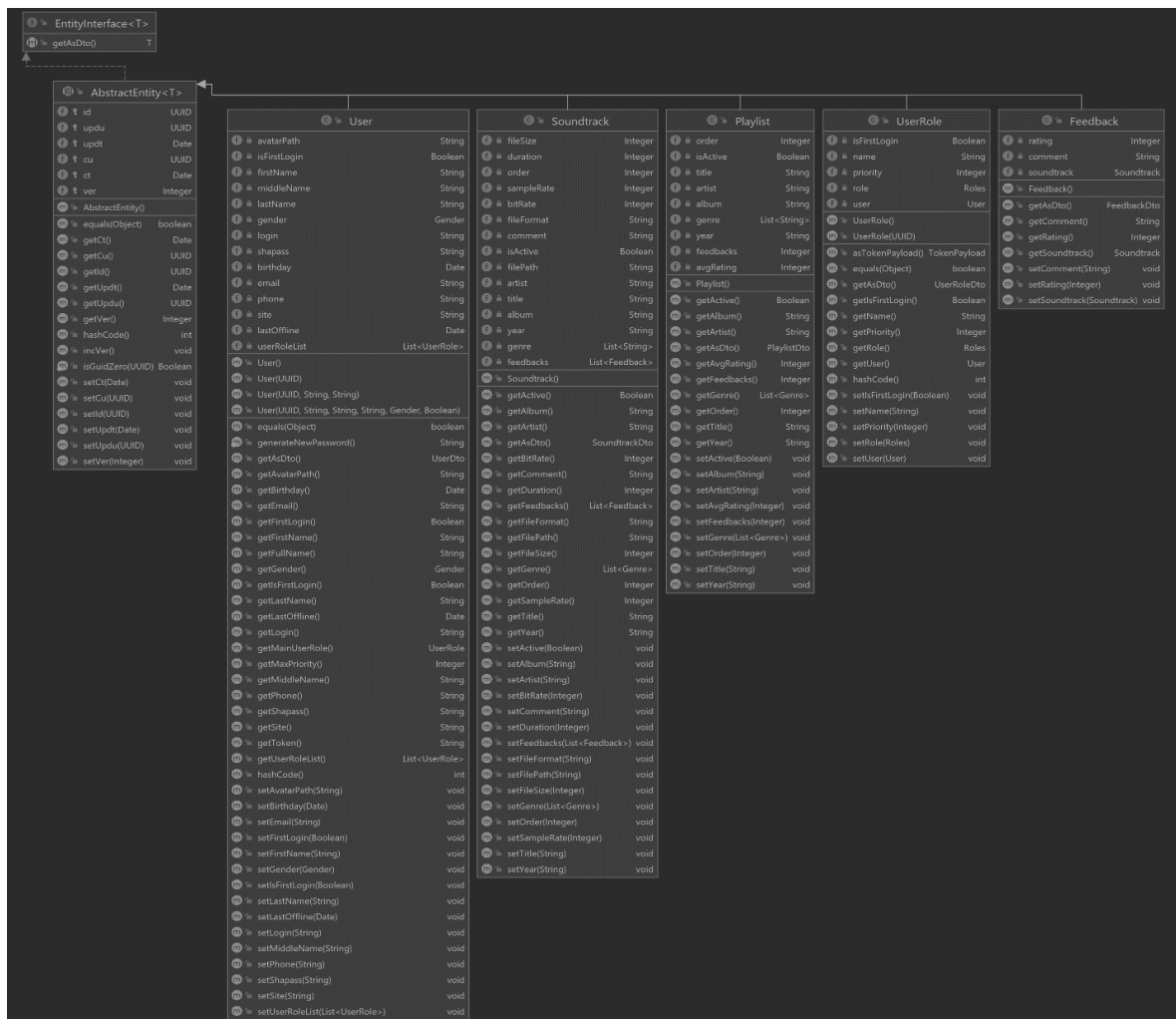


Рисунок 20 – Диаграмма классов слоя моделей

На рисунке 20 изображена диаграмма классов слоя моделей. Каждый из представленных на диаграмме классов реализует Java сущность, которая в результате работы библиотеки Java Persistence преобразуется к обычным SQL сущностям, которые могут храниться в базе данных.

Преобразование Java-моделей к SQL-сущностям выполняют классы фасады, соответствующие хранимым в базе данных сущностям. Классы данного слоя также реализуют базовые запросы к базе данных: создание, удаление, изменение сущностей. Диаграмма классов данного слоя представлена на рисунке 21.

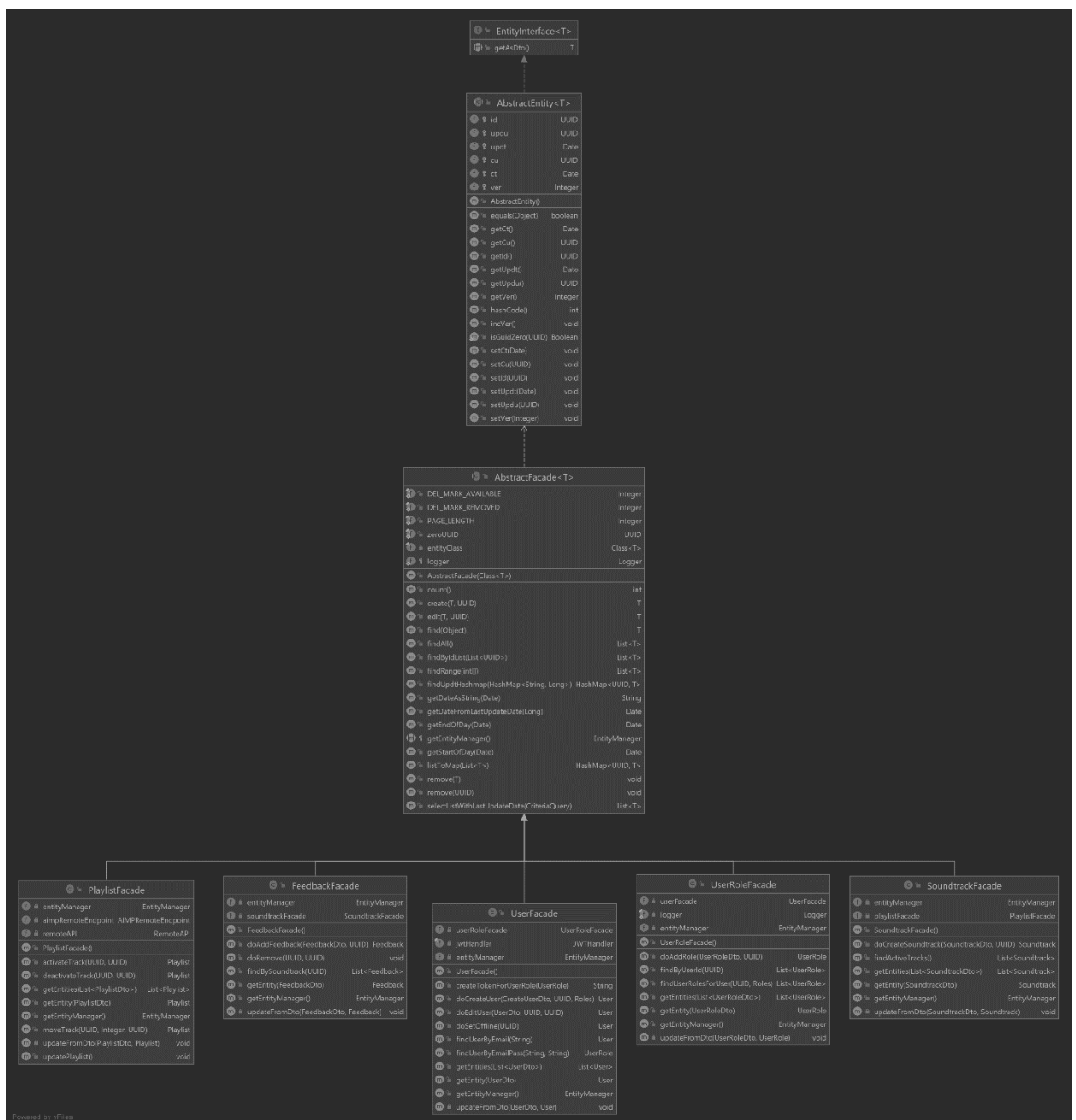


Рисунок 21 – Диаграмма классов слоя фасадов

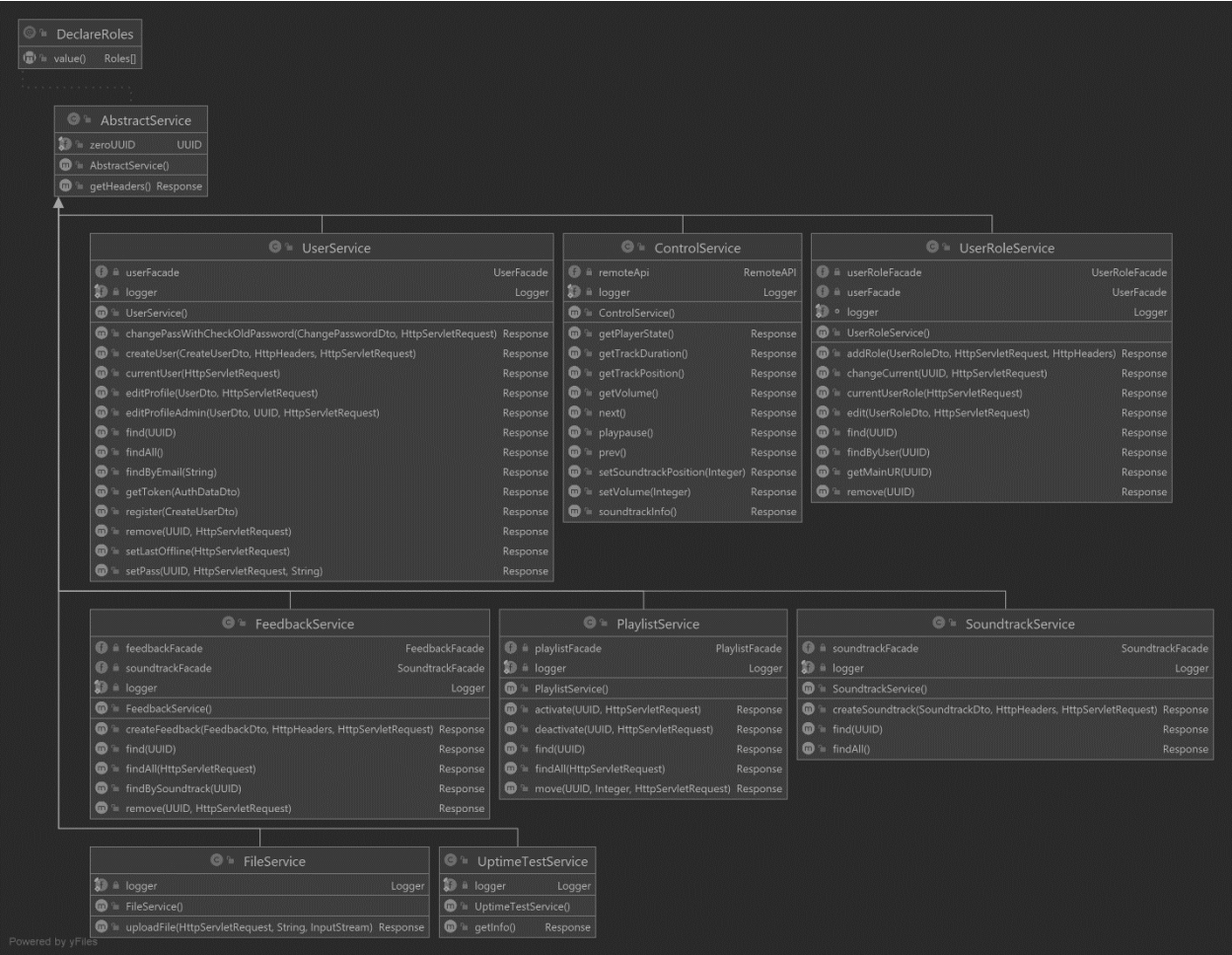


Рисунок 22 – Диаграмма классов слоя сервисов

Диаграммы созданы с использованием языка UML [4]. На каждой диаграмме классы сгруппированы по признаку принадлежности их к слою архитектуры приложения. Таким образом, классы UserService, FeedbackService принадлежат к сервисам. Классы UserFacade, FeedbackFacade принадлежат к фасадам. Классы User, Feedback являются моделями.

Описание классов, присутствующих на диаграммах классов представлено в таблице 10.

Таблица 10 – Спецификация классов приложения

Название класса	Описание
EntityInterface	Интерфейс сущностей. Описывает методы формирования DTO объектов из сущностей.
AbstractEntity	Абстрактная сущность. Содержит общие для всех сущностей поля и методы.
User	Модель пользователя. Содержит UUID, логин, пароль, имя, дату рождения, телефонный номер пользователя.
UserRole	Модель роли пользователя. Содержит UUID, название роли и ссылку на пользователя.

Продолжение таблицы 10

Название класса	Описание
Soundtrack	Модель загруженной песни. Содержит UUID, информацию о файле, описание песни и данные о загрузившем пользователе.
Playlist	Модель списка воспроизведения. Содержит UUID и основную информацию о песнях, включенных в список воспроизведения.
Feedback	Модель отзывов о песнях. Содержит UUID, отзыв, пользовательскую оценку и ссылку на загруженную песню.
AbstractFacade	Абстрактный фасад. Содержит объект EntityManager, предоставляющий доступ к базе данных. Реализует базовые запросы к базе данных.
UserFacade	Фасад пользователя. Предоставляет доступ к запросам в базу данных для авторизации и регистрации пользователя. Содержит ссылку на объект типа EntityManager, предоставляющий доступ к базе данных.
UserRoleFacade	Фасад ролей пользователя. Предоставляет доступ к запросам в базу данных необходимым для управления ролями пользователя. Содержит ссылку на объект типа EntityManager, предоставляющий доступ к базе данных.
SoundtrackFacade	Фасад загруженных песен. Предоставляет доступ к запросам в базу данных для создания и редактирования загруженных песен. Содержит ссылку на объект типа EntityManager, предоставляющий доступ к базе данных.
PlaylistFacade	Фасад списка воспроизведения. Реализует запросы к базе данных, необходимые для управления списком воспроизведения. Содержит ссылку на объект типа EntityManager, предоставляющий доступ к базе данных.
FeedbackFacade	Фасад отзывов о песнях. Предоставляет доступ к запросам в базу данных для создания и редактирования пользовательских отзывов о песнях. Содержит ссылку на объект типа EntityManager, предоставляющий доступ к базе данных.
DeclareRoles	Аннотация-поставщик доступных ролей для разграничения доступа к сервисам приложения.
AbstractService	Абстрактный сервис. Содержит реализацию базовых интерфейсов и наследуемых классов. Также предоставляет систему ролей, разграничивающую доступ к остальным сервисам приложения.
UserService	Сервис пользователя. Содержит методы авторизации и регистрации пользователя доступные для вызова через HTTP запрос. Содержит ссылку на соответствующий фасад пользователя.
UserRoleService	Сервис ролей пользователя. Содержит все методы, необходимые для управления ролями пользователя. Методы доступны для вызова через HTTP запрос. Содержит ссылку на соответствующий фасад ролей пользователя.
SoundtrackService	Сервис загруженных песен. Содержит методы для загрузки и редактирования загруженных песен. Методы доступны для вызова через HTTP запрос. Содержит ссылку на соответствующий фасад загруженных песен.

Окончание таблицы 10

Название класса	Описание
PlaylistService	Сервис списка воспроизведения. Данный сервис содержит методы управления списком воспроизведения. Все методы, реализованные в сервисе доступны для вызова по средствам HTTP запроса. Содержит ссылку на соответствующий фасад списка воспроизведения.
FeedbackService	Сервис отзывов о песнях. Данный сервис содержит доступные к вызову через HTTP запрос методы, позволяющие оставлять новые отзывы, а также управлять уже оставленными ранее отзывами о песнях. Сервис содержит ссылку на соответствующий ему фасад отзывов о песнях.
ControlService	Сервис управления плеером. Сервис содержит все методы, необходимые для управления состоянием плеера. Методы, реализуемые данным сервисом, доступны для вызова через HTTP запрос. Сервис содержит ссылку на API управления плеером.
FileService	Сервис файлов. Данный сервис реализует методы работы с файлами. Поддерживаемые методы доступны для вызова по средствам HTTP запроса. Сервис содержит ссылку на API для работы с файлами.
UptimeService	Сервис состояния приложения. Позволяет по средствам HTTP запроса получить текущее состояние серверного приложения.

Спецификация доступных в серверном приложении Java RESTful сервисов, а также краткое описание их работы и типы обрабатываемых данных представлены в приложении Ж в таблице Ж.1.

Таким образом, данные сервисы полностью обеспечивают связь клиентского приложения с серверным приложением и базой данных.

3.3 Разработка клиентской части приложения

Пользовательский интерфейс в React реализуется в виде компонентов и хуков языка разметки JSX. В коде которого компонента содержится логика обновления пользовательского интерфейса этого компонента, данные, необходимые данному компоненту для рендера интерфейса и обработчики событий. Хук представляет собой упрощенную версию компонента лишенную UI. Хук выполняет лишь обработку данных. Кроме того, компоненты обладают возможностью обрабатывать изменения состояния всего приложения и других компонентов пользовательского интерфейса.

