МИНОБРНАУКИ РОССИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

Факультет компьютерных наук

Кафедра программирования и информационных систем

Разработка стримингового музыкального веб-приложения «Musicman»

Курсовая работа

09.03.04 Программная инженерия

Информационные системы и технологии

Допущено к защите в ГЭК \_\_.\_\_.2023

Зав. кафедрой *С.Д. Махортов, д. ф.-м. н., профессор*

Обучающийся *А.Р. Сашина, 3 курс, д/о*

Обучающийся *З.С. Казмиров, 3 курс, д/о*

Обучающийся *Е.М. Охрямкин, 3 курс, д/о*

Руководитель *В.С. Тарасов, старший преподаватель*

**Воронеж 2023**

Содержание

[Содержание 2](#_Toc145620495)

[Введение 4](#_Toc145620496)

[1 Постановка задачи 5](#_Toc145620497)

[1.1 Постановка задачи 5](#_Toc145620498)

[1.2 Требования к разрабатываемой системе 5](#_Toc145620499)

[1.2.1 Функциональные требования 5](#_Toc145620500)

[1.2.2 Требования к приложению и программному обеспечению 5](#_Toc145620501)

[2 Анализ предметной области 7](#_Toc145620502)

[2.1 Терминология (глоссарий) 7](#_Toc145620503)

[2.2 Анализ рынка 7](#_Toc145620504)

[2.3 Обзор аналогов 8](#_Toc145620505)

[2.3.1 Spotify 8](#_Toc145620506)

[2.3.2 Deezer 9](#_Toc145620507)

[2.3.3 Яндекс Музыка 11](#_Toc145620508)

[2.3.4 Youtube Music 12](#_Toc145620509)

[3 Реализация 14](#_Toc145620510)

[3.1 Средства реализации 14](#_Toc145620511)

[3.2 Анализ поставленной задачи 15](#_Toc145620512)

[3.2.1 Диаграмма IDEF0 15](#_Toc145620513)

[3.2.2 Диаграмма вариантов использования 15](#_Toc145620514)

[3.2.3 Диаграмма последовательности 17](#_Toc145620515)

[3.2.4 Диаграмма состояний 19](#_Toc145620516)

[3.2.5 Диаграмма классов сущностей 19](#_Toc145620517)

[3.2.6 Диаграмма развертывания 21](#_Toc145620518)

[3.3 Реализация серверной части приложения 21](#_Toc145620519)

[3.3.1 Конфигурации 23](#_Toc145620520)

[3.3.2 Контроллеры 23](#_Toc145620521)

[3.3.3 Мапперы 23](#_Toc145620522)

[3.3.4 Модель 24](#_Toc145620523)

[3.3.5 Репозитории 24](#_Toc145620524)

[3.3.6 Планировщик 25](#_Toc145620525)

[3.3.7 Сервисы 25](#_Toc145620527)

[3.4 Реализация клиентской части приложения 26](#_Toc145620528)

[3.4.1 Описание главной страницы 27](#_Toc145620529)

[3.4.2 Описание страницы с композициями 29](#_Toc145620530)

[3.4.3 Описание страниц профиля пользователя и друзей 29](#_Toc145620531)

[3.4.4 Описание страницы поиска песен 31](#_Toc145620532)

[3.4.5 Описание страницы администратора 32](#_Toc145620533)

[4 Метрики 33](#_Toc145620534)

[5 Тестирование 34](#_Toc145620535)

[Заключение 35](#_Toc145620536)

[Список использованных источников 36](#_Toc145620537)

Введение

В современном мире стриминговые музыкальные сервисы получают все большую популярность. Они предоставляют пользователю доступ к миллионам аудиозаписей в любое время и в любом месте, что позволяет прослушивать музыку в комфортных условиях.

Цель данной курсовой работы – разработка веб-приложения музыкального стримингового сервиса, позволяющего пользователю в удобной форме слушать любимую музыку и делиться песнями с другими пользователями.

В работе будет рассмотрен процесс разработки приложения, включая описание его проектирования функциональности, дизайна и архитектуры системы.

Для реализации проекта будут использованы современные технологии и инструменты, необходимые для создания качественного веб-приложения. Работа будет состоять из нескольких этапов, включая анализ потребностей пользователей, проектирование интерфейса и архитектуры, разработку функциональности и интеграции с внешними сервисами, а также тестирование и оптимизацию приложения для улучшения его производительности и удобства использования.

Результатом данной работы будет функциональное и удобное веб-приложение музыкального стримингового сервиса, удовлетворяющее потребностям пользователей и соответствующее современным требованиям в области веб-разработки.

1. Постановка задачи
   1. Постановка задачи

Основной задачей курсового проекта является создания стримингового сервиса для прослушивания песен с возможностью настройки приложения под нужды пользователя.

Основными целями создания системы являются:

* возможность регулировать настройки прослушивания песен;
* показ песен пользователю на основе его музыкальных предпочтений;
* возможность рекомендации песни другим пользователям.
  1. Требования к разрабатываемой системе
     1. Функциональные требования

Для реализации поставленных целей система должна обладать следующими функциональными возможностями:

* регистрация пользователей;
* загрузка и удаление музыки;
* ведение статистики за пользователем по жанрам прослушиваемой музыки;
* просмотр рекомендуемых песен;
* возможность рекомендации песни пользователям;
* возможность добавления в друзья других пользователей;
* настройка звукового сигнала.
  + 1. Требования к приложению и программному обеспечению

К разрабатываемому приложению выдвигаются следующие требования:

* кроссплатформенность. Приложение должно быть доступно пользователю вне зависимости от используемого им браузера.
* отзывчивость. Приложение должно достаточно быстро предоставлять корректную информацию пользователю.
* удобство использования. Приложение должно иметь простой и понятный интерфейс.
* надежность. Приложение должно стабильно работать и не иметь ошибок, способных привести к потери информации пользователя.
* масштабируемость. Приложение должно быть способно обрабатывать большое количество запросов без значительных потерь в скорости.
* безопасность. Приложение должно обеспечивать безопасность пользовательских данных от взлома и иных хакерских атак.

1. Анализ предметной области
   1. Терминология (глоссарий)

Администратор сайта – специалист, осуществляющий информационную поддержку сайта, управление контентом.

Веб-браузер (браузер) - клиентская программа, поставляемая третьими сторонами и позволяющая просматривать содержимое веб-страниц.

Контент – совокупность информационного наполнения веб-сайта.

Неавторизованный пользователь (гость) – человек, который может авторизоваться в системе, если был зарегистрирован ранее, или пройти регистрацию.

Пользователь – человек, который зарегистрирован в системе и имеет доступ к личному кабинету и основному функционалу системы.

Система администрирования – закрытая от посетителей часть сайта. Управляется администратором.

Эквалайзер – программа, позволяющая регулировать громкость отдельных зон частотного диапазона и выравнивать амплитудно-частотную характеристику звукового сигнала.

* 1. Анализ рынка

Музыкальные стриминговые сервисы являются одними из наиболее популярных услуг, которые с каждым годом набирают все большую аудиторию. С 2017 продажи альбомов на цифровых и физических носителях упали на 17% и 20% соответственно. Несмотря на это, объем продаж музыки все равно вырос [1]. Связано это с ростом стриминговых сервисов – количество прослушиваний в них выросло на 60%.

Аналогичная ситуация в 2016 году наблюдалась и в России — наиболее активный прирост дало потоковое вещание. Более миллиона платящих пользователей Apple Music, «Яндекс.Музыки» и других сервисов помогли удвоить доходы этих компаний в данном сегменте [2]. Теперь почти 40% индустрии приходится на стриминг.