Во время проектирования веб приложения было принято решение использовать библиотеку React Router 6. Данная библиотека позволяет структурировать в удобном формате логику навигации между компонентами приложения. Основным инструментом библиотеки является дерево навигации – специальный JS объект, либо же JSX компонент, с описанием связи компонентов интерфейса с URL браузера. Структуру данного дерева можно изменять «на лету», что позволяет изменять роутинг для пользователя в соответствии с изменением его роли в рамках сессии. Используя данное дерево навигации, библиотека React Router 6 позволяет активировать заданные переходы между компонентами. Пример задания элемента дерева навигации приведен в листинге 9.

Листинг 9 – Пример задания элемента дерева навигации

```
{name: "UsersPage", component: <UsersPage/>, visible: true}
```

Приложение использует специальный компонент под названием Routes для связи компонентов с URL браузера и вспомогательный класс RouterProvider для создания дерева навигации для текущей роли пользователя.

Иерархия компонентов приложения для ролей системы представлена на рисунках 21 – 23. Пользователь с ролью «Администратор» также наследует роутинг роли «Пользователь».

```
export const GuestRoutes = {
  parentRoutes: null,
  routes: [
    {name: "LoginPage", component: <LoginPage/>, visible: true},
  ]
}
```

Рисунок 21 – Иерархия компонентов для роли «Гость»

```
export const RegisteredRoutes = {
  parentRoutes: null,
  routes: [
    {name: "LoginPage", component: <LoginPage/>, visible: true},
    {name: "ProfilePage", component: <ProfilePage/>, visible: true},
  ]
};
```

Рисунок 22 – Иерархия компонентов для роли «Пользователь»

```
export const AdminRoutes = {
  parentRoutes: RegisteredRoutes,
  routes: [
    {name: "UsersPage", component: <UsersPage/>, visible: true},
  ]
};
```

Рисунок 23 – Иерархия компонентов для роли «Администратор»

Описание компонентов страниц приложения, пути к этим страницам и доступные роли представлены в таблице 11. Исходный код клиентского приложения представлен в репозитории [14].

Таблица 11 – Описание компонентов страниц приложения

Название	URI страницы	Роли	Описание
LoginPage	/	«Гость»	Страница авторизации. Отображается не авторизованным пользователям при запуске приложения. На странице присутствует форма авторизации, а также модальное окно авторизации. Шаблон страницы – MainTemplateNoAuth.js
MainPage	/	«Пользователь», «Администратор»	Стартовая страница авторизованных пользователей. На странице отображаются элементы управления плеером и плейлистом. Шаблон страницы – MainTemplate.js
ProfilePage	/profile	«Пользователь», «Администратор»	Страница личного кабинета пользователя. Содержит информацию об авторизованном пользователе и предоставляет возможность изменения личных данных. Шаблон страницы – MainTemplate.js
UsersPage	/users	«Администратор»	Страница списка пользователей, доступная только администраторам сервиса. На странице представлен список всех пользователей системы, а также предоставляется возможность их модерации.
NotFoundPage	*	«Гость», «Пользователь», «Администратор»	Компонент страница, отображающийся при переходе по несуществующему в системе URI.

Описанные выше компоненты страницы были спроектированы с помощью UI элементов библиотеки Ant.Design React. Стилизация компонентов страниц проводилась с помощью библиотеки-плагина babel.

Ниже представлено детальное описание страниц.

Рассмотрим страницу авторизации. Данная страница отображается по умолчанию пользователю с ролью «Гость». Роль «гостя» получает любой не авторизованный в системе пользователь. Страница авторизации содержит форму авторизации, включающую поля для ввода email и пароля. Данная форма используется для авторизации пользователей в сервисе. Также, на странице авторизации находится модальное окно с формой регистрации. Форма регистрации включает в себя поля для ввода email, логина пользователя и пароля. Форма регистрации используется для регистрации новых пользователей в системе. Данная страница использует шаблон MainTemplateNoAuth.js. Данный шаблон поставляет основные элементы управления – хедер и футер страницы, а также разметку страницы. Элементы страницы продемонстрированы на рисунке 24 и 25.

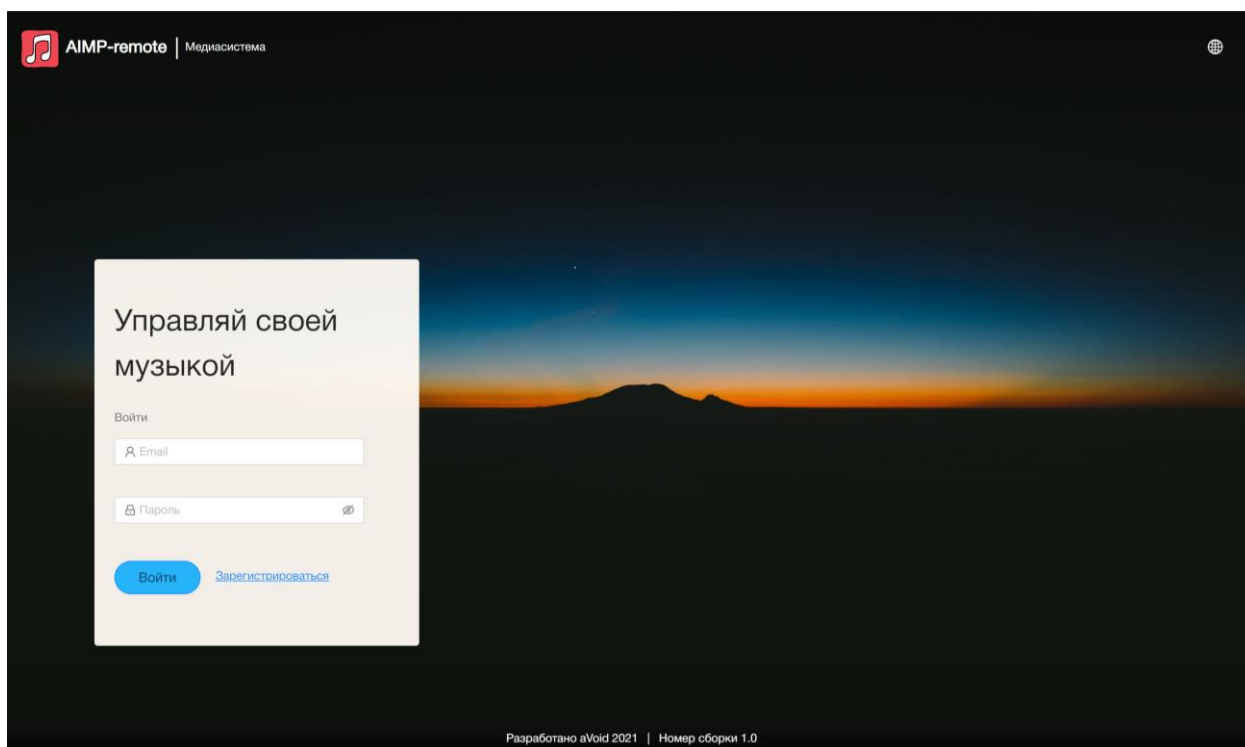


Рисунок 24 – Страница авторизации

Регистрация

* Email

* Login

* Пароль

* Подтверждение пароля

Отмена

OK

Рисунок 25 – Модальное окно регистрации

Рисунок 26 отображает главную страницу сервиса.

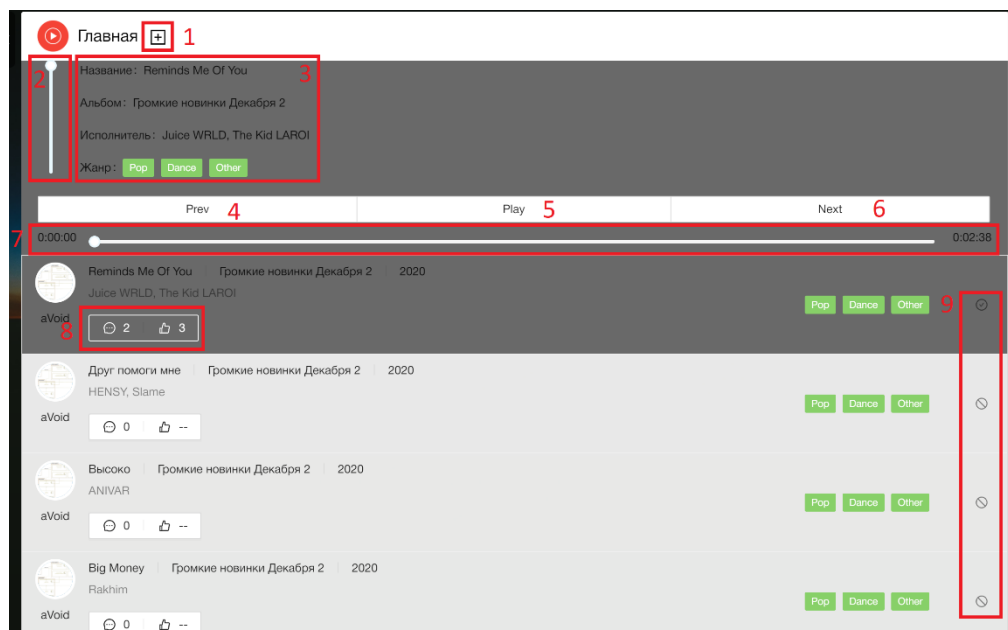


Рисунок 26 – Главная страница

Цифрой 1 на рисунке отмечена кнопка, открывающая модальное окно загрузки нового трека в систему. Модальное окно загрузки нового трека

содержит поле для загрузки файла песни и текстовое поле для комментария пользователя. Цифрой 2 отмечен ползунок управления громкостью воспроизведения плеера. Он изменяется от 0 до 100%. Цифра 3 показывает обновляемое поле, содержащее основную информацию о воспроизводимой песне. Цифрами 4 – 6 отмечены основные кнопки управления плеером: переключение песни, а также изменение состояния воспроизведения. Цифра 7 показывает ползунок управляющий позицией воспроизведения песни. Слева и справа от ползунка находятся текстовые поля, содержащие текущую позицию воспроизведения песни, а также ее продолжительность. Цифрой 8 отмечена кнопка, открывающие модальное окно, содержащее подробную информацию о выбранной песне, а также интерфейс добавления отзывов и оценок пользователей о песне. Цифра 9 отмечены кнопки управления состоянием песен. Они используются для включения и исключения песен из плейлиста плеера. Также элементы плейлиста – песни поддерживают перемещение по средствам drag and drop для изменения позиции песни в плейлисте.

На рисунке 27 изображена страница профиля пользователя. Данная страница доступна всем авторизованным пользователям. На странице присутствуют поля, содержащие личные данные текущего пользователя, а также предоставляется возможность их изменить. Также присутствует поле для установки аватара профиля.

Рисунок 27 – Страница профиля пользователя

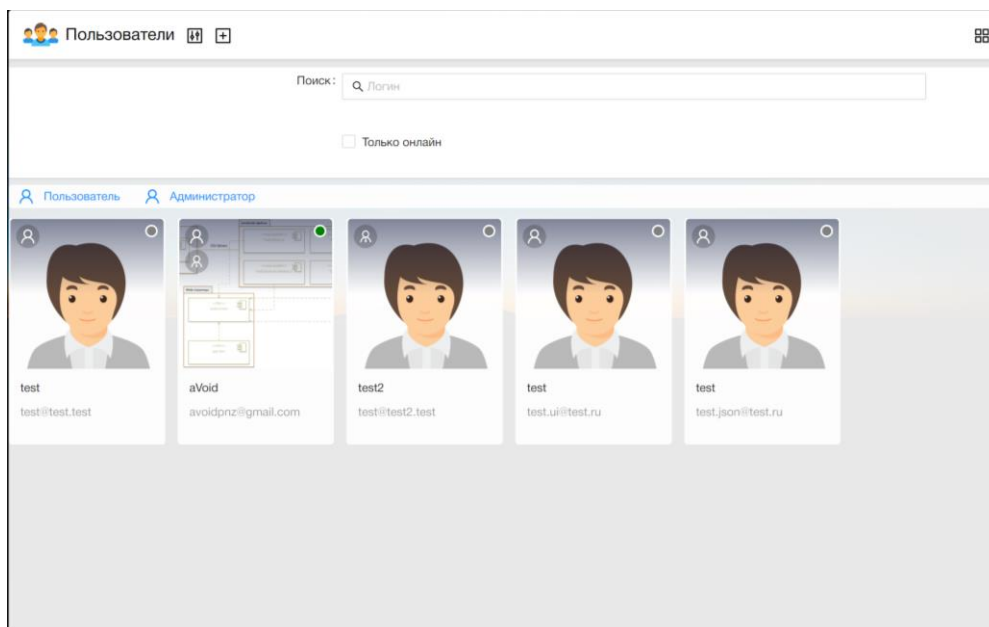


Рисунок 28 – Страница списка пользователей

Рисунок 28 показывает страницу списка пользователей. На данной странице показаны все пользователи, зарегистрированные в системе. Для удобного поиска на странице расположен фильтр, который позволяет отображать только пользователей, находящихся онлайн, а также искать их по соответствию логину. Из средств модерации данная страница позволяет создавать новых пользователей, удалять существующих пользователей из системы, управлять ролями пользователей, а также редактировать и просматривать личные данные пользователей. Данная страница доступна только администраторам сервиса.

3.4 Тестирование программного сервиса уделенного управления

3.4.1 Выбор режима тестирования

Для выбора режима тестирования созданной информационной системы, рассмотрим основные подходы к тестированию программного обеспечения и проанализируем их применимость для данного программного продукта.

Модульное тестирование (Unit tests) – метод низкоуровневого тестирования отдельных модулей или классов программы на правильность выполнения тех или иных функций. Выполняется автоматически, однако существенным минусом является необходимость ручного создания тестов по

средствам программного кода. Кроме того, модульное тестирование может не выявить некоторые типы ошибок, например ошибки в содержании данных.

Тестируемая система состоит из двух программных компонентов, что может потребовать написание отдельных модульных тестов для каждого из них, так как модульные тесты не принимают во внимание взаимодействие на уровне системы.

Системное тестирование – это высокоуровневая оценка работоспособности программы или системы в целом. При тестировании система рассматривается по принципу «черного ящика». Данный подход требует создание сложных систем автоматизации, а для небольших программ и информационных архитектур, системное тестирование без автоматизации равносильно функциональному тестированию. Следовательно, данный вид тестирования не подходит для созданной системы.

Интеграционное тестирование – высокоуровневое тестирование, направленное на проверку интеграции модулей системы и их взаимодействия между собой, а также на интеграцию подсистем в одну общую систему. Модули тестируемой системы связаны по средствам HTTP запросов, что является универсальным методом взаимодействия компонентов через сеть. Данная модель связи обладает низким уровнем интеграции и не требует отдельного тестирования связей между компонентами. Ошибки связей между компонентами могут быть покрыты функциональными тестами.

Тестирование производительности, включая нагрузочное и стресс-тестирование. Позволяет оценить работу системы под нагрузкой больших массивов данных и большого количества единовременных запросов. Из недостатков стоит отметить, что данный вид тестирования требует использования дополнительных программных средств для тестирования на отказоустойчивость высоконагруженных систем. На текущей итерации разработки тестируемой системы данными параметрами можно пренебречь. Кроме того, данный вид тестирования не позволяет оценить правильность работы программы с точки зрения функциональных требований.

Юзабилити тестирование позволяет получить данные о качестве пользовательского интерфейса, о его уровне удобства и эргономики с точки зрения пользователя. Данный вид тестирования не подходит для текущего этапа разработки, так как подразумевает, что функции программы должны быть уже протестированы и отлажены.

Функциональное тестирование – вид тестирования, направленный на проверку функциональных требований программы или системы. Может проводиться как в режиме «черного ящика», так и в режиме «белого ящика». В данном случае тестирование полностью имитирует фактическое использование системы и может быть выполнено вручную. Из преимуществ стоит отметить, то, что тестирование может проводиться на ранних этапах разработки программного продукта, над только частично законченными продуктами. Следовательно, выявленные ошибки могут быть исправлены на одном этапе с тестированием.

Перечисленные преимущества функционального тестирования позволяют сделать вывод о том, что применение данного типа тестирования может максимально оперативно дать информацию о стабильности работы системы и выполнении всех функциональных требований.

3.4.2 Тестирование базы данных

Тестирование разработанной базы данных было приведено с помощью среды DBeaver. Выполненные функциональные тесты базы данных представлены в таблице 12.

Таблица 12 – Результаты тестирования базы данных

№	Назначение теста	Описание теста	Результаты теста
1	Тестирование аналитического запроса выборки песен, упорядоченной по средней оценке пользователей	Выполнение запроса Select soundtrack.*,count(feedback.soundtrack) as feedbacks,avg(feedback.rating) as avg_rating from soundtrack left join feedback on soundtrack.uuid = feedback.soundtrack group by soundtrack.uuid order by avg_rating desc	Тест пройден. Выборка песен отсортирована по средней оценке пользователей. Результат показан на рисунке Е.1.