Еще 4 года назад лишь четверть россиян готова была заплатить за контент в интернете. Через год эта цифра увеличилась вдвое.

Однако в 2022 году российский рынок стриминга показал спад [3]. Сокращение составило до 5-7 млрд рублей. В 2021 аналитики оценивали объем рынка в 9,5 млрд рублей.

При этом российские музыкальные сервисы заявляют о росте аудитории. Например, в «Яндексе» рассказали, что количество подписчиков «Плюса» за год увеличилось на 53%.

Таким образом, можно сделать вывод о том, что рынок стриминговых сервисов в России продолжает развиваться. Сейчас доля пользователей, готовых к покупке контента, достигла свыше 70%. Поэтому существует огромный потенциал развития, возможность завоевать долю этого развивающегося рынка, имея инновационный продукт и правильную стратегию развития проекта.

* 1. Обзор аналогов

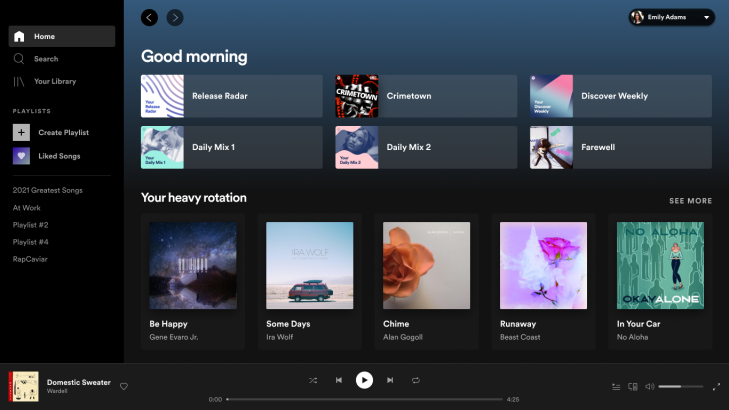
Перед разработкой приложения был проведен обзор существующих аналогов на рынке.

* + 1. Spotify

Spotify – это один из самых популярных музыкальных стриминговых сервисов в мире. Этот сервис имеет огромную библиотеку с миллионами песен и плейлистов, которые можно слушать как онлайн, так и офлайн. Spotify имеет простой и интуитивно понятный интерфейс, а также множество функций, таких как персонализированные рекомендации, функция обнаружения новой музыки, плейлисты, созданные другими пользователями, и многое другое.

Плюсы:

* большая библиотека с миллионами песен и плейлистов;
* простой и удобный интерфейс, показанный на рисунке 1;
* персонализированные рекомендации и функция обнаружения новой музыки;
* доступно на большинстве устройств, включая смартфоны, компьютеры, телевизоры и т.д.;
* функция офлайн-воспроизведения песен.



1. Главная страница Spotify

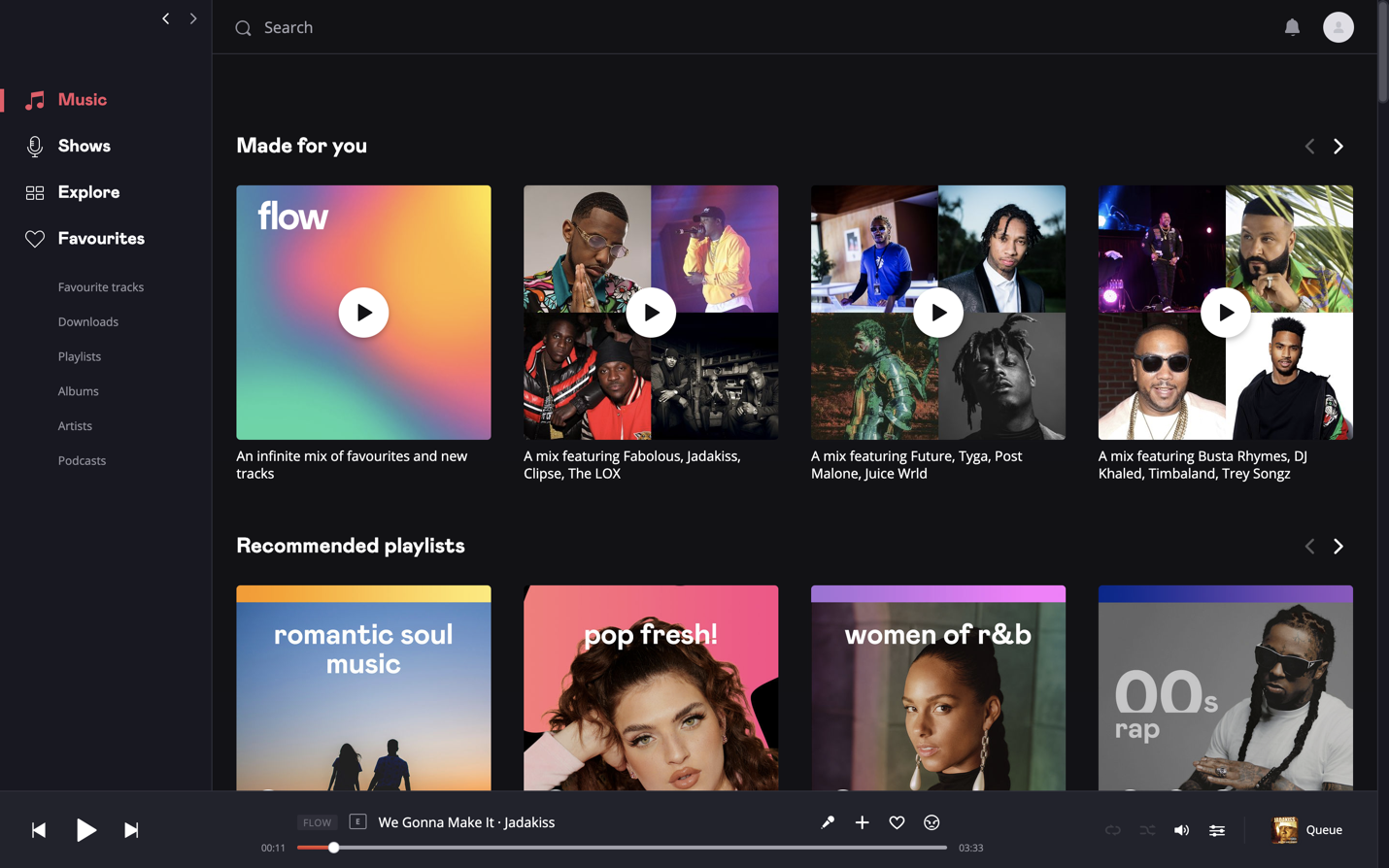
Минусы:

* реклама в бесплатной версии;
* ограниченный доступ к новым альбомам для бесплатной версии;
* некоторые пользователи могут считать плату за премиум-версию слишком высокой.
  + 1. Deezer

Deezer – это музыкальный стриминговый сервис, который доступен в более чем 180 странах. Этот сервис также имеет огромную библиотеку с миллионами песен и плейлистов, а также множество функций, таких как подборки песен на основе настроения, автоматическая настройка на любимые исполнители, подкасты и многое другое.

Плюсы:

* огромная библиотека с миллионами песен и плейлистов;
* множество функций, таких как подборки песен на основе настроения, автоматическая настройка на любимые исполнители, подкасты и многое другое;
* доступно на большинстве устройств, включая смартфоны, компьютеры, телевизоры;
* имеет удобный интерфейс, показанный на рисунке 2;
* качество звука выше, чем у конкурентов;
* функция офлайн-воспроизведения песен.