Продолжение таблицы 12

№	Назначение теста	Описание теста	Результаты теста
2	Тестирование аналитического запроса выборки пользователей упорядоченных по количеству загруженных песен	Выполнение запроса SELECT "user".uuid,"user".login,"user".last_offline,"user".avatar_path,"user".phone,count(soundtrack.*) AS soundtracks FROM "user" LEFT JOIN soundtrack ON soundtrack.cu = "user".uuid GROUP BY "user".uuid order by soundtracks desc	Тест пройден. Выборка пользователей отсортирована по количеству загруженных песен. Результат показан на рисунке Е.2.
3	Тестирование представления playlist	Выполнение запроса SELECT * FROM playlist	Тест пройден. Представление работает правильно. Результат показан на рисунке Е.3.
4	Тестирование представления user_public	Выполнение запроса SELECT * FROM user_public	Тест пройден. Представление работает правильно. Результат показан на рисунке Е.4.
5	Тестирование триггера insert_soundtrack_trigger	Производится попытка добавить запись в таблицу soundtrack insert into soundtrack (bit_rate, is_active, file_path, title, "order", ver, file_format) values (100, true, 'test', 'test', 1, 0, 'test');	Тест пройден. Операция INSERT перехвачена триггером, выполнена вставка записи в таблицу soundtrack. Рисунок Е.5.
6	Тестирование триггера update_playlist_trigger	Производится попытка изменить запись представления playlist update playlist set "order" = 6 where uuid = '9c671f00-c2b9-11eb-a5c8-fb3579c7300c'	Тест пройден. Операция UPDATE перехвачена триггером. Запись в таблице soundtrack изменена. Рисунок Е.6.
7	Тестирование триггера delete_soundtrack_trigger	Производится попытка удалить запись из таблицы soundtrack	Тест пройден. Операция DELETE перехвачена триггером. Запись из таблицы soundtrack удалена. Рисунок Е.7.

Результаты тестирования представлены на скриншотах из среды DBeaver в приложении Е.

Тестирование показывает стабильную работу базы данных на данном наборе тестов.

3.4.3 Тестирование серверного приложения

3.4.3.1 Тестирование веб-сервисов

Тестирование веб-сервисов, предоставляемых созданным серверным приложением, проводилось с использованием программы Advanced REST Client, которая позволяют отправлять любые HTTP запросы и получать ответы. Согласно парадигме REST-технологий, программа ведет обмен информацией с сервером по стандарту JSON. Тестирование веб-сервисов серверного приложения представлено в таблице 13.

Таблица 13 – Результаты тестирования веб-сервисов

Веб-сервис	Описание теста	Результат
user/getToken	HTTP запрос POST Body: { "email": "avoidpnz@gmail.com", "password": "dff3f9f8bce8f7c9de2a373d233020d36c10d265275d7e29c027d30a3f95c058" }	Тест пройден. Получен токен пользователя. Рисунок Е.8.
user/currentUser	HTTP запрос GET Header: { "token": "eyJhbGciOiJSUzI1NiJ9..." }	Тест пройден. Получен текущий пользователь. Рисунок Е.9.
user/register	HTTP запрос POST Body: { "email": "test.ui@test.ru", "login": "test", "shapass": "9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08" }	Тест пройден. Получен результат регистрации. Рисунок Е.10.
userRole/findByUser/{id}	HTTP запрос GET Header: { "token": "eyJhbGciOiJSUzI1NiJ9..." } URL: userRole/findByUser/9c26e931-d6dd-4a13-b59b-71871479cc16	Тест пройден. Получен список ролей пользователя.
feedback/bySoundtrack/{soundtrackId}	HTTP запрос GET Header: { "token": "eyJhbGciOiJSUzI1NiJ9..." } URL: feedback/bySoundtrack/5d7f40c9-ed63-4398-9165-ad1b21d97d68	Тест пройден. Получен список отзывов пользователей для песни.
feedback/addFeedback	HTTP запрос POST Header: { "token": "eyJhbGciOiJSUzI1NiJ9..." } Body: { "soundtrack": "2c1f097a-9b0f-4a49-bbb3-11efa28c4dc0", "comment": "Test", "rating": 4 }	Тест пройден. Отзыв пользователя добавлен к песне.

Продолжение таблицы 13

Веб-сервис	Описание теста	Результат
playlist	HTTP запрос GET Header: { "token": "eyJhbGciOiJSUzI1NiJ9..." }	Тест пройден. Получен полный плейлист. Рисунок Е.11.
playlist/deactivate/{trackId}	HTTP запрос GET Header: { "token": "eyJhbGciOiJSUzI1NiJ9..." } URL: playlist/deactivate/5d7f40c9-ed63-4398-9165-ad1b21d97d68	Тест пройден. Песня исключена из плейлиста.
control/playpause	HTTP запрос GET Header: { "token": "eyJhbGciOiJSUzI1NiJ9..." }	Тест пройден. Изменено состояние плеера.
control/volume	HTTP запрос GET Header: { "token": "eyJhbGciOiJSUzI1NiJ9..." }	Тест пройден. Получено текущее значение громкости воспроизведения плеера.
control/soundtrackPosition/{value}	HTTP запрос PUT Header: { "token": "eyJhbGciOiJSUzI1NiJ9..." } URL: control/soundtrackPosition/104780	Тест пройден. Установлена новая позиция воспроизведения песни. Рисунок Е.12.

Результаты тестирования веб сервисов представлены в приложении Е. Тестирование проводилось с помощью сервисного приложения Advanced Rest Client.

Тестирование веб-сервисов показало стабильную работу серверного приложения на заданном наборе тестов.

3.4.3.1 Модульное тестирование

Модульное тестирование позволяет проверить корректность работы отдельных модулей программы. При помощи модульного тестирования можно в автоматическом режиме проверить корректно ли работают все функции программы, а также убедиться в том, что изменение кода не привело к ошибкам.

В качестве библиотеки для модульного тестирования разработанного программного сервиса была выбрана JUnit. В сравнении с другими библиотеками модульного тестирования Junit имеет следующие преимущества:

- легкая интеграция со средой разработки;

- в качестве языка для написания тестов используется Java;
- наличие подробной документации.

Модульные тесты в JUnit Модульные тесты в JUnit представлены в виде обычного Java класса, методы которого помечены аннотацией `@Test`. Каждый метод отвечает за конкретный тестовый случай (кейс).

В таблице 14 представлено описание разработанных модульных тестов.

Таблица 14 – Описание модульных тестов

Тестовый класс	Тестовый метод	Описание теста
Remote API	determineHWNDFromWindowClasses	Найти в системе окно-обработчик плеера AIMP (AIMP2_RemoteInfo) и вернуть его дескриптор.
Remote API	init	Инициализирует список воспроизведения и опции плейлиста. Ожидаемый результат: список воспроизведения и его опции проинициализированы.
Remote API	ExecuteCommand	Выполнить команду с ID 13. Ожидаемый результат: остановится воспроизведение музыки.
Remote API	GetProperty	Получить значение свойства version, ID 16. Ожидаемый результат: будет получена версия плеера (4.70.220).
Remote API	SetProperty	Установить значение свойства плеера 17 (version) в значение 4.70.220_test. Ожидаемый результат: версии плеера будет присвоено новое значение.
Remote API	GetTrackInfo	Получить информацию о воспроизводимом треке. Ожидаемая информация: получена информация о текущем треке.

Продолжение таблицы 14

Тестовый класс	Тестовый метод	Описание теста
Remote API	getPlayList	Получить список воспроизведения. Ожидаемый результат: получен список воспроизведения.
Remote API	getPlayListProp	Получить опции списка воспроизведения. Ожидаемый результат: получены опции списка воспроизведения.
Remote API	modifyPlayList	Переместить трек с позиции 2 на позицию 15. Ожидаемый результат: позиция трека будет изменена в соответствии с параметрами.
Remote API	createWindow	Создать окно-обработчик с базовым классом AIMP_Helper. Ожидаемый результат: будет создано окно-обработчик.

Всего было составлено 10 модульных тестов. Все тесты были пройдены успешно (рисунок Е.14). Таким образом, модульное тестирование разработанного программного сервиса показало корректность его работы на заданных тестовых случаях.

3.4.4 Тестирование клиентской части приложения

Для проверки корректности работы разработанного программного обеспечения сервиса удаленного управления медиасистемой будет использовано функциональное тестирование в соответствии с выделенными функциональными требованиями (бизнес-процессами). Результаты тестирования представлены в таблице 18.

Алгоритм авторизации в программном сервисе удаленного управления медиасистемой будет проведен с помощью функциональное тестирование методом функциональных диаграмм. Результаты тестирования представлены в таблице 15.

Таблица 15 – Функциональное тестирование бизнес-процесса «Авторизироваться»

№	Описание теста	Предполагаемый результат	Полученный результат	Результат тестирования
1	Логин: aVoid Пароль: correctPassword	Успешная авторизация в системе. Отрисовка пользовательского интерфейса соответствующего роли пользователя в системе.	Успешная авторизация в системе. Отрисовка пользовательского интерфейса соответствующего роли пользователя в системе.	Тест пройден. Рисунок Е. 15.
2	Логин: aVoid Пароль:	Ошибка. Сообщение об ошибке с текстом: «Введите пароль»	Ошибка. Сообщение об ошибке с текстом: «Введите пароль»	Тест пройден
3	Логин: Пароль: test	Ошибка. Сообщение об ошибке с текстом: «Введите логин»	Ошибка. Сообщение об ошибке с текстом: «Введите логин»	Тест пройден
4	Логин: Пароль:	Ошибка. Сообщения об ошибках с текстом: «Введите логин» и «Введите пароль»	Ошибка. Сообщения об ошибках с текстом: «Введите логин» и «Введите пароль»	Тест пройден
5	Логин: aVoid Пароль: incorrect	Ошибка. Сообщение с текстом: «Неверный логин или пароль»	Ошибка. Сообщение с текстом: «Неверный логин или пароль»	Тест пройден
6	Логин: incorrect Пароль: incorrect	Ошибка. Сообщение с текстом: «Неверный логин или пароль»	Ошибка. Сообщение с текстом: «Неверный логин или пароль»	Тест пройден

Для тестирования алгоритма авторизации методом функциональных диаграмм определим причины и следствия.

Причины:

- 1) введен логин пользователя, зарегистрированного в системе;
- 2) введен несуществующий логин пользователя;
- 3) логин пользователя не введен;
- 4) введен верный пароль пользователя;
- 5) введен неверный пароль пользователя;

б) пароль пользователя не введен.

Следствия:

- а) успешная авторизация в системе и отрисовка пользовательского интерфейса соответствующего роли пользователя в системе;
- б) сообщение об ошибке с текстом: «Введите логин»;
- с) сообщение об ошибке с текстом: «Введите пароль»;
- д) сообщение с текстом: «Неверный логин или пароль».

На основе выявленных причин и следствий построим функциональную диаграмму. Диаграмма представлена на рисунке 29.

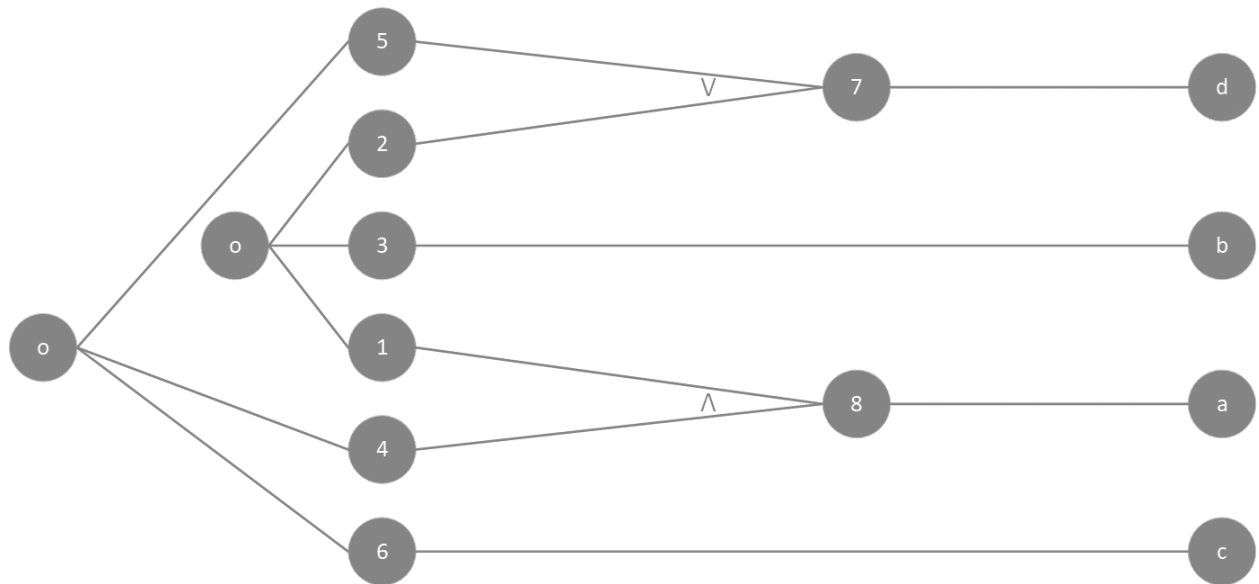


Рисунок 29 – Функциональная диаграмма

При построении графа для наглядности были использованы промежуточные вершины 7 и 8.

Таблица 16 – Таблица решений по графу

Причина	1	1	0	-	0
	2	0	0	-	1
	3	0	1	-	0
	4	1	-	0	0
	5	0	-	0	1
	6	0	-	1	0
Следствие	а	1	-	-	-

Продолжение таблицы 16

Следствие	b	-	1	-	-
	c	-	-	1	-
	d	-	-	-	1

Таким образом, на основании данных, полученных в ходе функционального тестирования, можно сделать выводы о том, что алгоритм авторизации в системе является надежным, так как он устойчив к различным тестам.

Таблица 17 – Результаты функционального тестирования приложения

Бизнес-процесс (Функц. треб.)	Тест и параметры	Результат
Зарегистрироваться	Ввести валидные данные регистрации. Email: test1@testfn.ru Логин: test1 Пароль: test1	Тест пройден. Пользователь зарегистрирован в системе. Рисунок Е.13.
	Ввести данные регистрации с отличающимися паролями. Пароль: test1 Повтор пароля: test2	Тест пройден. Выведено уведомление о несовпадении паролей.
Зарегистрироваться	Ввести данные с email пользователя, существующего в системе. Email: avoidpnz@gmail.com	Тест пройден. Выведено уведомление об использовании данного email адреса.
	Ввести не валидный email адрес. Email: test1	Тест пройден. Выведено уведомление о не валидности email адреса
	Ввести не валидные данные. Email: test1@test.test Пароль: test1234	Тест пройден. Выведено уведомление.

Продолжение таблицы 17

Бизнес-процесс (Функц. треб.)	Тест и параметры	Результат
Формирование списка воспроизведения	Загрузить страницу с плейлистом.	Тест пройден. Плейлист отображен. Рисунок Е.16.
	Попытка загрузить страницу плейлиста без доступа к интернету.	Тест пройден. Показано оповещение об ошибке.
Просмотр подробной информации о песне	Просматривать подробную информацию о песне.	Тест пройден. Загружена подробная информация о песне. Рисунок Е.17.
Оставить оценку с отзывом о песне	Оставить отзыв и оценку песне	Тест пройден. Отзыв добавлен на страницу песни.
	Оставить только оценку песне без текстового отзыва.	Тест пройден. Выведено сообщение об ошибке.
Формирование списка пользователей	Загрузить страницу списка пользователей.	Тест пройден. Список пользователей загружен. Рисунок Е.18.
Управлять списком воспроизведения	Исключить песню из плейлиста. Исключаема песня: Reminds Me Of You	Тест пройден. Песня исключена из плейлиста. Рисунок Е.19.
	Изменить позицию песни в плейлисту. Изменяемая песня: Reminds Me Of You Конечная позиция: 1	Тест пройден. Песня успешно перемещена на конечную позицию.
Управлять плеером	Изменить громкость воспроизведения плеера. Текущее значение громкости: 45% Новое значение громкости: 100%	Тест пройден. Значение громкости воспроизведения успешно изменено. Рисунок Е.20.
	Изменить позицию воспроизведения песни.	Тест пройден. Позиция воспроизведения песни успешно изменена.

Окончание таблицы 17

Бизнес-процесс (Функц. треб.)	Тест и параметры	Результат
Управлять плеером	Изменить состояние плеера. Текущее состояние: выполняется воспроизведение	Тест пройден. Состояние плеера успешно изменено. Воспроизведение приостановлено.
	Переключить текущую воспроизводимую песню к следующей.	Тест пройден. Успешно установлена к воспроизведению следующая песня.

Скриншоты результатов тестирования представлены на рисунках в приложении Е.

Результаты функционального тестирования показывают стабильную работу приложения на данном наборе тестов. Приложение выполняет все функциональные требования.

3.5 Анализ качества кода

Для анализа качества кода был вычислен комплексный показатель качества кода. Этот показатель разработан специалистами из Carnegie Mellon Software Engineering Institute. Комплексный показатель качества кода может принимать значения от 0 до 100 и показывает относительную сложность поддержки кода. Чем больше значение этой метрики, тем легче поддерживать код.

Для вычисления комплексного показателя качества кода необходимо подсчитать количество операторов и операндов.

Вычисление показателя качества для всего проекта нецелесообразно из-за сложности вычислений. Также нет необходимости вычислять данный показатель для модулей, обновление которых планируется крайне редко. Поэтому показатель качества кода стоит вычислять для критически важных в проекте модулей или же методов.

В таблице 18 представлены список операторов и операндов для функции изменения списка воспроизведения (modifyPlaylist).