1. Главная страница Deezer

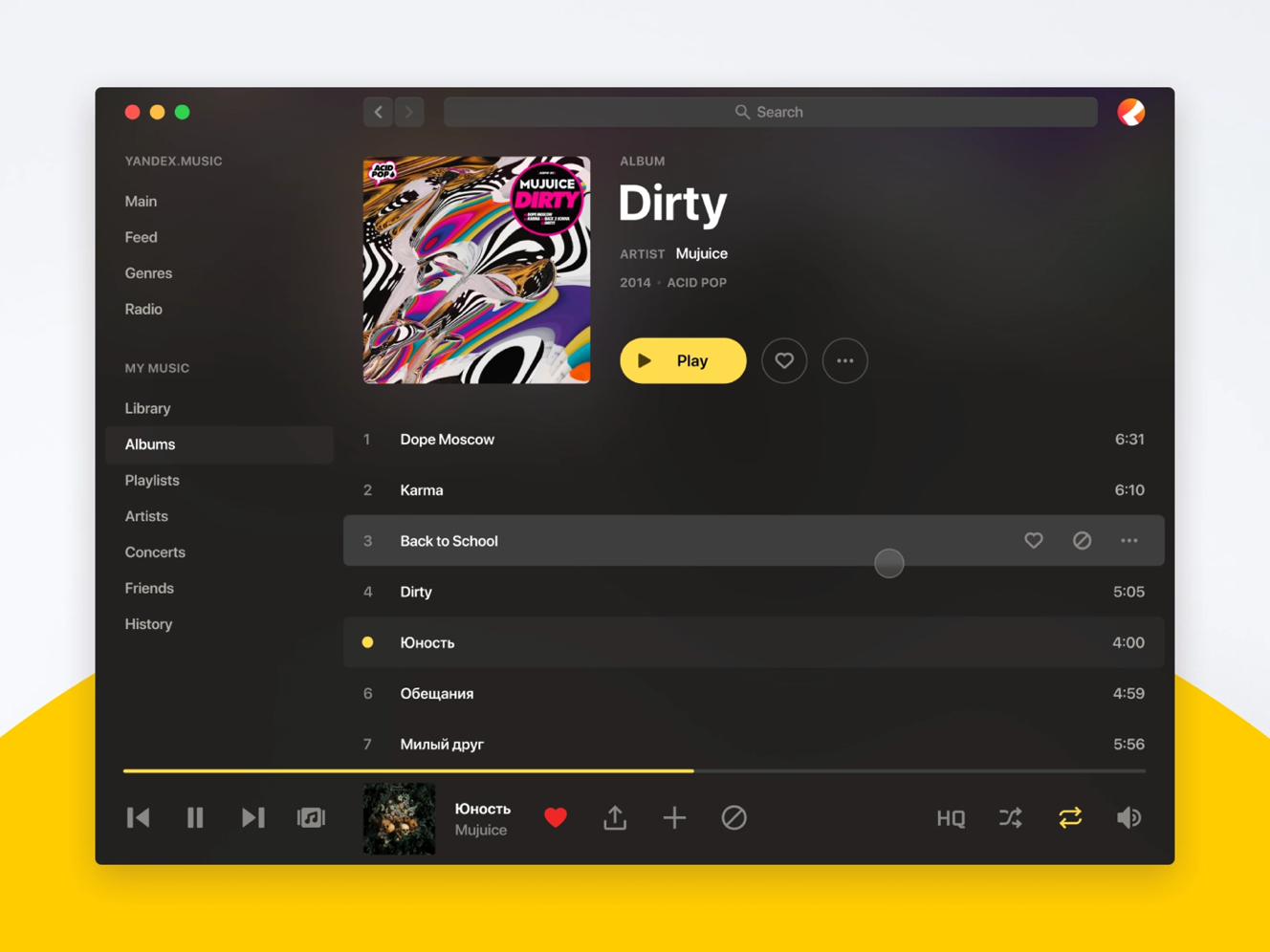
Минусы:

* некоторые пользователи могут считать плату за премиум-версию слишком высокой;
* не все песни доступны в некоторых странах;
* реклама в бесплатной версии.
  + 1. Яндекс Музыка

Яндекс Музыка – это музыкальный стриминговый сервис, разработанный Яндексом. Сервис имеет огромную библиотеку с миллионами песен и плейлистов, а также множество функций, таких как рекомендации на основе настроения, адаптивный плейлист, функция персонализации и многое другое.

Плюсы:

* большая библиотека с миллионами песен и плейлистов;
* множество функций, таких как рекомендации на основе настроения, адаптивный плейлист, функция персонализации и многое другое;
* доступно на большинстве устройств;
* обладает удобным и стильным интерфейсом, показанным на рисунке 3;



1. Главная страница Яндекс Музыки

* бесплатный доступ к музыке для пользователей Яндекс.Плюс;
* функция офлайн-воспроизведения песен.

Минусы:

* некоторые пользователи могут считать интерфейс сложным и запутанным;
* ограниченный доступ к новым альбомам для бесплатной версии;
* реклама в бесплатной версии;
* некоторые пользователи могут испытывать проблемы с качеством звука.
  + 1. Youtube Music

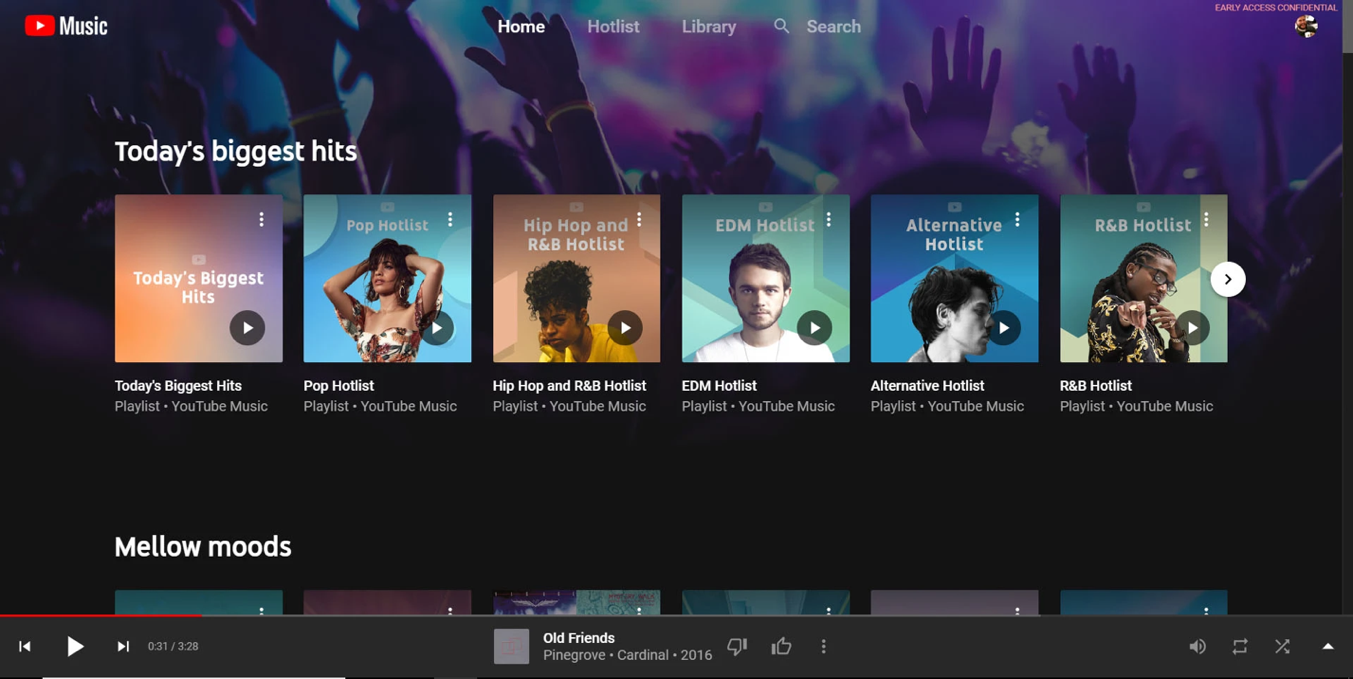
Youtube Music – это музыкальный стриминговый сервис, разработанный Google. Сервис имеет библиотеку с миллионами песен и плейлистов, а также множество функций, таких как персонализированные рекомендации, офлайн-воспроизведение и многое другое.