Таблица 18 – Данные для расчета комплексного показателя качества кода

Операторы		Операнды	
Оператор	Количество	Операнд	Количество
!=	1	playlist	6
try	2	fostream	3
;	18	PLAYLIST_FILE	1
+	4	bw	2
for	1	s	2
:	1	tmpList	3
new	5	jsMap	4
<>	5	socket	3
{}	6	gsonObject	3
=	12	track	3
catch	1	playListProp	1
\r	1	size	2
\n	1	gsonMapBuilder	2

Для вычисления комплексного показателя качества кода необходимо подсчитать следующие переменные:

- η_1 – число уникальных операторов;
- η_2 – число уникальных операндов;
- N_1 – общее число операторов;
- N_2 – общее число операндов;
- CC – структурная сложность программы;
- LoC – количество строк кода в программе.

Основываясь на таблице 17, были подсчитаны необходимые параметры:

- $\eta_1 = 13$;

- $\eta_2 = 13$;
- $N_1 = 58$;
- $N_2 = 35$.

Для вычисления словаря программы необходимо использовать следующую формулу:

$$\eta = \eta_1 + \eta_2.$$

Длина программы вычисляется при помощи данной формулы:

$$N = N_1 + N_2.$$

Таким образом, получаем что словарь программы $\eta = 26$, а длина программы $N = 96$. Используя данные значения, можно подсчитать вычислительную сложность программы при помощи следующей формулы:

$$V = N \cdot \log_2 \eta.$$

После подстановки словаря и длины программы получим, что вычислительная сложность программы $V = 451,2$.

Основываясь на листинге 1, были вычислены показатели:

- структурной сложности программы $CC = 5$;
- количества строк кода $LoC = 25$.

Для вычисления комплексного показателя сложности кода необходимо использовать следующую формулу [N]:

$$MI = \max(0, (171 - 5,2 \cdot \ln(V) - 0,23 * CC - 16,2 \cdot \ln(LoC)) \cdot 100/171).$$

При помощи данной формулы был вычислен комплексный показатель сложности кода $MI = 50,3$. Показатель является достаточно высоким, что говорит о хорошем качестве кода. Понимание и поддержка разработанного алгоритма не должны вызывать проблем.

3.6 Планирование разработки и оценка бюджета

Планирование разработки и оценка бюджета была выполнена при помощи программы для управления проектами Microsoft Project. Проект был начат 19 апреля 2021 года. При планировании разработки было выделено 25 задач. Список задач и их описание представлено на рисунке 30.

	i	Название задачи	Длительность	Начало	Окончание	Пре	Названия ресурсов
1		Начало проекта	0 дней	Пн 19.04.21	Пн 19.04.21		
2		«Анализ требований и постановка задачи	4 дней	Пн 19.04.21	Чт 22.04.21	1	
3		Анализ предметной области	1 день	Пн 19.04.21	Пн 19.04.21		Аналитик[30%];Программист
4		Сравнение программ-аналогов	2 дней	Пн 19.04.21	Вт 20.04.21		Аналитик[70%]
5		Постановка задачи	1 день	Ср 21.04.21	Ср 21.04.21	3;4	Аналитик
6		Анализ функциональных требований	1 день	Чт 22.04.21	Чт 22.04.21	5	Аналитик;Программист
7		Анализ требования и постановка задачи завершена	0 дней	Чт 22.04.21	Чт 22.04.21	6	
8		«Проектирование программных средств	8 дней	Пт 23.04.21	Вт 04.05.21	2	
9		Проектирование структур данных сервиса	4 дней	Пт 23.04.21	Ср 28.04.21		Аналитик
10		Проектирование REST API приложения	4 дней	Пт 23.04.21	Ср 28.04.21		Программист
11		Проектирование интерфейса	2 дней	Чт 29.04.21	Пт 30.04.21	9;10	Аналитик
12		Проектирование клиентского приложения	2 дней	Пн 03.05.21	Вт 04.05.21	11	Аналитик
13		Проектирование программных средств завершено	0 дней	Вт 04.05.21	Вт 04.05.21	11;12	
14		«Реализация	12 дней	Ср 05.05.21	Чт 20.05.21	8	
15		Реализация структур данных сервиса	5 дней	Ср 05.05.21	Вт 11.05.21		Программист
16		Реализация REST API приложения	5 дней	Ср 12.05.21	Вт 18.05.21	15	Программист[80%]
17		Реализация интерфейса	2 дней	Ср 12.05.21	Чт 13.05.21	15	Программист[20%]
18		Реализация клиентского приложения	2 дней	Ср 19.05.21	Чт 20.05.21	16;17	Программист
19		Реализация завершена	0 дней	Чт 20.05.21	Чт 20.05.21	18	
20		«Тестирование	4 дней	Пт 21.05.21	Ср 26.05.21	14	
21		Функциональное тестирование	2 дней	Пт 21.05.21	Пн 24.05.21		Тестировщик
22		Модульное тестирование	2 дней	Вт 25.05.21	Ср 26.05.21	21	Тестировщик
23		Тестирование завершено	0 дней	Ср 26.05.21	Ср 26.05.21	22	
24		Составление и оформление документации	3 дней	Чт 27.05.21	Пн 31.05.21	20	Аналитик;Программист
25		Конец проекта	0 дней	Пн 31.05.21	Пн 31.05.21	24	

Рисунок 30 – Список задач проекта

Для реализации проекта было выделено три трудовых ресурса: аналитик, программист и тестировщик. Описание ресурсов представлено на рисунке 31.

	i	Название ресурса	Тип	Единицы измерения	Краткое название	Группа	Макс. единиц	Стандартная ставка	Ставка сверхурочных	Затраты на исполн.	Начисление	Базовый календарь
1		Аналитик	Трудовой		А		100%	170,00 Р/ч	340,00 Р/ч	0,00 Р	Пропорциональное	Стандартный
2		Программист	Трудовой		П		100%	160,00 Р/ч	320,00 Р/ч	0,00 Р	Пропорциональное	Стандартный
3		Тестировщик	Трудовой		Т		100%	140,00 Р/ч	280,00 Р/ч	0,00 Р	Пропорциональное	Стандартный

Рисунок 31 – Список ресурсов проекта

В результате выделения задач и необходимых ресурсов проекта была построена диаграмма Ганта (рисунок 32). На диаграмме Ганта представлены задачи продолжительностью и взаимосвязи с между задачами в проекте. Также на диаграмме отображены даты начала и конца каждой задачи и назначенные ресурсы для выполнения задачи.

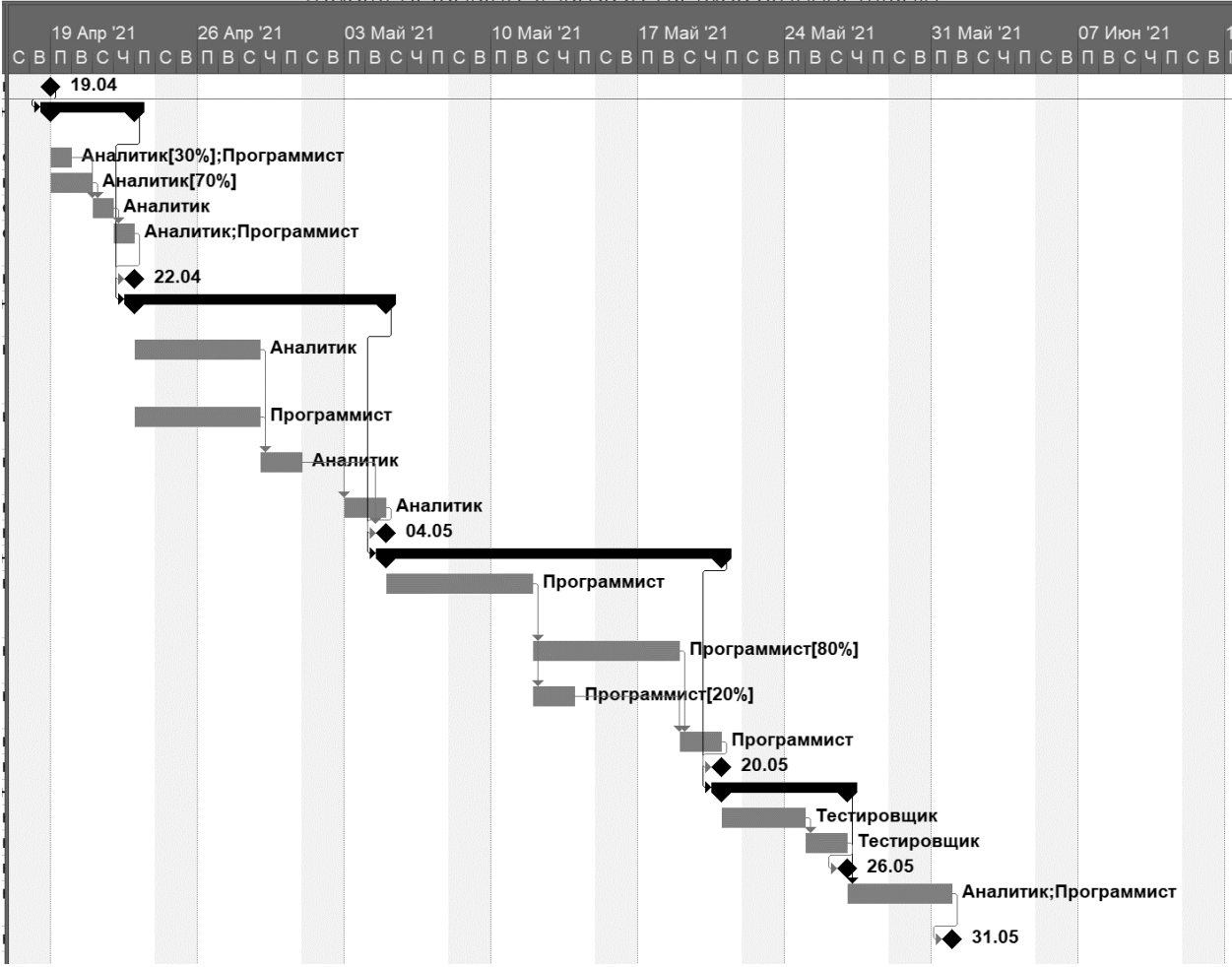


Рисунок 32 – Диаграмма Ганта

На рисунке 33 и 34 представлены гистограмма загрузки ресурсов и гистограмма затрат на задачи соответственно.

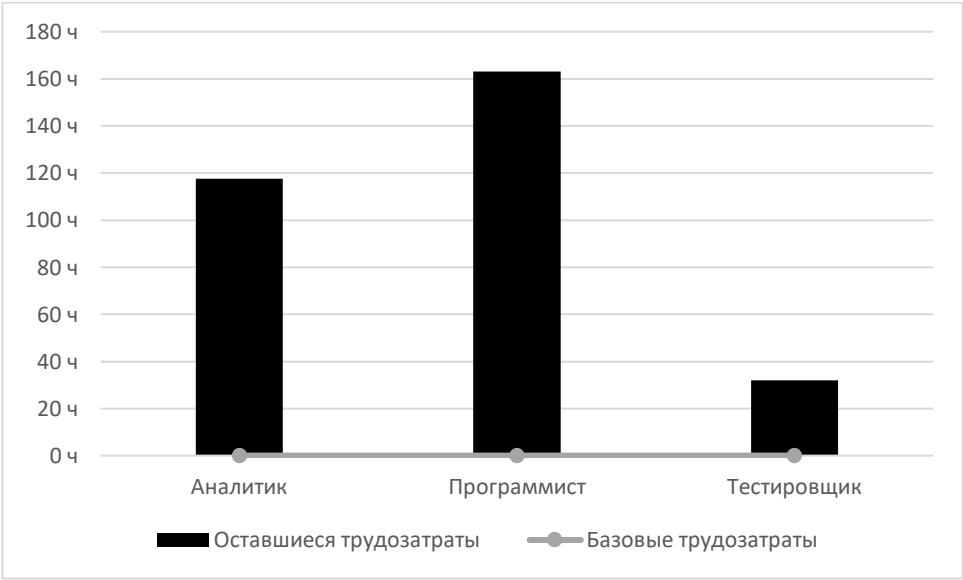


Рисунок 33 – Гистограмма загрузки ресурсов

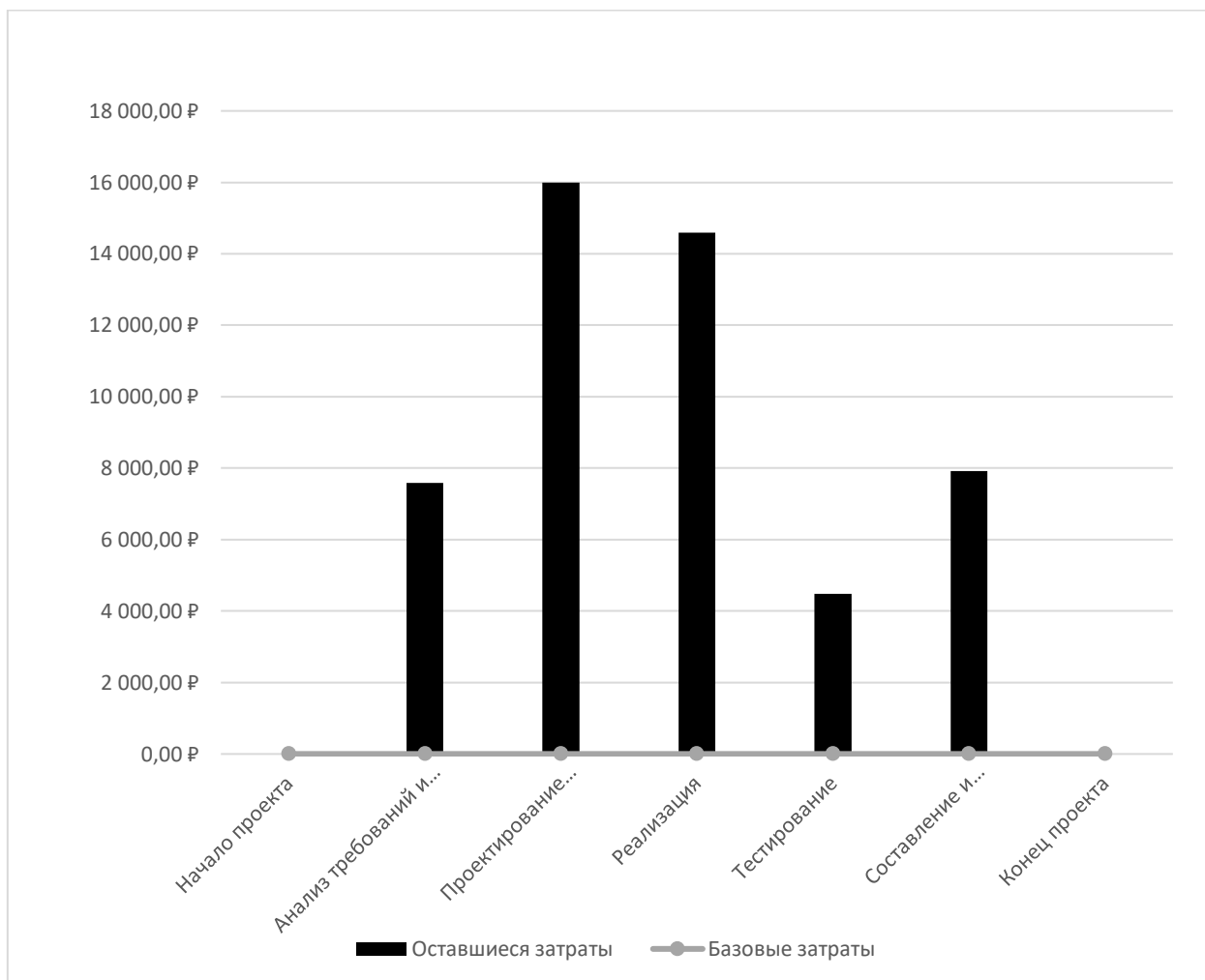


Рисунок 34 – Гистограмма затрат на задачи

Таким образом разработка проекта составит 29 полных рабочих дней с 19 апреля 2021 года по 31 мая 2021 года. Трудозатраты составят 312 человеко-часов. Оценка бюджета проекта составляет 50 584 рубля.

Заключение

В ходе выполнения выпускной квалификационной работы разработан программный сервис удаленного управления медиасистемой.

В результате научной исследовательской работы был проведен анализ сервисов удаленного управления компьютерами. Были рассмотрены и охарактеризованы технологии удаленного управления компьютерами.

Протоколы и технологии удаленного управления компьютерными системами были классифицированы. Для каждой категории классификации были приведены описание и примеры.

На этапе анализа предметной области и постановки требований были выделены бизнес-процессы, для решения которых разрабатывалось приложение. Была создана модель данных предметной области. На основе модели данных было проведено логическое и физическое моделирование базы данных. Для базы данных были разработаны запросы, процедуры и триггеры.

Клиентское приложение было разработано с использованием языка программирования JavaScript React. Разработка проводилась в среде программирования WebStorm 2020.01.

Серверная часть представлена СУБД PostgreSQL и серверным приложением, разработанным на языке программирования Java. СУБД содержит в себе сущности, их атрибуты и связи, задействованные в предметной области и основную бизнес-логику приложения, в виде представлений, хранимых процедур и триггеров.

По всем компонентам полученного приложения было проведено функциональное тестирование. Тестирование показало стабильность работы системы на использованном наборе тестов.

В ходе выполнения выпускной квалификационной работы проведено планирование и оценка бюджета разработки при помощи программы для управления проектами Microsoft Project. Затраты на проект составили 50 584 рубля. Время разработки составило 29 рабочих дней.

Таким образом в результате выполнения выпускной квалификационной работы разработан программный сервис удаленного управления

медиасистемой, который удовлетворяет всем требованиям технического задания.

Modeling Suite / Т.Е. Точилкина. – Академия бюджета и казначейства, 2009. – 169 с.

10. Джим Арлоу, Айла Нейштадт / UML2 и Унифицированный процесс. Практический объектно-ориентированный анализ и проектирование – Санкт-Петербург, Изд-во Символ-Плюс, 2007.

11. Вирт Н. Алгоритмы и структуры данных. Новая версия для Оберона + CD / Никлаус Вирт – Москва: ДМК Пресс, 2010 – 272 с

12. PostgreSQL // Официальная документация PostgreSQL [Электронный ресурс]. URL: <https://www.postgresql.org/docs/> (дата обращения: 28.05.2021).

13. GitHub // Серверное приложение программного сервиса удаленного управления медиасистемой [Электронный ресурс]. URL: <https://github.com/aVoidPNZ/aimp-control-server> (дата обращения: 28.05.2021).

14. GitHub // Клиентское приложение программного сервиса удаленного управления медиасистемой [Электронный ресурс]. URL: <https://github.com/aVoidPNZ/aimp-control-client> (дата обращения: 28.05.2021).

15. Гамма Э. Приемы объектно-ориентированного программирования. Паттерны программирования. / Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. – Санкт Петербург: Питер, 2001. – 368 с

Приложение А
ГЛОССАРИЙ
(справочное)

БД – база данных – совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняют в соответствии с правилами средств моделирования данных.

Веб-интерфейс – совокупность средств, при помощи которых пользователь взаимодействует с веб-сайтом или веб-приложением через браузер.

Веб-сервер – сервер, принимающий HTTP-запросы от клиентов, обычно веб-браузеров, и выдающий им HTTP-ответы, как правило, вместе с HTML-страницей, изображением, файлом, медиа-поток или другими данными.

Хостинг (англ. hosting) – услуга по предоставлению ресурсов для размещения информации на сервере, постоянно находящемся в сети (обычно Интернет).

СУБД – система управления базами данных – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

API – программный интерфейс приложения - набор готовых классов, процедур, функций, структур и констант, предоставляемых для использования во внешних программных продуктах.

Алгоритм – конечная совокупность точно заданных правил решения произвольного класса задач или набор инструкций, описывающих порядок действий исполнителя для решения некоторой задачи.

Приложение Б
КОНЦЕПТУАЛЬНАЯ МОДЕЛЬ БАЗЫ ДАННЫХ
(рекомендуемое)

Таблица Б.1 – Описание сущностей предметной области

№	Наименование	Описание	Категория	Количество экземпляров	Режим обновления
1	Пользователь (пользователь, администратор)	Записи пользователей сервиса	Стержневая	15000	Каждый раз, когда регистрируется новый пользователь, а также при выходе пользователя из сети. При обновлении профиля пользователя.
2	Роль пользователя	Записи ролей пользователя в системе. Необходимы для разграничения функционала и задач системы	Стержневая	20000	Каждый раз, когда регистрируется или создается новый пользователь. При изменении ролей пользователя
3	Песня	Основной объект бизнес-процессов. Содержит записи загруженных в систему песен	Стержневая	150000	Каждый раз, когда загружается новая песня в систему. А также при обновлении списка воспроизведения
4	Оценка	Оценка и отзыв пользователя о песне.	Стержневая	200000	Если пользователь захочет оценить песню, либо изменить уже оставленный отзыв

Таблица Б.2 – Описание связей сущностей предметной области

№	Тип сущности 1	Тип сущности 2	Описание	Мощность	Обязательность	Ограничения
1	Пользователь (пользователь, администратор)	Роль пользователя	Пользователь может обладать множеством ролей в системе	1:M	Обязательно	Обязательно наличие хотя бы одной основной роли. Также необходимо избегать дублирование одинаковых ролей для пользователя
2	Песня	Оценка	Одна песня может иметь множество оценок от разных пользователей сервиса	1:M	Необязательно	

Таблица Б.3 – Атрибуты сущности «Пользователь»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения null	Роль	Пример	Примечание
Email	email пользователя	Символьный	20	A-z, 0-9, . , @	Нет		avoidpnz@gmail.com	Должен удовлетворять паттерну email адреса
Логин	Логин пользователя	Символьный	20	A-z, 0-9	Нет		Avoid	
Пароль	Пароль пользователя	Символьный	20	A-z, 0-9	Нет		Password	Не менее 8 символов
Имя	Имя пользователя	Символьный	20	A-z, A-я	Да		Павел	
Фамилия	Фамилия пользователя	Символьный	25	A-z, 0-9	Да		Бесчетвертев	
Отчество	Отчество пользователя	Символьный	20	A-z, 0-9	Да		Александрович	
Дата рождения	Дата рождения пользователя	Символьный	10	##.##.#### где # - это 0-9	Да		18.11.1999	Соответствие шаблону
Номер телефона	Номер телефона пользователя	Числовой	11	0-9	Да		89004002233	

Таблица Б.4 – Атрибуты сущности «Роль пользователя»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения null	Роль	Пример	Примечание
Название	Название роли	Символьный	20	символы UTF-8	Да		Тестовая роль для пользователя	
Роль пользователя	Тип роли пользователя	Символьный	20	A-z	Нет		Registered	Значение должно удовлетворять заданным в системе ролям
Приоритет роли	Значение приоритета для данной роли	Числовой	4	0-9	Нет		0	Роль с меньшим приоритетом является основной

Таблица Б.5 – Атрибуты сущности «Песня»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения null	Роль	Пример	Примечание
Название	Название каталога	Символьный	20	символы UTF-8	Нет		Dreams	
Исполнитель	Исполнитель (Исполнители) песни	Символьный	30	символы UTF-8	Да		David Guetta, MORTEN, Lanie Gardner	

Продолжение таблицы Б.5

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения null	Роль	Пример	Примечание
Альбом	Альбом песни	Символьный	30	символы UTF-8	Да		Громкие новинки Декабря 2	
Год выпуска	Год выпуска песни	Символьный	4	0-9	Да		2020	
Жанр	Жанр песни	Символьный	20	A-z	Да		Pop Dance Other	
Продолжительность	Продолжительность песни	Числовой	10	0-9	Нет		204	
Дата и время	Метка времени, когда была загружена песня	Дата и время (метка времени)	8 байт		Нет		13.05.21 14:00	
Комментарий	Комментарий к загружаемой песне	Символьный	255	символы UTF-8	Да		Крутая песня	

Таблица Б.6 – Атрибуты сущности «Оценка»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения null	Роль	Пример	Примечание
Оценка	Оценка песни	Числовой	1 байт	{0,1,2,3,4,5}	Нет		5	
Отзыв	Отзыв пользователя о песне	Символьный	255	Любые символы принятой кодировки	Да		Очень понравилась эта песня!	
Дата и время	Метка времени, когда была поставлена оценка	Дата и время (метка времени)	8 байт		Нет		13.05.21 14:00	

Приложение В
СРИПТЫ ДЛЯ СОЗДАНИЯ ОБЪЕКТОВ БАЗЫ ДАННЫХ
НА ЯЗЫКЕ PPOSTGRESQL
(рекомендуемое)

```
--
-- Name: feedback; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.feedback (
    rating integer,
    comment character varying(255),
    updt timestamp without time zone DEFAULT CURRENT_TIMESTAMP,
    ct timestamp without time zone DEFAULT CURRENT_TIMESTAMP,
    ver integer DEFAULT 0 NOT NULL,
    uuid uuid DEFAULT public.uuid_generate_v1mc() NOT NULL,
    updu uuid,
    cu uuid,
    soundtrack uuid NOT NULL
);

--
-- Name: soundtrack; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.soundtrack (
    file_size integer,
    duration integer,
    "order" integer NOT NULL,
    sample_rate integer,
    bit_rate integer,
    file_format character varying(32) NOT NULL,
    comment character varying(255),
    updt timestamp without time zone DEFAULT CURRENT_TIMESTAMP,
    ct timestamp without time zone DEFAULT CURRENT_TIMESTAMP,
    ver integer DEFAULT 0 NOT NULL,
    is_active boolean DEFAULT true NOT NULL,
    uuid uuid DEFAULT public.uuid_generate_v1mc() NOT NULL,
    updu uuid,
    cu uuid,
    file_path text DEFAULT ''::text NOT NULL,
    genre character varying[],
    artist character varying,
    title character varying,
    album character varying,
    year character varying(4)
);

--
-- Name: user; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public."user" (
    first_name character varying(128),
```

```

        middle_name character varying(128),
        last_name character varying(128),
        login character varying(128) NOT NULL,
        shapass character varying(128) NOT NULL,
        birthday timestamp without time zone,
        email character varying(127) NOT NULL,
        phone character varying(45),
        site character varying(255),
        last_offline timestamp without time zone DEFAULT
CURRENT_TIMESTAMP,
        updt timestamp without time zone DEFAULT CURRENT_TIMESTAMP,
        ct timestamp without time zone DEFAULT CURRENT_TIMESTAMP,
        ver integer DEFAULT 0 NOT NULL,
        is_first_login boolean DEFAULT true,
        uuid uuid DEFAULT public.uuid_generate_v1mc() NOT NULL,
        updu uuid,
        cu uuid,
        avatar_path text DEFAULT '':text,
        gender character varying,
        title character varying
    );
--
-- Name: user_role; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.user_role (
    name character varying(1024),
    priority integer DEFAULT 2 NOT NULL,
    updt timestamp without time zone DEFAULT CURRENT_TIMESTAMP,
    ct timestamp without time zone DEFAULT CURRENT_TIMESTAMP,
    ver integer DEFAULT 0 NOT NULL,
    is_first_login boolean DEFAULT true,
    uuid uuid DEFAULT public.uuid_generate_v1mc() NOT NULL,
    updu uuid,
    cu uuid,
    "user" uuid NOT NULL,
    role character varying NOT NULL
);
--
-- Name: feedback_feedback_pkey; Type: CONSTRAINT; Schema: public;
Owner: postgres
--

ALTER TABLE ONLY public.feedback
    ADD CONSTRAINT feedback_pkey PRIMARY KEY (uuid);
--

```

```

-- Name: soundtrack soundtrack_pkey; Type: CONSTRAINT; Schema:
public; Owner: postgres
--

ALTER TABLE ONLY public.soundtrack
    ADD CONSTRAINT soundtrack_pkey PRIMARY KEY (uuid);

--

-- Name: user user_pkey; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public."user"
    ADD CONSTRAINT user_pkey PRIMARY KEY (uuid);

--

-- Name: user_role user_role_pkey; Type: CONSTRAINT; Schema: public;
Owner: postgres
--

ALTER TABLE ONLY public.user_role
    ADD CONSTRAINT user_role_pkey PRIMARY KEY (uuid);

--

-- Name: user user_un; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public."user"
    ADD CONSTRAINT user_un UNIQUE (email);

--

-- Name: user_birthday_idx; Type: INDEX; Schema: public; Owner:
postgres
--

CREATE INDEX user_birthday_idx ON public."user" USING btree
(birthday);

--

-- Name: user_email_idx; Type: INDEX; Schema: public; Owner:
postgres
--

```

```
CREATE UNIQUE INDEX user_email_idx ON public."user" USING btree
(email);
```

```
--
-- Name: user_first_name_idx; Type: INDEX; Schema: public; Owner:
postgres
```

```
--
CREATE INDEX user_first_name_idx ON public."user" USING btree
(first_name);
```

```
--
-- Name: user_last_name_idx; Type: INDEX; Schema: public; Owner:
postgres
```

```
--
CREATE INDEX user_last_name_idx ON public."user" USING btree
(last_name);
```

```
--
-- Name: user_role_role_idx; Type: INDEX; Schema: public; Owner:
postgres
```

```
--
CREATE INDEX user_role_role_idx ON public.user_role USING btree
(role);
```

```
--
-- Name: user_role_user_idx; Type: INDEX; Schema: public; Owner:
postgres
```

```
--
CREATE INDEX user_role_user_idx ON public.user_role USING btree
("user");
```

```
--
-- Name: playlist _RETURN; Type: RULE; Schema: public; Owner:
postgres
```

```
--
CREATE OR REPLACE VIEW public.playlist AS
SELECT soundtrack.uuid,
```

```

    soundtrack."order",
    soundtrack.title,
    soundtrack.artist,
    soundtrack.album,
    soundtrack.genre,
    soundtrack.is_active,
    soundtrack.cu,
    soundtrack.ct,
    soundtrack.year,
    count(feedback.soundtrack) AS feedbacks,
    avg(feedback.rating) AS avg_rating,
    soundtrack.updu,
    soundtrack.updt,
    soundtrack.ver
FROM (public.soundtrack
      LEFT JOIN public.feedback ON ((soundtrack.uuid =
feedback.soundtrack)))
GROUP BY soundtrack.uuid;

--
-- Name: user_public _RETURN; Type: RULE; Schema: public; Owner:
postgres
--

CREATE OR REPLACE VIEW public.user_public AS
SELECT "user".uuid,
       "user".login,
       "user".last_offline,
       "user".avatar_path,
       "user".birthday,
       "user".first_name,
       "user".middle_name,
       "user".last_name,
       "user".gender,
       "user".phone,
       count(soundtrack.*) AS soundtracks
FROM (public."user"
      LEFT JOIN public.soundtrack ON ((soundtrack.cu = "user".uuid)))
GROUP BY "user".uuid;

--
-- Data for Name: feedback; Type: TABLE DATA; Schema: public; Owner:
postgres
--

COPY public.feedback (rating, comment, updt, ct, ver, uuid, updu,
cu, soundtrack) FROM stdin;

```

```

2  23456 2021-05-27 21:24:22.011    2021-05-27 21:24:22.011    0
    a465eac9-109e-4395-b041-a5a007dce13c 9a8db856-16b3-469e-85d0-
8f408b2441b7    9a8db856-16b3-469e-85d0-8f408b2441b7 2c1f097a-9b0f-
4a49-bbb3-11efa28c4dc0
1  ghj  2021-05-27 21:35:49.015    2021-05-27 21:35:49.015    0
    11e5e8d7-5ab1-4bc0-89f6-5f70a20ca218 9a8db856-16b3-469e-85d0-
8f408b2441b7    9a8db856-16b3-469e-85d0-8f408b2441b7 f83e4381-1617-
4151-83bb-11abd44c8a23
\.
```

--

```
-- Data for Name: soundtrack; Type: TABLE DATA; Schema: public;
Owner: postgres
--
```

```

COPY public.soundtrack (file_size, duration, "order", sample_rate,
bit_rate, file_format, comment, updt, ct, ver, is_active, uuid, updu,
cu, file_path, genre, artist, title, album, year) FROM stdin;
9984515 248 5 44100 320000 mp3 \N 2021-05-25
21:01:28.027 2021-05-24 21:51:51.258 33 t df92a5a8-cec0-
4585-9010-7a9f7b33cb8e 9a8db856-16b3-469e-85d0-8f408b2441b7
9a8db856-16b3-469e-85d0-8f408b2441b7 /9a8db856-16b3-469e-85d0-
8f408b2441b7/14.-dj-antonio-bertie-scott---losing-my-religion.mp3
{Pop,Dance,Other} Dj Antonio, Bertie Scott Losing My
Religion Громкие новинки Декабря 2 2020
6657530 165 2 44100 320000 mp3 \N 2021-05-31
10:56:44.874 2021-05-24 21:50:33.615 45 t bb3c1176-01bd-
4327-9253-add309364574 9a8db856-16b3-469e-85d0-8f408b2441b7
9a8db856-16b3-469e-85d0-8f408b2441b7 /9a8db856-16b3-469e-85d0-
8f408b2441b7/42.-slame-hensy---drug-pomogi-mne.mp3 {Pop,Dance,Other}
HENSY, Slame Друг помощи мне Громкие новинки Декабря 2 2020
6362899 158 1 44100 320000 mp3 \N 2021-05-31
15:51:13.575 2021-05-24 00:22:57.596 67 t 2c1f097a-9b0f-
4a49-bbb3-11efa28c4dc0 9a8db856-16b3-469e-85d0-8f408b2441b7
9a8db856-16b3-469e-85d0-8f408b2441b7 /9a8db856-16b3-469e-85d0-
8f408b2441b7/22.-juice-wrld-the-kid-laroi.---reminds-me-of-you.mp3
{Pop,Dance,Other} Juice WRLD, The Kid LAROI Reminds Me Of You
Громкие новинки Декабря 2 2020
7801700 194 8 44100 320000 mp3 ppp 2021-05-24 21:35:29.81
2021-05-24 17:36:13.232 26 t cd5616a8-02cb-4517-81f6-
6ce3b838829e 9a8db856-16b3-469e-85d0-8f408b2441b7 9a8db856-16b3-
469e-85d0-8f408b2441b7 /9a8db856-16b3-469e-85d0-8f408b2441b7/33.-
markul-kuok---trenirovochnyy-den.mp3 {Pop,Dance,Other} КУОК,
Markul Тренировочный день Громкие новинки Декабря 2 2020
6685731 166 7 44100 320000 mp3 \N 2021-05-24
21:57:21.621 2021-05-24 20:03:06.072 39 t 8059141c-de34-
449f-a8ad-958c35cbf1e7 9a8db856-16b3-469e-85d0-8f408b2441b7