Плюсы:

* большая библиотека с миллионами песен и плейлистов;
* множество функций, таких как персонализированные рекомендации, офлайн-воспроизведение и многое другое;
* доступно на большинстве устройств;
* обладает удобным интерфейсом, изображенном на рисунке 4;
* бесплатный доступ к музыке для пользователей Youtube Premium;
* интеграция с другими сервисами Google, такими как Google Assistant и Google Home.

Минусы:

* некоторые пользователи могут считать интерфейс сложным и запутанным;
* высокая стоимость премиум-подписки;
* ограниченный доступ к новым альбомам для бесплатной версии;
* реклама в бесплатной версии.



1. Главная страница Youtube Music
2. Реализация
   1. Средства реализации

Система должна состоять из сервера приложения, реляционной базы данных, клиентской части.

Основной используемый стек технологий:

Back-end (серверная часть):

* Java 17;
* Spring Framework;
* PostgreSQL, Liquibase.

Java [4] является кроссплатформенным языком, т.е. для запуска достаточно иметь Java Virtual Machine. Самым популярным фреймворком для Java является Spring [5]. Основным преимуществом является наличие множества готовых компонент и внутренних библиотек, что позволяет быстро и качественно разрабатывать серверную часть приложения.

В качестве базы данных была выбрана широко распространенная PostgreSQL [7], умеющая работать с различными типами данных и позволяющая ускорять запросы с помощью индексов. Для управления изменениями в базу данных будет использоваться библиотека Liquibase [8]. Основным преимуществом является возможность написания миграционных файлов в виде yaml или xml файлов.

Front-end (клиентская часть):

* CSS3, HTML5;
* React.js, TypeScript;
* Effector;
* Material UI.

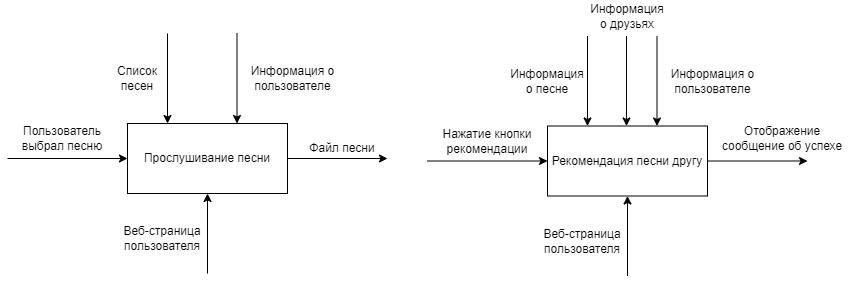
CSS3 и HTML5 были выбраны потому, что это самый современный стандарт вёрстки и разметки. Он поддерживается большинством браузеров и предоставляет множество новых свойств, упрощающих разработку.

React [6] является популярным фреймворком для разработки клиентской части. Главным его преимуществом является виртуальная объектная модель документа (DOM), занимающая мало места. А значит позволяет быстрее обновлять страницу с изменениями и повышает производительность приложения.

Язык TypeScript может быть компилирован в JavaScript код, но в отличие от последнего нивелирует недостатки слабой типизации и берёт множество проверок безопасности кода на себя во время трансляции в JavaScript код.

Effector [9] является стейт-менеджером, который хорошо взаимодействует с React-ом и позволяет легко разделять работу с данными по разным хранилищам (декомпозиция). Material UI [10] является библиотекой готовых компонент для React, которые обладают приятным дизайном.

* 1. Анализ поставленной задачи
     1. Диаграмма IDEF0



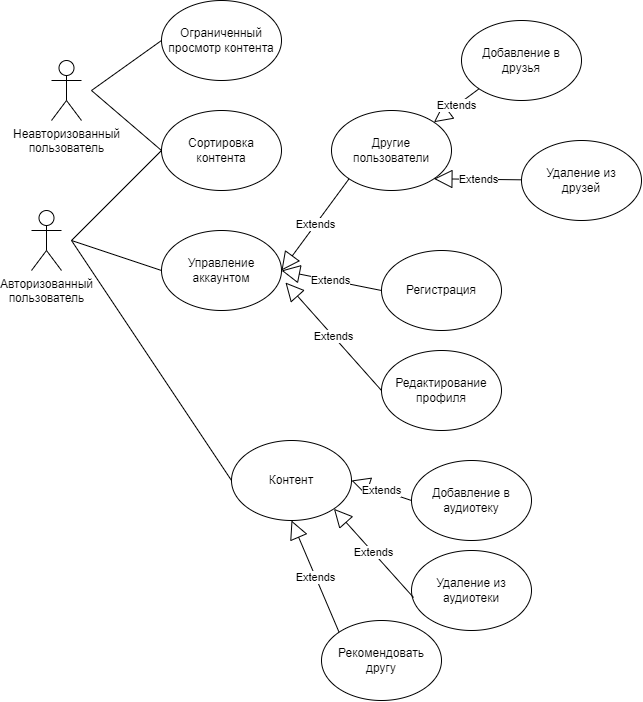
1. Диаграмма IDEF0

На рисунке 5 представлена IDEF0 диаграмма для описания двух бизнес-процессов: прослушивание песни и рекомендация песни другу.

* + 1. Диаграмма вариантов использования

Ниже приведено подробное описание диаграммы использования, представленной на рисунках 6-7. Система используется следующими группами пользователей:

* неавторизованный пользователь;
* авторизованный пользователь;
* администратор.



1. Диаграмма вариантов использования (часть 1)

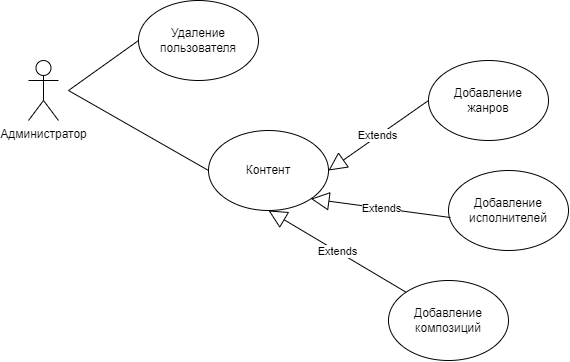
Неавторизованный пользователь может только ограниченно просматривать (без возможности прослушать композиции) и сортировать контент.

Авторизованный пользователь может:

* редактировать свой профиль;
* добавлять в друзья или удалять пользователей из друзей;
* добавлять песни в личную аудиотеку или удалять оттуда;
* рекомендовать другу композиции.

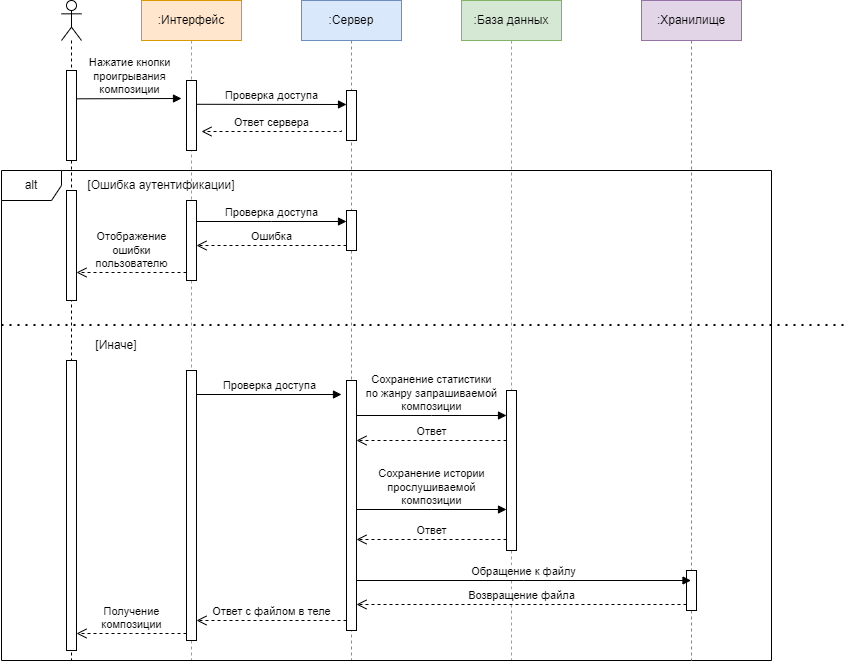
Администратор может:

* удалять пользователей из системы в случае нарушения соблюдений правил платформы;
* управлять контентом: добавлять жанры, исполнителей и композиций.



1. Диаграмма вариантов использования (часть 2)
   * 1. Диаграмма последовательности

Одна из основных целей приложения является показ пользователю композиций на основе его музыкальных предпочтений. Это достигается не только за счет возможности фильтрации контента по жанрам и исполнителю, но и с помощью учета статистики прослушивания пользователя. Далее рассматривается диаграмма последовательности действий при нажатии кнопки прослушивания композиции, изображенная на рисунке 8.

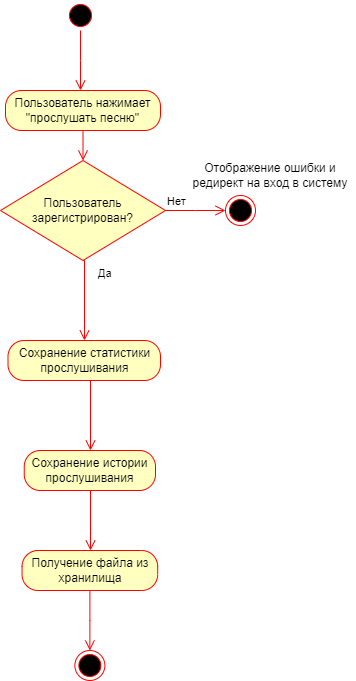


1. Диаграмма последовательности при нажатии прослушивании композиции

При нажатии кнопки прослушивания композиции с вебсайта посылается запрос на сервер, где происходит обработка. В случае если запрос был некорректен (например, неавторизованный пользователь отправил запрос или была запрошена несуществующая композиция), пользователю демонстрируется экран с ошибкой. В противном случае перед получением файла композиции из хранилища в базу данных записывается информация о жанре композиции и уникальный идентификатор пользователя, а также в отношение «история» заносится информация о песне.

* + 1. Диаграмма состояний

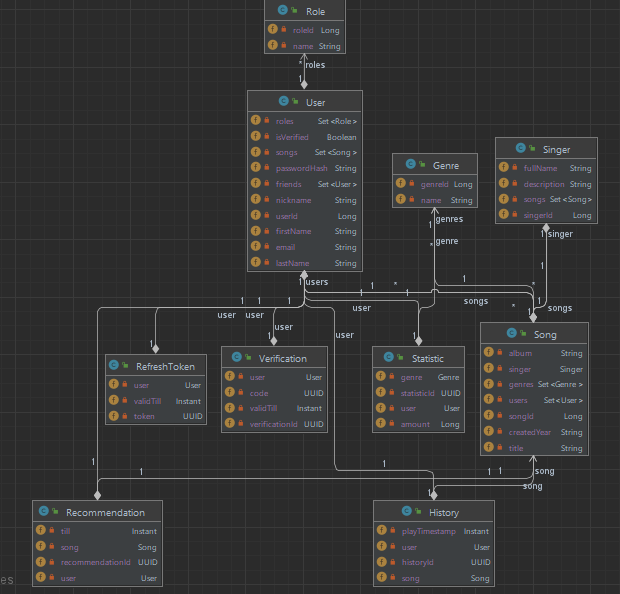
Ниже на рисунке 9 представлена диаграмма состояний для получения песни для прослушивания пользователем.



1. Диаграмма состояний
   * 1. Диаграмма классов сущностей

Для работы сервера используются основные сущности, изображенные на диаграмме классов на рисунке 10. Далее следует описание классов:

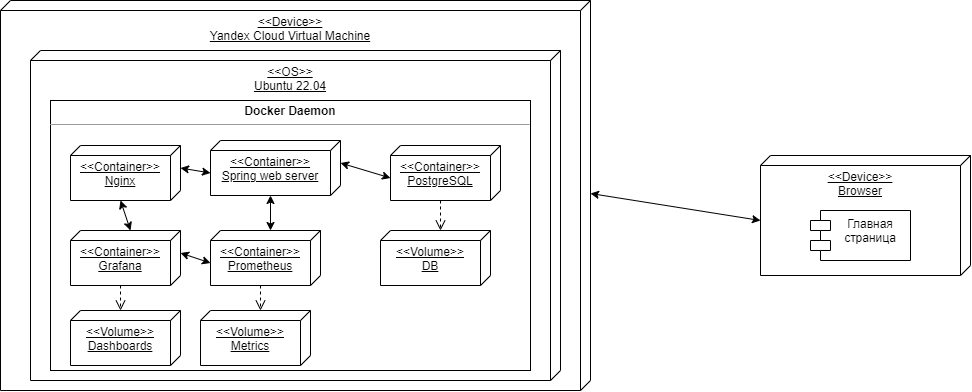
* User – класс пользователя;
* Genre – класс жанра;
* Singer – класс исполнителя;
* RefreshToken – класс токена для обновления access токена, нужный для отправки авторизованных запросов;
* Verification – класс для хранения информации о верификации пользователя;



1. Диаграмма классов сущностей

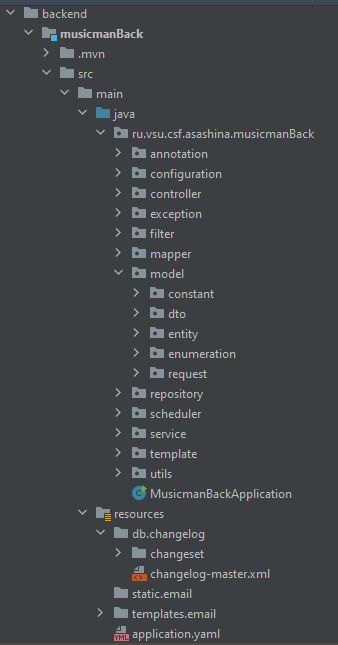
* Statistic – класс статистики пользователя по прослушиваемым жанрам;
* Song – класс композиции;
* History – класс истории прослушанных композиций;
* Recommendation – класс для хранении информации о рекомендованной песни другим пользователем.
  + 1. Диаграмма развертывания

На рисунке 11 изображена диаграмма развертывания приложения. Основными компонентами приложения являются сервер, который использует базу данных и компоненты для слежения за метриками бэкенд части, и клиентская часть с веб-браузером. Для работы серверной и клиентской частей необходим выход в интернет.