```

9a8db856-16b3-469e-85d0-8f408b2441b7 /9a8db856-16b3-469e-85d0-8f408b2441b7/78.-yarmak---chyo-tam-u-hohlov.mp3 {Pop,Dance,Other}
 YARMAK Чё там у хохлов Громкие новинки Декабря 2 2020
 5339890 132 4 44100 320000 mp3 \N 2021-05-31
 10:55:26.418 2021-05-24 19:54:07.711 67 t 61aea7e0-b94c-413f-8468-303b31493e49 9a8db856-16b3-469e-85d0-8f408b2441b7
 9a8db856-16b3-469e-85d0-8f408b2441b7 /9a8db856-16b3-469e-85d0-8f408b2441b7/38.-rakhim---big-money.mp3 {Pop,Dance,Other} Rakhim
 Big Money Громкие новинки Декабря 2 2020
 8206097 204 10 44100 320000 mp3 \N 2021-05-25
 14:35:52.395 2021-05-24 00:24:40.357 32 f f8ac10e9-3ee9-46c1-979a-6025ed8562bc 9a8db856-16b3-469e-85d0-8f408b2441b7
 9a8db856-16b3-469e-85d0-8f408b2441b7 /9a8db856-16b3-469e-85d0-8f408b2441b7/12.-david-guetta-morten-lanie-gardner---dreams.mp3
 {Pop,Dance,Other} David Guetta, MORTEN, Lanie Gardner Dreams
 Громкие новинки Декабря 2 2020
 6057729 150 9 44100 320000 mp3 \N 2021-05-27
 11:32:34.426 2021-05-24 22:14:48.292 14 f 5d7f40c9-ed63-4398-9165-ad1b21d97d68 9a8db856-16b3-469e-85d0-8f408b2441b7
 9a8db856-16b3-469e-85d0-8f408b2441b7 /9a8db856-16b3-469e-85d0-8f408b2441b7/39.-ramil---aromat.mp3 {Pop,Dance,Other} Ramil'
 Аромат Громкие новинки Декабря 2 2020
 3841500 95 12 44100 320000 mp3 1234567 2021-05-27
 14:06:59.002 2021-05-27 14:06:34.256 7 t f83e4381-1617-4151-83bb-11abd44c8a23 9a8db856-16b3-469e-85d0-8f408b2441b7
 9a8db856-16b3-469e-85d0-8f408b2441b7 /9a8db856-16b3-469e-85d0-8f408b2441b7/67.-mia-boyka---auf.mp3 {Pop,Dance,Other} Mia Boyka
 АУФ Громкие новинки Декабря 2 2020
 6314790 156 6 44100 320000 mp3 \N 2021-05-25
 14:35:58.754 2021-05-24 19:59:49.877 33 t cb361d80-c3ae-43c0-b8da-49550ec05e8e 9a8db856-16b3-469e-85d0-8f408b2441b7
 9a8db856-16b3-469e-85d0-8f408b2441b7 /9a8db856-16b3-469e-85d0-8f408b2441b7/77.-hanna---status-vlyublana.mp3 {Pop,Dance,Other}
 ХАННА СТАТУС ВЛЮБЛЕНА Громкие новинки Декабря 2 2020
 6805888 169 11 44100 320000 mp3 1234 2021-05-31
 18:54:16.747 2021-05-31 18:54:16.747 0 t 934a2f8e-955c-4797-a172-ef6bd68f84e0 9a8db856-16b3-469e-85d0-8f408b2441b7
 9a8db856-16b3-469e-85d0-8f408b2441b7 /9a8db856-16b3-469e-85d0-8f408b2441b7/15.-elvira-t---melanholiya.mp3 {Pop,Dance,Other}
 Elvira T Меланхолия Громкие новинки Декабря 2 2020
 9050317 225 3 44100 320000 mp3 \N 2021-05-31
 10:57:27.876 2021-05-24 00:21:15.959 28 t 87d5575d-3320-4542-b717-4f3973f6a6a6 9a8db856-16b3-469e-85d0-8f408b2441b7
 9a8db856-16b3-469e-85d0-8f408b2441b7 /9a8db856-16b3-469e-85d0-8f408b2441b7/03.-anivar---vysoko.mp3 {Pop,Dance,Other} ANIVAR
 Высоко Громкие новинки Декабря 2 2020
 \.

```
--
-- Data for Name: user; Type: TABLE DATA; Schema: public; Owner:
postgres
--
```

```
COPY public."user" (first_name, middle_name, last_name, login,
shapass, birthday, email, phone, site, last_offline, updt, ct, ver,
is_first_login, uuid, updu, cu, avatar_path, gender, title) FROM
stdin;
```

```
\N \N \N test2
937e8d5fbb48bd4949536cd65b8d35c426b80d2f830c5c308e2cdec422ae2244
\N test@test2.test \N \N 2021-05-23 14:38:04.018 2021-
05-23 14:38:04.018 \N 1 \N f7eae3c-3643-4b2f-850b-
6bf268652b8e f7eae3c-3643-4b2f-850b-6bf268652b8e \N \N \N
\N
test test test test
937e8d5fbb48bd4949536cd65b8d35c426b80d2f830c5c308e2cdec422ae2244
\N test@test.test test test 2021-05-23 15:01:08.611 2021-
05-23 15:01:08.611 2021-05-19 12:42:19.936668 4 t 814d08b6-
b886-11eb-9a34-7f0ea57a76dd 814d08b6-b886-11eb-9a34-7f0ea57a76dd
\N \N Male \N
Павел \N Бесчетвертев aVoid
dff3f9f8bce8f7c9de2a373d233020d36c10d265275d7e29c027d30a3f95c058
\N avoidpnz@gmail.com \N \N 2021-06-01 09:04:01.015
2021-06-01 09:04:01.015 \N 332 \N 9a8db856-16b3-469e-
85d0-8f408b2441b7 9a8db856-16b3-469e-85d0-8f408b2441b7 \N
/9a8db856-16b3-469e-85d0-8f408b2441b7/dokument1.png Male \N
\.
```

```
--
-- Data for Name: user_role; Type: TABLE DATA; Schema: public;
Owner: postgres
--
```

```
COPY public.user_role (name, priority, updt, ct, ver,
is_first_login, uuid, updu, cu, "user", role) FROM stdin;
```

```
\N 0 2021-05-22 09:04:46 2021-05-23 16:29:06.319979 8 t
9a8db856-16b3-469e-85d0-8f408b2441b7 9a8db856-16b3-469e-85d0-
8f408b2441b7 \N 814d08b6-b886-11eb-9a34-7f0ea57a76dd Registered
\N 1 2021-05-23 14:37:16.426 \N 0 t 32868917-5a99-
411d-a058-f8d2a28ce220 9a8db856-16b3-469e-85d0-8f408b2441b7 \N
f7eae3c-3643-4b2f-850b-6bf268652b8e Admin
\N 0 2021-05-25 14:39:50.292 \N 12 t 8e7ba102-5043-
4951-829b-b2220fd150e5 9a8db856-16b3-469e-85d0-8f408b2441b7 \N
9a8db856-16b3-469e-85d0-8f408b2441b7 Registered
```

```
test      0      2021-05-31 18:32:55.41      2021-05-19 14:11:56.018458 9
      t      05b25dfc-b893-11eb-a97b-cbd267af08b1 9a8db856-16b3-469e-
85d0-8f408b2441b7      \N      9a8db856-16b3-469e-85d0-8f408b2441b7 Admin
\.
```

Приложение Г
ЛИСТИНГ ХРАНИМЫХ ПРОЦЕДУР
(рекомендуемое)

Листинг Г.1 – Процедура update_soundtrack.

```
--
-- Name: update_soundtrack(); Type: FUNCTION; Schema: public; Owner:
postgres
--

CREATE FUNCTION public.update_soundtrack() RETURNS trigger
    LANGUAGE plpgsql
    AS $$

DECLARE max_order integer;

DECLARE rec RECORD;

BEGIN

    select max("order") into max_order from soundtrack;

    if NEW."order" is null then RAISE EXCEPTION 'order must have a
value'; end if;

    if NEW."order" < 1 then RAISE EXCEPTION 'order (%) must be >= 1',
NEW."order"; end if;

    if NEW."order" > max_order then RAISE EXCEPTION 'order (%) must be
<= max(order) (%)', NEW."order", max_order; end if;

    -- Temporarily, move the current record at the old position "out
of the way"

    -- Don't forget the other columns

UPDATE soundtrack set

    "order" = 0,

    "comment" = new."comment",

    is_active = new.is_active,

    ver = new.ver,

    updt = new.updt,

    updu = new.updu

    where "order" = OLD."order";

    if NEW."order" > OLD."order" then

        FOR rec IN

            select "order" from soundtrack

            where "order" > OLD."order" and "order" <= NEW."order"
```

```

        order by "order" ASC

    LOOP

        UPDATE soundtrack set "order" =  "order" - 1, ver = ver +
1 where "order" = rec."order";

        END LOOP;

    else

        FOR rec IN

            select "order" from soundtrack

            where "order" >= NEW."order" and "order" < OLD."order"

            order by "order" DESC

        LOOP

            UPDATE soundtrack set "order" =  "order" + 1, ver = ver +
1 where "order" = rec."order";

            END LOOP;

        end if;

        -- Put the current row back to the new position

        UPDATE soundtrack set "order" = NEW."order", ver = ver + 1 where
"order" = 0;

        RETURN NULL;      -- DO NOT PROCEED

END;

$$;

```

Листинг Г.2 – Процедура update_playlist.

```

--
-- Name: update_playlist(); Type: FUNCTION; Schema: public; Owner:
postgres
--

CREATE FUNCTION public.update_playlist() RETURNS trigger
    LANGUAGE plpgsql
    AS $$

DECLARE max_order integer;

DECLARE rec RECORD;

BEGIN

    select max("order") into max_order from soundtrack;

```

```

        if NEW."order" is null then RAISE EXCEPTION 'order must have a
value'; end if;

        if NEW."order" < 1 then RAISE EXCEPTION 'order (%) must be >= 1',
NEW."order"; end if;

        if NEW."order" > max_order then RAISE EXCEPTION 'order (%) must be
<= max(order) (%)', NEW."order", max_order; end if;

-- Temporarily, move the current record at the old position "out
of the way"

-- Don't forget the other columns

UPDATE soundtrack set

    "order" = 0,

    is_active = new.is_active,

    ver = new.ver,

    updt = new.updt,

    updu = new.updu

where "order" = OLD."order";

if NEW."order" > OLD."order" then

    FOR rec IN

        select "order" from soundtrack

        where "order" > OLD."order" and "order" <= NEW."order"

        order by "order" ASC

    LOOP

        UPDATE soundtrack set "order" = "order" - 1, ver = ver +
1 where "order" = rec."order";

    END LOOP;

else

    FOR rec IN

        select "order" from soundtrack

        where "order" >= NEW."order" and "order" < OLD."order"

        order by "order" DESC

    LOOP

```

```

        UPDATE soundtrack set "order" = "order" + 1, ver = ver +
1 where "order" = rec."order";

        END LOOP;

    end if;

    -- Put the current row back to the new position

    UPDATE soundtrack set "order" = NEW."order", ver = ver + 1 where
"order" = 0;

    RETURN NEW;      -- DO NOT PROCEED

END;

$$;

```

Листинг Г.3 – Процедура insert_soundtrack.

```

--
-- Name: insert_soundtrack(); Type: FUNCTION; Schema: public; Owner:
postgres
--

CREATE FUNCTION public.insert_soundtrack() RETURNS trigger
    LANGUAGE plpgsql
    AS $$

DECLARE max_order integer;

DECLARE rec RECORD;

BEGIN

    if NEW."order" < 1 then RAISE EXCEPTION 'order (%) must be >= 1)',
NEW."order"; end if;

    select max("order") into max_order from soundtrack;

    if max_order is null then

        NEW."order" = 1;

    elsif NEW."order" > max_order + 1 then

        RAISE EXCEPTION 'order (%) must be <= max(order) + 1 (%)',
NEW."order", max_order + 1;

    else

        FOR rec IN select "order" from soundtrack where "order" >=
NEW."order" order by "order" DESC LOOP

```

```

        UPDATE soundtrack set "order" = "order" + 1, ver = ver + 1
where "order" = rec."order";

        END LOOP;

    end if;

    RETURN NEW;

END;

$$;

```

Листинг Г.4 – Процедура delete_soundtrack.

```

--
-- Name: delete_soundtrack(); Type: FUNCTION; Schema: public; Owner:
postgres
--

CREATE FUNCTION public.delete_soundtrack() RETURNS trigger
    LANGUAGE plpgsql
    AS $$

DECLARE rec RECORD;

BEGIN

    FOR rec IN select "order" from soundtrack where "order" >
OLD."order" order by "order" ASC LOOP

        UPDATE soundtrack set "order" = "order" - 1, ver = ver +
1 where "order" = rec."order";

        END LOOP;

    RETURN OLD;

END;

$$;

```

Приложение Д
ЛИСТИНГ ТРИГГЕРОВ
(рекомендуемое)

Листинг Д.1 - Триггер delete_soundtrack_trigger

```
--
-- Name: soundtrack delete_soundtrack_trigger; Type: TRIGGER; Schema:
public; Owner: postgres
--

CREATE TRIGGER delete_soundtrack_trigger AFTER DELETE ON
public.soundtrack FOR EACH ROW WHEN ((pg_trigger_depth() = 0)) EXECUTE
FUNCTION public.delete_soundtrack();
```

Листинг Д.2 – Триггер insert_soundtrack_trigger

```
--
-- Name: soundtrack insert_soundtrack_trigger; Type: TRIGGER; Schema:
public; Owner: postgres
--

CREATE TRIGGER insert_soundtrack_trigger BEFORE INSERT ON
public.soundtrack FOR EACH ROW WHEN ((pg_trigger_depth() = 0)) EXECUTE
FUNCTION public.insert_soundtrack();
```

Листинг Д.3 – Триггер update_playlist_trigger

```
--
-- Name: playlist update_playlist_trigger; Type: TRIGGER; Schema:
public; Owner: postgres
--

CREATE TRIGGER update_playlist_trigger INSTEAD OF UPDATE ON
public.playlist FOR EACH ROW EXECUTE FUNCTION
public.update_playlist();
```

Листинг Д.4 – Триггер update_soundtrack_trigger

```
--
-- Name: soundtrack update_soundtrack_trigger; Type: TRIGGER; Schema:
public; Owner: postgres
--

CREATE TRIGGER update_soundtrack_trigger BEFORE UPDATE ON
public.soundtrack FOR EACH ROW WHEN ((pg_trigger_depth() = 0)) EXECUTE
FUNCTION public.update_soundtrack();
```

Приложение Е
РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ
(рекомендуемое)

	genre	artist	title	album	year	feedbacks	avg_rating
1	amil-- (Pop,Dance,Other)	Ramil'	Аромат	Громкие новинки Декабря 2	2020	0	[NULL]
2	dj-antc (Pop,Dance,Other)	Dj Antonio, Bertie Scott	Losing My Religion	Громкие новинки Декабря 2	2020	0	[NULL]
3	armak (Pop,Dance,Other)	YARMAK	Чё там у хохлов	Громкие новинки Декабря 2	2020	0	[NULL]
4	markul (Pop,Dance,Other)	KYOK, Markul	Тренировочный день	Громкие новинки Декабря 2	2020	0	[NULL]
5	anna- (Pop,Dance,Other)	XAHHA	СТАТУС ВЛЮБЛЕНА	Громкие новинки Декабря 2	2020	0	[NULL]
6	slame- (Pop,Dance,Other)	HENSY, Slame	Друг помоги мне	Громкие новинки Декабря 2	2020	0	[NULL]
7	rakhim (Pop,Dance,Other)	Rakhim	Big Money	Громкие новинки Декабря 2	2020	0	[NULL]
8	anivar- (Pop,Dance,Other)	ANIVAR	Высоко	Громкие новинки Декабря 2	2020	0	[NULL]
9	david- (Pop,Dance,Other)	David Guetta, MORTEN, Lanie Gardner	Dreams	Громкие новинки Декабря 2	2020	0	[NULL]
10	elvira-t (Pop,Dance,Other)	Elvira T	Меланхолия	Громкие новинки Декабря 2	2020	0	[NULL]
11	juice-w (Pop,Dance,Other)	Juice WRLD, The Kid LAROI	Reminds Me Of You	Громкие новинки Декабря 2	2020	1	2
12	mia-bc (Pop,Dance,Other)	Mia Boyka	АУФ	Громкие новинки Декабря 2	2020	1	1

Рисунок Е.1 – Результат тестирования запроса

	uuid	soundtracks	login	last_offline	avatar_path	phone
1	9a8db856-16b3-469e-85d0-8f408b2441b7	12	aVoid	2021-06-01 09:04:01	/9a8db856-16b3-469e-85d0-8f408b2441b7/dokument1.png	[NULL]
2	814d08b6-b886-11eb-9a34-7f0ea57a76dd	0	test	2021-05-23 15:01:08	[NULL]	test
3	f7eae3c-3643-4b2f-850b-6bf268652b8e	0	test2	2021-05-23 14:38:04	[NULL]	[NULL]