1. Диаграмма развертывания
   1. Реализация серверной части приложения

Серверная часть приложения представляет собой монолитную архитектуру. В рамках данного проекта не имело смысла выделять части логики в отдельные микросервисы, так как при неисправности основного сервиса параллельные задачи (например, очистка неактуальных рекомендаций) не должны работать. Организацию кода можно увидеть на рисунке 12. Каждый класс помещен в соответствующую его предназначению папку, структурируя код. Папки с классами организованы в соответствии архитектуре MVC (Model-View-Controller). Данная архитектура была выбрана из-за возможности ограничения ответственности между моделью, представлением и контроллером.



1. Структура кода серверной части приложения

Для соблюдения принципов SOLID и удобочитаемости кода [11] разработка классов следовала цепочке model (сущности базы данных и приложения) – repository (класс, отвечающий за взаимодействие с базой данных) – service (класс с бизнес-логикой) – controller (класс, к которому происходит обращение по https запросу извне).

Далее будет подробно рассмотрено содержимое наиболее важных пакетов проекта.

* + 1. Конфигурации

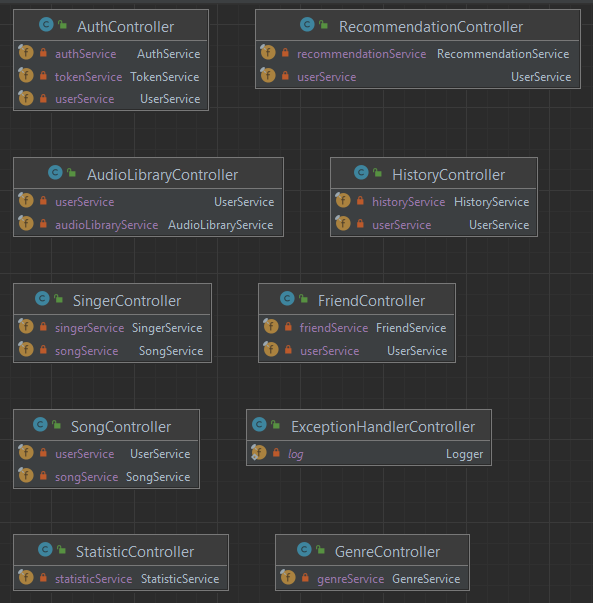
В конфигурациях хранится информация о создании классов из библиотек для обеспечения работы приложения. Одной из основных конфигурацией является SecurityConfiguration, которая отвечает за определение доступа по переданному в запросе access токену. Также реализована конфигурация для генерирования Swagger документации.

* + 1. Контроллеры

Всего реализовано 10 контроллеров, изображенных на рисунке 13. Внутри себя контроллеры содержат только ссылки на сервисы, с которыми и взаимодействуют.

* + 1. Мапперы

В серверной части были реализованы специальные классы мапперы, которые преобразуют класс сущности в класс представления, и наоборот. Классы представления являются DTO (data transfer object). Использование DTO необходимо для быстрой работы приложения, чтобы в качестве ответов возвращать не всю сущность с ее дочерними элементами, а лишь некоторую часть информации об объекте. Также мапперы помогают содержать код в чистоте, иначе сервисы бизнес-логики были бы перегружены set() методами.



1. Контроллеры
   * 1. Модель

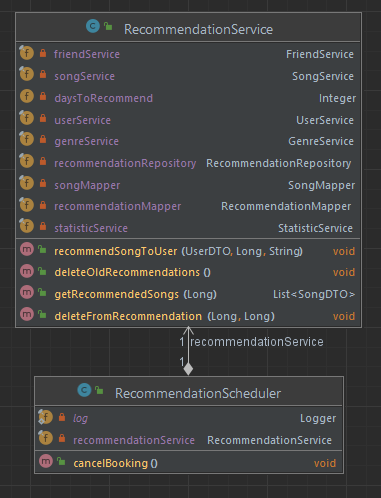
Данная папка отвечает за хранение моделей, которые включают в себя DTO, сущности и объекты запросов. Последние являются телами для запросов с методами POST и PUT. Сущности представляют собой отношения из базы данных [14].

* + 1. Репозитории

Представляют собой классы посредников между базой данных и бизнес-логикой. Все репозитории наследуются от JpaRepository, в котором уже реализованы простейшие операции, например, получение сущности по ее первичному ключу. Каждый репозиторий соответствует своей сущности.

* + 1. Планировщик

Планировщик удаления рекомендаций на рисунке 14 представляет собой класс, который асинхронно раз в неделю очищает отношение «рекомендации» в базе данных от неактуальных записей. Использует сервис рекомендаций для обращения к методу удаления неактуальных композиций.

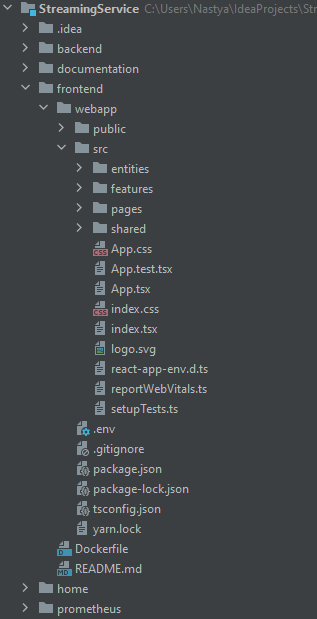


1. Планировщик удаления рекомендаций
   * 1. Сервисы

Сервисы являются классами по осуществлению бизнес-логики. Всего реализовано 14 сервисов, которые работают со своими компонентами. Например, сервис SongService отвечает за просмотр, создания и удаления песен. Этот класс в качестве полей содержит ссылку на репозиторий, работающий с сущностью песни. Если в каком-либо сервисе нужен доступ к логике с композициями, то представляется ссылка не на репозиторий, а на сервис песен. Таким образом, усилен контроль за вызовом методов модификации тех или иных сущностей в разных частях логики.

* 1. Реализация клиентской части приложения

Архитектурным решением для клиентской части приложения, является feature-sliced design. Проект клиентского приложения был разделен на следующие каталоги, показанные на рисунке 15:



1. Структура кода клиентской части приложения

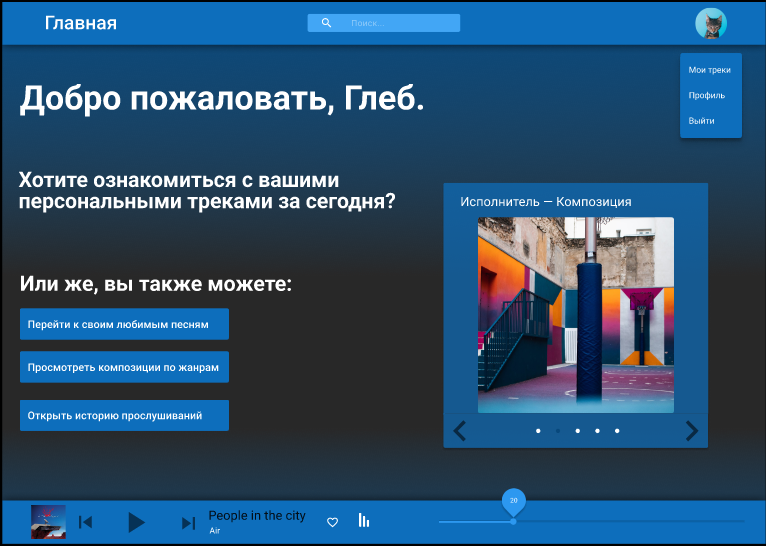
* Entities – компоненты, хранящиеся в данном каталоге, являются сущностями без глобального функционала, не взаимодействуют с состоянием приложения.
* Features – компоненты, которые собираются из entity и работают с состоянием приложения.
* Pages – «страницы» приложения. Эти компоненты собираются из feature и имеют доступ к состоянию приложения.
* Shared – компоненты, которые являются побочными, но при этом используются во всем приложении. В данном каталоге, находятся утилиты, функции для взаимодействия с API.

Такая структура позволяет разделить функциональность на маленькие и независимые куски кода, обеспечивая повторное использование компонентов и легкость сопровождения.

* + 1. Описание главной страницы

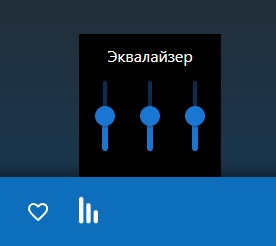
На рисунке 16 изображена главная страница приложения. С нее пользователь может перейти к своим песням, просмотреть композиции по жанрам, а также открыть историю прослушивания. Если пользователь уже прослушивал какую-либо композицию, то она будет проигрываться в нижней панели приложения. При нажатии на иконку профиля пользователь может просмотреть свои треки, свой профиль, а также выйти из системы.

Нижняя и верхние панели на всех страницах приложения (кроме входа и регистрации) одинаковы.



1. Главная страница

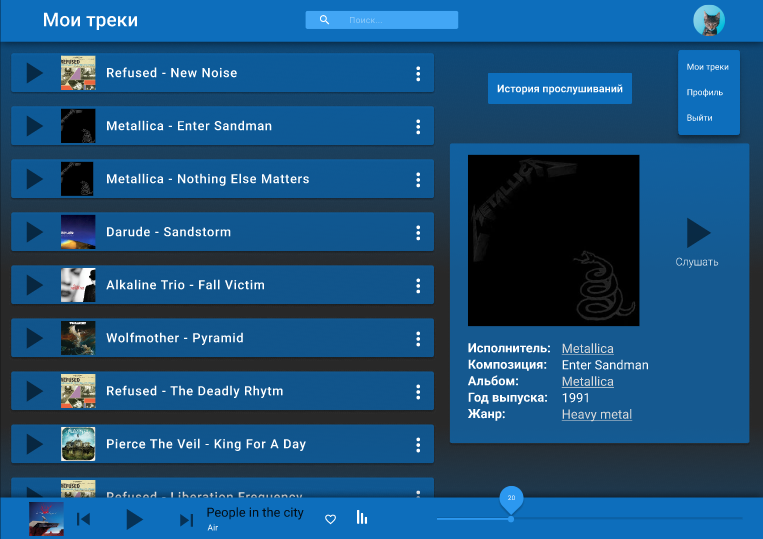
Эквалайзер находится в нижней панели страницы и изображен на рисунке 17. Левая колонка отвечает за низкие частоты, средняя за средние и правая за высокие.



1. Эквалайзер

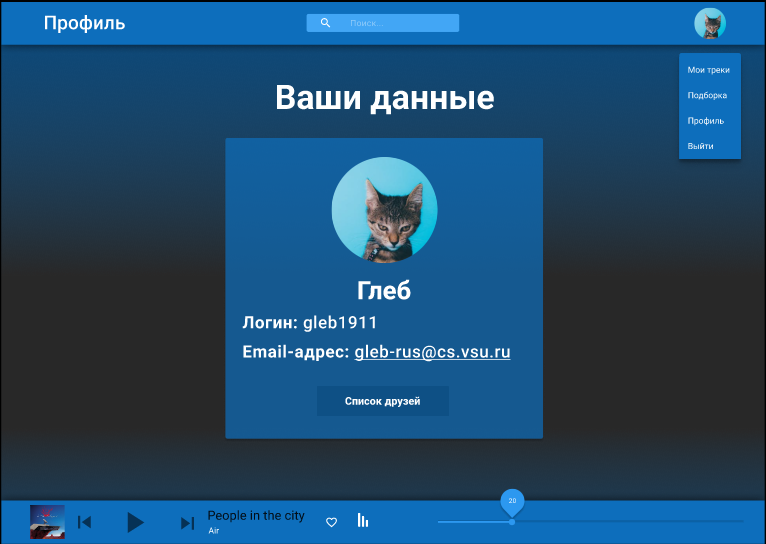
* + 1. Описание страницы с композициями

Страница с треками пользователя, изображенная на рисунке 18, показывает список добавленных песен. При нажатии на конкретную песню по ней выводится информация, включающая в себя название, исполнителя песни, альбом, год выпуска и жанры. Также в нижней панели отображается песня с возможностью настройки звука с помощью эквалайзера. При нажатии на троеточие в правом углу композиции выпадает меню, где можно удалить песню из личной аудиотеки или порекомендовать ее другу.



1. Страница с треками пользователя
   * 1. Описание страниц профиля пользователя и друзей

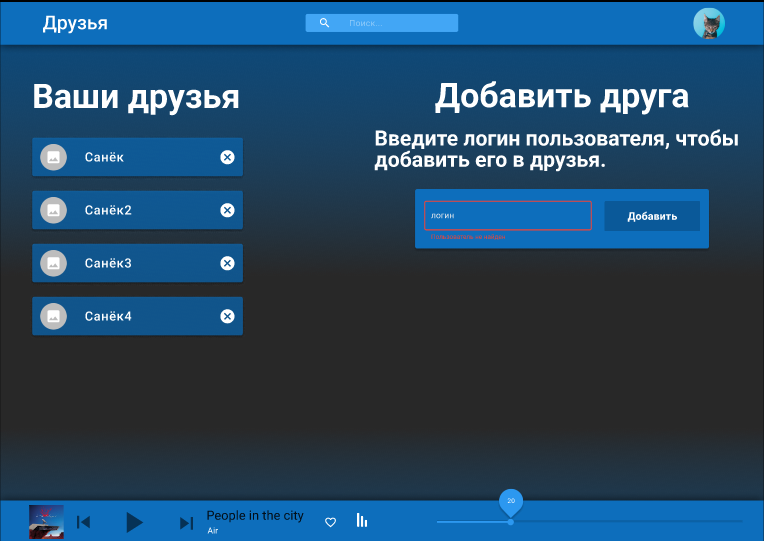
На рисунке 19 изображен профиль пользователя. На данной странице кратко представлена информация о текущем пользователе системы. Кнопка «список друзей» ведет на страницу с друзьями, изображенной на рисунке 20.