Рисунок Е.2 – Результат тестирования запроса

	uuid	order	title	artist	album	genre
1	cd5616a8-02cb-4517-81f6-6ce3b838829e	8	Тренировочный день	KYOK, Markul	Громкие новинки Декабря 2	{Pop,Dance,Other}
2	bb3c1176-01bd-4327-9253-add309364574	2	Друг помоги мне	HENSY, Slame	Громкие новинки Декабря 2	{Pop,Dance,Other}
3	cb361d80-c3ae-43c0-b8da-49550ec05e8e	6	СТАТУС ВЛЮБЛЕНА	XAHHA	Громкие новинки Декабря 2	{Pop,Dance,Other}
4	5d7f40c9-ed63-4398-9165-ad1b21d97d68	9	Аромат	Ramil'	Громкие новинки Декабря 2	{Pop,Dance,Other}
5	f83e4381-1617-4151-83bb-11abd44c8a23	12	АУФ	Mia Boyka	Громкие новинки Декабря 2	{Pop,Dance,Other}
6	87d5575d-3320-4542-b717-4f3973f6a6a6	3	Высоко	ANIVAR	Громкие новинки Декабря 2	{Pop,Dance,Other}
7	f8ac10e9-3ee9-46c1-979a-6025ed8562bc	10	Dreams	David Guetta, MORTEN, Lanie Gardner	Громкие новинки Декабря 2	{Pop,Dance,Other}
8	61aea7e0-b94c-413f-8468-303b31493e49	4	Big Money	Rakhim	Громкие новинки Декабря 2	{Pop,Dance,Other}
9	df92a5a8-cec0-4585-9010-7a9f7b33cb8e	5	Losing My Religion	Dj Antonio, Bertie Scott	Громкие новинки Декабря 2	{Pop,Dance,Other}
10	2c1f097a-9b0f-4a49-bbb3-11efa28c4dc0	1	Reminds Me Of You	Juice WRLD, The Kid LAROI	Громкие новинки Декабря 2	{Pop,Dance,Other}
11	8059141c-de34-449f-a8ad-958c35cbf1e7	7	Чё там у хохлов	YARMAK	Громкие новинки Декабря 2	{Pop,Dance,Other}
12	934a2f8e-955c-4797-a172-ef6bd68f84e0	11	Меланхолия	Elvira T	Громкие новинки Декабря 2	{Pop,Dance,Other}

Рисунок Е.3 – Результат тестирования представления

	uuid	login	last_offline	avatar_path	birthday	first_name
1	814d08b6-b886-11eb-9a34-7f0ea57a76dd	test	2021-05-23 15:01:08	[NULL]	[NULL]	test
2	f7eae3c-3643-4b2f-850b-6bf268652b8e	test2	2021-05-23 14:38:04	[NULL]	[NULL]	[NULL]
3	9a8db856-16b3-469e-85d0-8f408b2441b7	aVoid	2021-06-01 09:04:01	/9a8db856-16b3-469e-85d0-8f408b2441b7/dokument	[NULL]	Павел

Рисунок Е.4 – Результат тестирования представления

	file_size	duration	order	sample_rate	bit_rate	file_format	comment	updt	ct
1	3 841 500	95	13	44 100	320 000	mp3	1234567	2021-05-27 14:06:59	2021-05-27 14:06:34
2	6 805 888	169	12	44 100	320 000	mp3	1234	2021-05-31 18:54:16	2021-05-31 18:54:16
3	8 206 097	204	11	44 100	320 000	mp3	[NULL]	2021-05-25 14:35:52	2021-05-24 00:24:40
4	6 057 729	150	10	44 100	320 000	mp3	[NULL]	2021-05-27 11:32:34	2021-05-24 22:14:48
5	7 801 700	194	9	44 100	320 000	mp3	ppp	2021-05-24 21:35:29	2021-05-24 17:36:13
6	6 685 731	166	8	44 100	320 000	mp3	[NULL]	2021-05-24 21:57:21	2021-05-24 20:03:06
7	6 314 790	156	7	44 100	320 000	mp3	[NULL]	2021-05-25 14:35:58	2021-05-24 19:59:49
8	9 984 515	248	6	44 100	320 000	mp3	[NULL]	2021-05-25 21:01:28	2021-05-24 21:51:51
9	5 339 890	132	5	44 100	320 000	mp3	[NULL]	2021-05-31 10:55:26	2021-05-24 19:54:07
10	9 050 317	225	4	44 100	320 000	mp3	[NULL]	2021-05-31 10:57:27	2021-05-24 00:21:15
11	6 657 530	165	3	44 100	320 000	mp3	[NULL]	2021-05-31 10:56:44	2021-05-24 21:50:33
12	6 362 899	158	2	44 100	320 000	mp3	[NULL]	2021-05-31 15:51:13	2021-05-24 00:22:57
13	[NULL]	[NULL]	1	[NULL]	100	test	[NULL]	2021-06-01 12:13:21	2021-06-01 12:13:21

Рисунок Е.5 – Результат работы триггера insert_soundtrack_trigger

	123 file_size 📄 ⬆️	123 duration 📄 ⬆️	123 order 📄 ⬆️	123 sample_rate 📄 ⬆️	123 bit_rate 📄 ⬆️	ABC file_format 📄 ⬆️	ABC comment 📄 ⬆️	🕒 updt 📄 ⬆️	🕒 ct 📄 ⬆️
1	6 657 530	165	2	44 100	320 000	mp3	[NULL]	2021-05-31 10:56:44	2021-05-24 21:50:33
2	3 841 500	95	13	44 100	320 000	mp3	1234567	2021-05-27 14:06:59	2021-05-27 14:06:34
3	6 805 888	169	12	44 100	320 000	mp3	1234	2021-05-31 18:54:16	2021-05-31 18:54:16
4	9 050 317	225	3	44 100	320 000	mp3	[NULL]	2021-05-31 10:57:27	2021-05-24 00:21:15
5	8 206 097	204	11	44 100	320 000	mp3	[NULL]	2021-05-25 14:35:52	2021-05-24 00:24:40
6	5 339 890	132	4	44 100	320 000	mp3	[NULL]	2021-05-31 10:55:26	2021-05-24 19:54:07
7	9 984 515	248	5	44 100	320 000	mp3	[NULL]	2021-05-25 21:01:28	2021-05-24 21:51:51
8	6 057 729	150	10	44 100	320 000	mp3	[NULL]	2021-05-27 11:32:34	2021-05-24 22:14:48
9	7 801 700	194	9	44 100	320 000	mp3	ppp	2021-05-24 21:35:29	2021-05-24 17:36:13
10	6 685 731	166	8	44 100	320 000	mp3	[NULL]	2021-05-24 21:57:21	2021-05-24 20:03:06
11	6 314 790	156	7	44 100	320 000	mp3	[NULL]	2021-05-25 14:35:58	2021-05-24 19:59:49
12	6 362 899	158	1	44 100	320 000	mp3	[NULL]	2021-05-31 15:51:13	2021-05-24 00:22:57
13	[NULL]	[NULL]	6	[NULL]	100	test	[NULL]	2021-06-01 12:13:21	2021-06-01 12:13:21

	123 file_size 📄	123 duration 🕒	123 order 📄	123 sample_rate 📄	123 bit_rate 📄	abc file_format 📄	abc comment 📄	🕒 updt 🕒	🕒 ct 🕒
1	6 657 530	165	2	44 100	320 000	mp3	[NULL]	2021-05-31 10:56:44	2021-05-24 21:50:33
2	9 050 317	225	3	44 100	320 000	mp3	[NULL]	2021-05-31 10:57:27	2021-05-24 00:21:15
3	5 339 890	132	4	44 100	320 000	mp3	[NULL]	2021-05-31 10:55:26	2021-05-24 19:54:07
4	9 984 515	248	5	44 100	320 000	mp3	[NULL]	2021-05-25 21:01:28	2021-05-24 21:51:51
5	6 314 790	156	6	44 100	320 000	mp3	[NULL]	2021-05-25 14:35:58	2021-05-24 19:59:49
6	6 685 731	166	7	44 100	320 000	mp3	[NULL]	2021-05-24 21:57:21	2021-05-24 20:03:06
7	7 801 700	194	8	44 100	320 000	mp3	ppp	2021-05-24 21:35:29	2021-05-24 17:36:13
8	6 057 729	150	9	44 100	320 000	mp3	[NULL]	2021-05-27 11:32:34	2021-05-24 22:14:48
9	8 206 097	204	10	44 100	320 000	mp3	[NULL]	2021-05-25 14:35:52	2021-05-24 00:24:40
10	6 805 888	169	11	44 100	320 000	mp3	1234	2021-05-31 18:54:16	2021-05-31 18:54:16
11	3 841 500	95	12	44 100	320 000	mp3	1234567	2021-05-27 14:06:59	2021-05-27 14:06:34
12	6 362 899	158	1	44 100	320 000	mp3	[NULL]	2021-05-31 15:51:13	2021-05-24 00:22:57

200 OK

83.73 ms

DETAILS ^

POST

http://127.0.0.1/api/user/getToken

Response headers 2

Request headers 1

Redirects 0

Timings

content-type: application/json

Source message

<>

token

eyJhbGciOiJSUzI1NiJ9.eyJ1c2VySwQioiI5YThkYjg1Ni0xNmIzLTQ2W0t0DVkMc04ZjQwOGYlbnQxYjciLCJ1c2lkIjoiaGU3YmExMDItNTA0My00TUxLTGyOWItYjYlMjBmZDE1MGU1Iiwicm9sZS5hr_LSeQplggUQrd8FTB8Wgr-DQpGnK8X4yQU53W09oyyoH7QJ_0S28do58TODhlxH3tfrCLEksU8Nhr79qCGyZPTfT4gGa4fgxdNQ1pZv8rAHeL023AmJmpWuLH-lD0UEPmPnyBymVxaAKPrYn9iW0DbExK2R4sPX13jPwenZ_92O3qJ2l0mn-f6pgL2TSyP9AJEb6govhvgkglPQ9te5o5wDoFo3tAcRmwyOrFeE-g738rFgBxVmgH9n7JLTEAfBDacnlvPxOxKf6-3gwlu5U0U8P3PspYCuWofV0IPL4YexrEJEROEc3fdRlCRO7Aaye3gHWP1V8KePhRxoA0lh2P486j0XhM_QxXZSB-R8M5FA9mJgpJ3QJScNgcw82jSQxsqBFEPaxKp5Ahh-OQG7M9Q-IsKieBlAXp9Klul7rsufD0nb3peJkPqQC2b3l0SQTDHZal_wAdAl4MVdkSUOGnKfJKin56dvF6c7JhzYXpX7LQZHRy9fK0tFHxbKwtw9Za8xCZ1exbhuNTCHN7Em24nb2TghYowaEhVkrZ8KdulJeJXKKYEuR_F4Q1GApbr2ni3Aph8Kvg2m2n7mN3U9r1GL0N10IH8Mtpqjyvc09Ecg

isFirstLogin

true

Рисунок Е.9 – Тестирование сервиса авторизации

200 OK 94.55 ms DETAILS ^

GET http://127.0.0.1/api/user/currentUser

Response headers 9 Request headers 2 Redirects 0 Timings

```

content-type: application/json
t eyJhbGciOiJSUzI1NiJ9.eyJ1c2VySWQiOiI5YThkYjg1Ni0xNmIzLTQ2OWUtODVhMC04ZjQwOGIyNDQxYjciLCJ1c2lkIjoia0GU3YmExMDItNTA0My00OTUxLTgyOWItVjIy
o MjBmZDEiMGU1Iiwicm9sZSI6IjI1Z2ZldGVyZWQlLCJpYXQ1OjE2MjI2MTYsImV4cCI6MTYyMjg3NTMzMH0.fSRhyYPrSkacmEF5qtn0iTaTSR3v6D3XKwUrA_UR0cL6qSqd
k aE9n-vV6tS_dzS0jzy200RrVzK97raIEhHfrJiaf80f5XYrG8tx2xdnYjVtn11s1aHs1IPPZ7qxHYVCNE1ZrMh7vM_sSMrO-UAw4TOu4loQrVDCr21dzJOLm1fppY1cB_Vf-
e 5KkaQB6lRw-y3i9QoqGbHcVDsa52AKb8HoSwOWXo3g4wHzT4SH24Zh5078NsMIyZABF2erokNwawWubyNMY2duk0RFAKX_Z1cUrfTse-
n kpkzwnF4hZvrsY3M5geFn7Wmnf2wvOT0GpN8VHbDpZ1dy0-xs1-FemQz6eRdoQsseOvcdRjLqSpL73ZAeoCMZKe0De1VH4G91s4xrqV6qcH9NC4F-
: _xQfB1YuwNOuIaXUY2PKuNVBLNVCKIf18Kwe4cwK8seZYPVcVnKs6j8Lu9Igl0yUPY94ZSP-bv5KFxjSI74e10n0FZBrPtg74EOPyWvAGW64quL8pa9tCSuWNaLP0-
qFXepLqMuEK-RBIZQLSL65UTN1A4wx1CjMLt04CG3noqAsEiv801KFTkrNX8cvDh1ABtgyoRbPWRMGBcf78xwWZgrWkQNM3f4rK0fWKGZo9yzLEkqo8-
aIPvKERs0FFog5yN0mbOLxkCFN03NU4Lf5P0fbb8

```

Source message ▾

id 9a8db856-16b3-469e-85d0-8f408b2441b7

ver 339

Рисунок Е.10 – Тестирование сервиса запроса пользователя

200 OK 30.77 ms DETAILS ^

POST http://127.0.0.1/api/user/register

Response headers 9 Request headers 1 Redirects 0 Timings

```

content-type: application/json

```

Source message ▾

```

{
  "id": "9c26e931-d6dd-4a13-b59b-71871479cc16",
  "ver": 0,
  "updt": 1622616487866,
  "login": "test",
  "email": "test.json@test.ru",
  "userRoles": [Array[1]]
  - 0: {
    "id": "4805dac8-918b-4f34-b955-393aa39d3440",
    "ver": 0,
    "updt": 1622616487867,
    "priority": 1,
    "role": "Registered",
    "user": "9c26e931-d6dd-4a13-b59b-71871479cc16",
    "firstLogin": true
  }
}

```

Рисунок Е.11 – Тестирование сервиса регистрации

Source message 

id	ver	updt	cu	order	isActive	title	artist	album	genre
61aea7e0-b94c-413f-8468-303b31493e49	69	1622447726418	9a8db856-16b3-469e-85d0-8f408b2441b7	4	true	Big Money	Rakhim	Громкие новинки Декабря 2	Pop, Dance, Other
df92a5a8-cec0-4585-9010-7a9f7b33cb8e	35	1621965688027	9a8db856-16b3-469e-85d0-8f408b2441b7	5	true	Losing My Religion	Dj Antonio, Bertie Scott	Громкие новинки Декабря 2	Pop, Dance, Other
8059141c-de34-449f-a8ad-958c35cbf1e7	41	1621882641621	9a8db856-16b3-469e-85d0-8f408b2441b7	7	true	Чё там у хохлов	YARMAK	Громкие новинки Декабря 2	Pop, Dance, Other
cd5616a8-02cb-4517-b000-000000000000	28	1621881329810	9a8db856-16b3-469e-85d0-8f408b2441b7	8	true	Тренировочный пень	KYOK, Maskul	Громкие новинки Декабря 2	Pop, Dance, Other

200 OK 83.17 ms

Source message 

name	position
value	104780

Рисунок Е.13 – Тестирование сервиса изменения позиции
воспроизведения песни

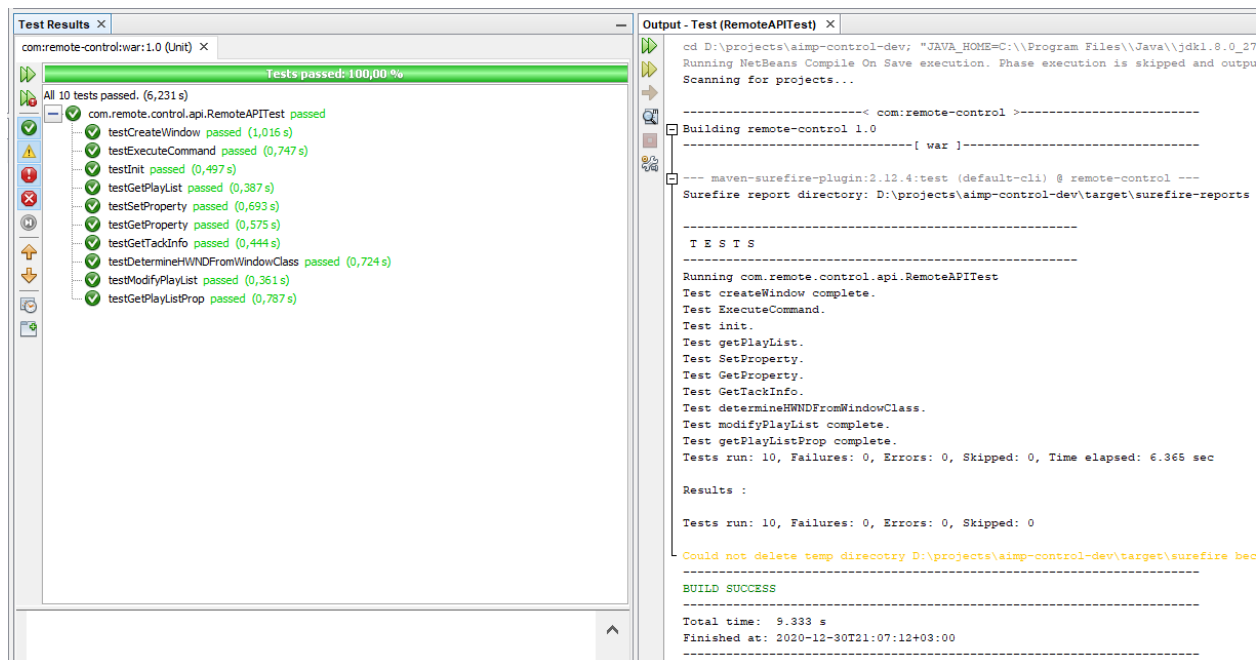


Рисунок Е.14 – Результаты модульного тестирования

* Email

* Login

* Password

* Confirmation password

Рисунок Е.15 – Тестирование регистрации

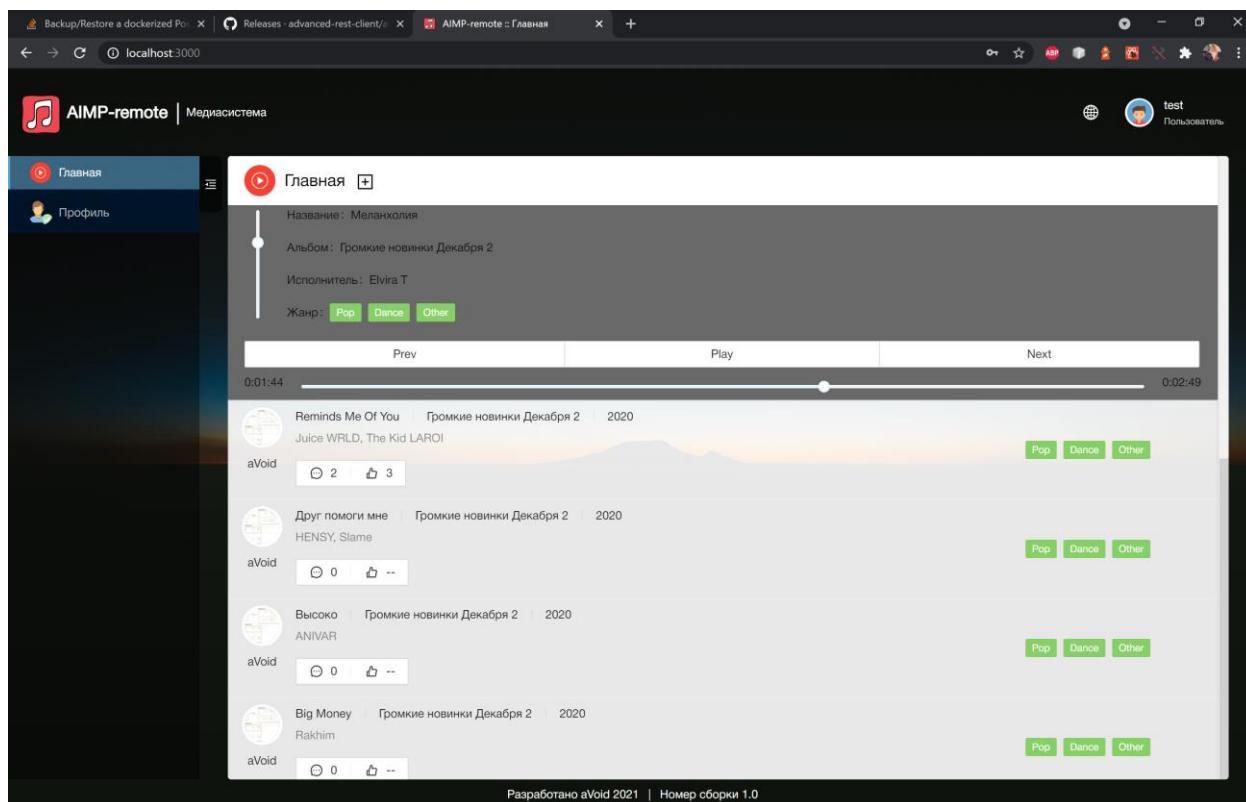


Рисунок Е.16 – Тестирование авторизации



Рисунок Е.17 – Тестирование загрузки плейлиста

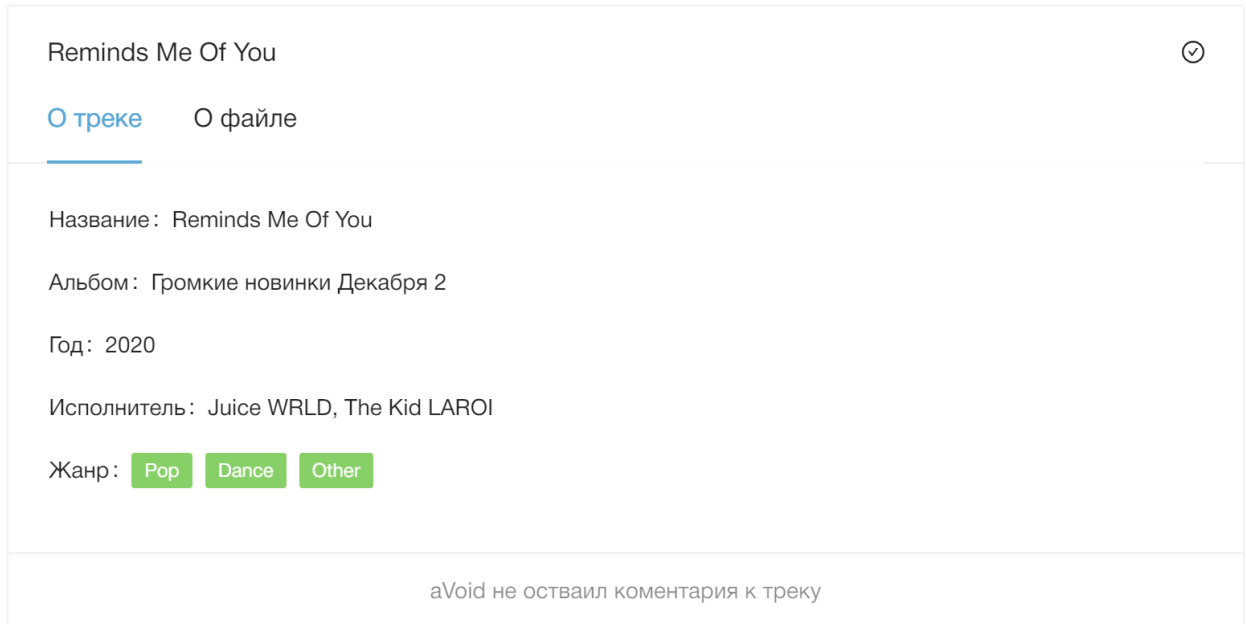


Рисунок Е.18 – Тестирование загрузки подробной информации о песне

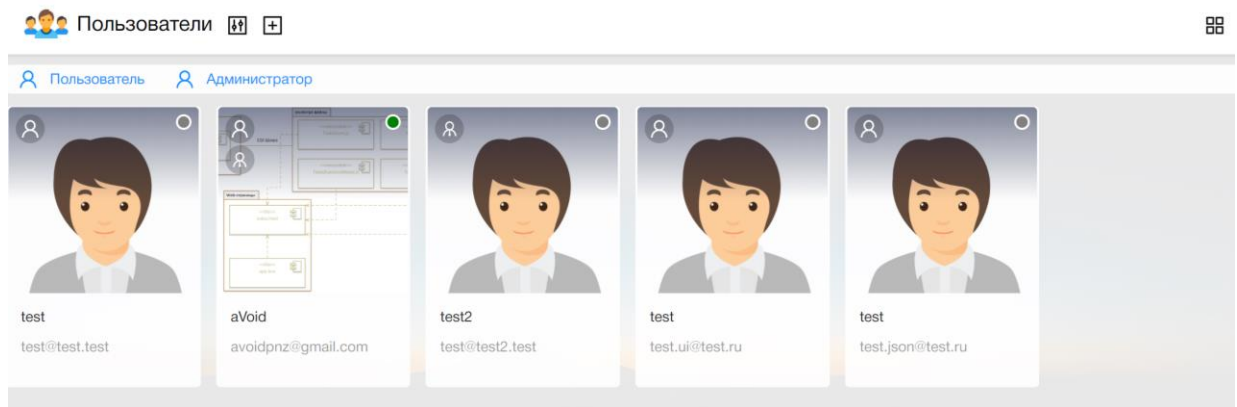


Рисунок Е.19 – Тестирование загрузки списка пользователей



Рисунок Е.20 – Тестирование исключения песни из плейлиста



Рисунок Е.21 – Тестирование изменения громкости воспроизведения плеера

Приложение Ж
СПЕЦИФИКАЦИЯ СЕРВИСОВ СЕРВЕРНОГО ПРИЛОЖЕНИЯ
(рекомендуемое)

Таблица Ж. 1 – Спецификация веб-сервисов

Название	Метод	Параметры	Возвращаемые данные	Краткое описание
control/next	GET	Заголовок запроса: токен пользователя	DTO описание воспроизводи мой песни	Метод позволяет переключить воспроизведение к следующей песни
control/prev	GET	Заголовок запроса: токен пользователя	DTO описание воспроизводи мой песни	Метод позволяет переключить воспроизведение к предыдущей песни
control/play pause	GET	Заголовок запроса: токен пользователя	DTO состояние плеера	Метод позволяет изменить состояние воспроизведения плеера
control/volu me	GET	Заголовок запроса: токен пользователя	DTO описание свойства плеера	Метод позволяет получить текущее значение громкости воспроизведения
control/play erState	GET	Заголовок запроса: токен пользователя	DTO состояние плеера	Метод позволяет получить текущее состояние воспроизведения плеера
control/volu me/{value}	PUT	Заголовок запроса: токен пользователя URL параметр: Новое значение громкости воспроизведения	DTO описание свойства плеера	Метод позволяет установить новое значение громкости воспроизведения плеера
control/soun dtrackPositi on	GET	Заголовок запроса: токен пользователя	DTO описание свойства плеера	Метод позволяет получить текущую позицию воспроизведения песни
control/soun dtrackPositi on/ {value}	PUT	Заголовок запроса: токен пользователя URL параметр: новое значение позиции воспроизведения	DTO описание свойства плеера	Метод позволяет установить новую позицию воспроизведения песни
control/soun dtrackDurat ion	GET	Заголовок запроса: токен пользователя	DTO описание свойства плеера.	Метод позволяет получить продолжительность воспроизведения текущей песни
control/soun dtrackInfo	GET	Заголовок запроса: токен пользователя	DTO описание воспроизводи мой песни	Метод позволяет получить описание текущей воспроизводимой песни
feedback/{i d}	GET	Заголовок запроса: токен пользователя URL параметр: Id отзыва	DTO отзыва о песне	Метод позволяет получить отзыв о песне по id отзыва

Продолжение таблицы Ж.1

Название	Метод	Параметры	Возвращаемые данные	Краткое описание
feedback/by Soundtrack/{soundtrackId}	GET	Заголовок запроса: токен пользователя URL параметр: Id песни	Список DTO отзывов о песне	Метод позволяет получить список отзывов о песне по id песни
feedback/addFeedback	POST	Заголовок запроса: токен пользователя Тело запроса: Объект отзыва в JSON формате	DTO отзыва о песне	Метод позволяет добавить в систему новый отзыв о песне.
feedback/{id}	DELETE	Заголовок запроса: токен пользователя URL параметр: Id отзыва	—	Метод позволяет удалить из системы отзыв о песне по id отзыва
feedback	GET	Заголовок запроса: токен пользователя	Список DTO отзывов о песне	Метод позволяет получить все отзывы о песнях, существующие в системе
file/{uploadFile}	POST	Заголовок запроса: токен пользователя, название загружаемого файла Тело запроса: Поток данных файла	Путь до загруженного файла	Метод позволяет загрузить в систему новый файл
playlist/{id}	GET	Заголовок запроса: токен пользователя URL параметр: Id песни	DTO элемента плейлиста	Метод позволяет получить элемент плейлиста по его id
playlist/activate/{trackId}	GET	Заголовок запроса: токен пользователя URL параметр: Id песни	DTO элемента плейлиста	Метод позволяет добавить песню в активный плейлиста по id песни
playlist/deactivate/{trackId}	GET	Заголовок запроса: токен пользователя URL параметр: Id песни	DTO элемента плейлиста	Метод позволяет исключить песню из активного плейлиста по id песни
playlist/move/{trackId}/{to}	GET	Заголовок запроса: токен пользователя URL параметр: Id песни, конечная позиция в плейлисте	DTO элемента плейлиста	Метод позволяет переместить песню с заданным id на нужную позицию в плейлисте. Также производится обновление позиций других элементов плейлиста, в случае необходимости
playlist	GET	Заголовок запроса: токен пользователя	Список DTO элементов плейлиста	Метод позволяет получить полный список плейлиста

Продолжение таблицы Ж.1

Название	Метод	Параметры	Возвращаемые данные	Краткое описание
soundtrack/ save	POST	Заголовок запроса: токен пользователя Тело запроса: Объект песни в JSON формате	DTO объект песни	Метод позволяет добавить новую песню в систему
soundtrack/ {id}	GET	Заголовок запроса: токен пользователя URL параметр: Id песни	DTO объект песни	Метод позволяет получить песню по ее id
soundtrack	GET	Заголовок запроса: токен пользователя	Список DTO объектов песен	Метод позволяет получить все песни, загруженные в систему
uptimeTest	GET	—	—	Метод возвращает актуальное состояние серверного приложения
userRole/ changeCurrent/{userId}	GET	Заголовок запроса: токен пользователя URL параметр: Id роли пользователя	DTO объект роли пользователя	Метод позволяет переключаться пользователю между доступными ему ролями
userRole/current	GET	Заголовок запроса: токен пользователя	DTO объект роли пользователя	Метод возвращает объект текущей роли пользователя
userRole	PUT	Заголовок запроса: токен пользователя Тело запроса: Объект роли пользователя в JSON формате	DTO объект роли пользователя	Метод позволяет редактировать роль пользователя
userRole/{id}	DELETE	Заголовок запроса: токен пользователя URL параметр: Id роли пользователя	—	Метод позволяет удалить роль пользователя
userRole/{id}	GET	Заголовок запроса: токен пользователя URL параметр: Id роли пользователя	DTO объект роли пользователя	Метод возвращает объект роли пользователя по ее id
userRole/findAllByUser/{id}	GET	Заголовок запроса: токен пользователя URL параметр: Id пользователя	Список DTO объектов ролей пользователя	Метод позволяет получить список всех ролей для пользователя по id пользователя

Продолжение таблицы Ж.1

Название	Метод	Параметры	Возвращаемые данные	Краткое описание
userRole/addRole	POST	Заголовок запроса: токен пользователя Тело запроса: Объект роли пользователя в JSON формате	DTO объект роли пользователя	Метод позволяет добавить новую роль пользователю
userRole/getMainUserRole/{id}	GET	Заголовок запроса: токен пользователя URL параметр: Id пользователя	DTO объект роли пользователя	Метод возвращает основную роль (роль с максимальным приоритетом) пользователя по id пользователя
user/currentUser	GET	Заголовок запроса: токен пользователя	DTO объект пользователя	Метод возвращает объект текущего пользователя
user/findByEmail/{email}	GET	URL параметр: email пользователя	DTO объект пользователя	Метод позволяет выполнить поиск пользователей по email
user/setLastOffline	GET	Заголовок запроса: токен пользователя	DTO объект пользователя	Метод устанавливает метку последнего выхода их системы для текущего пользователя
user/editProfile	PUT	Заголовок запроса: токен пользователя Тело запроса: Объект пользователя в JSON формате	DTO объект пользователя	Метод позволяет редактировать личные данные для текущего пользователя
user/editProfile/{userId}	PUT	Заголовок запроса: токен пользователя URL параметр: Id пользователя Тело запроса: Объект пользователя в JSON формате	DTO объект пользователя	Метод позволяет редактировать личные данные пользователя по id пользователя
user/create	POST	Заголовок запроса: токен пользователя Тело запроса: Объект пользователя в JSON формате	DTO объект пользователя	Метод позволяет создать нового пользователя в системе с возможностью выбора необходимой роли для пользователя

Окончание таблицы Ж.1

Название	Метод	Параметры	Возвращаемые данные	Краткое описание
user/register	POST	Тело запроса: Объект пользователя в JSON формате	DTO объект пользователя	Метод позволяет зарегистрироваться новым пользователям в системе
user/getToken	POST	Тело запроса: Объект с данными для авторизации пользователя в JSON формате	DTO токена пользователя	Метод возвращающий токен пользователя. Используется для авторизации в системе
user/changePass/{userId}	PUT	Заголовок запроса: токен пользователя URL параметр: Id пользователя Тело запроса: Строка sha256 хеша пароля в JSON формате	DTO объект пользователя	Метод позволяет изменить пароль для пользователя по id пользователя
user/changePass	PUT	Заголовок запроса: токен пользователя Тело запроса: Объект с данными для смены пароля пользователя в JSON формате	—	Метод изменения пароля текущего пользователя
user/{id}	DELETE	Заголовок запроса: токен пользователя URL параметр: Id пользователя	—	Метод позволяет удалить пользователя из системы по id пользователя
user/{id}	GET	Заголовок запроса: токен пользователя URL параметр: Id пользователя	DTO объект пользователя	Метод возвращает объект пользователя по полученному id пользователя
user	GET	Заголовок запроса: токен пользователя	Список DTO объектов пользователей	Метод позволяет получить список всех пользователей