1. Профиль пользователя

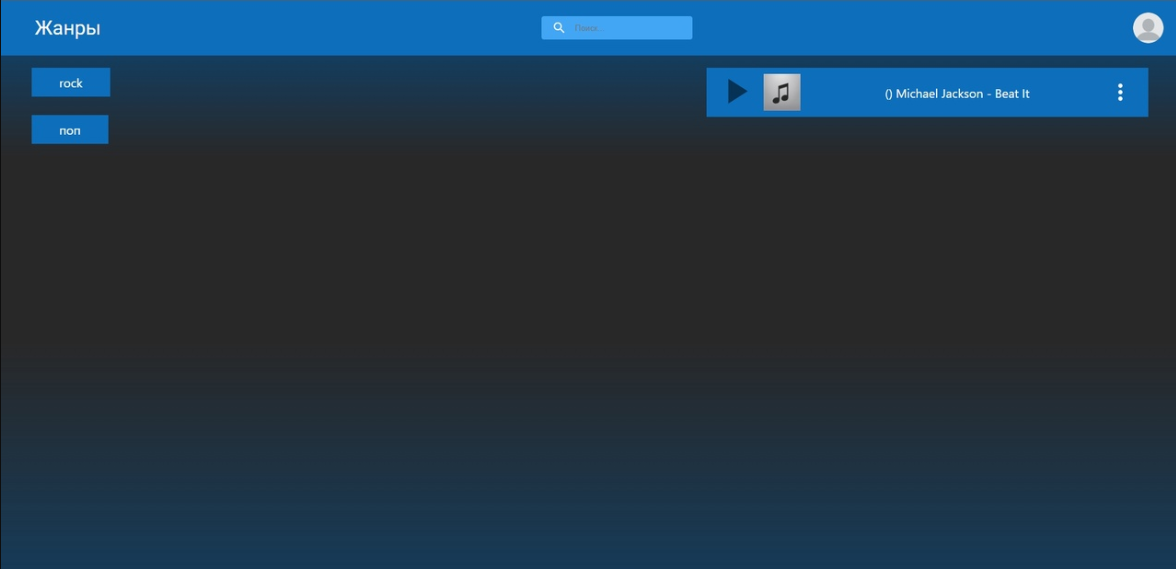
На странице с друзьями пользователь может их добавлять или удалять. Добавление друга происходит через поиск пользователя по его никнейму. Список уже добавленных друзей изображен в левой части страницы.

При наличии друзей пользователь может рекомендовать им песни, находя их по никнейму. Рекомендованные композиции отображаются на похожей по дизайну со страницей треков пользователя. Пользователь не может рекомендовать песню другому пользователю, если тот не был добавлен в друзья.



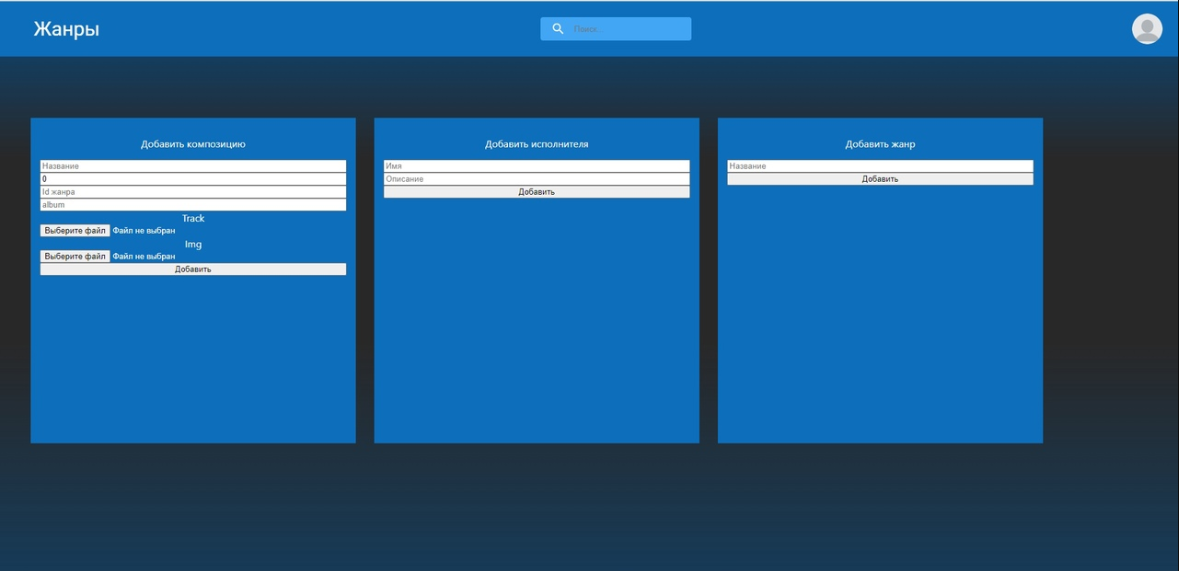
1. Страница с друзьями и их менеджментом
   * 1. Описание страницы поиска песен

На странице 21 изображена страница с результатом поиска песен. Справа отображаются все доступные жанры, а слева результат поиска. Искать композиции можно по названию или по исполнителю. Для поиска по жанрам пользователь нажимает на соответствующие кнопки справа, а слева динамически меняется список песен.



1. Страница с поиском песен по жанрам и названию
   * 1. Описание страницы администратора

На рисунке 22 изображена страница администратора. На ней администратор может добавлять композиции, исполнителей и жанры. Переход к этой странице осуществляется через выпадающее меню из профиля, где есть ссылка на панель.



1. Панель администратора
2. Метрики

Помимо ведения аналитики за пользователем по количеству прослушанных песен в том или ином жанре, был реализован доступ к просмотру панелей с метриками Java Virtual Machine, показанной на рисунке 23. Данные метрики были получены с помощью системы мониторинга Prometheus [12] и изображены с помощью интерфейса Grafana [13].



1. Метрики Java Virtual Machine

Одной из целью заведения метрик являлась возможность отслеживания расхода памяти и скорости выполнения запросов. Данную информацию необходимо учитывать для дальнейшего улучшения системы и ее логики, особенно в тех местах, где происходит выгрузка большого объема данных (например, выгрузка песен в личной аудиотеке пользователя).

1. Тестирование

Тестирование было проведено в ручном формате. Для всех функциональных возможностей приложения были написаны тест-кейсы. С помощью Swagger документации и запросов через Postman были протестированы все важные сценарии. При обнаружении багов создавалась отдельная ветка разработки. Если баг был критичен, то изменения сразу же попадали в master (основную) ветку как hotfix.

На серверной части приложения был реализован тест с запуском контекста фреймворка Spring. Данный тест может стать основой CI проекта (continuous integration) и помогает удостовериться, что изменения на бэкенде не повлияли на работоспособность сервера.

Заключение

В ходе выполненной работы было реализовано веб-приложение Musicman для прослушивания музыки. Оно обладает визуальным интерактивным интерфейсом, который необходим для взаимодействия пользователя с системой и позволяет выводить на экран всю нужную информацию. Были выполнены следующие задачи:

* разработана серверная часть приложения, развернутая на виртуальном машине;
* разработана клиентская часть, необходимая для взаимодействия с логикой приложения;
* с помощью API была реализована связь между клиентской и серверной частями;
* приложение учитывает статистику по композициям для рекомендации, осуществляет логику по возможности рекомендации песни другу и использует эквалайзер.

Список использованных источников

1. Миннебаева, Р.А. Анализ рынка стриминговых сервисов / Р.А. Миннебаева, Р.А. Миннебаев // Форум молодых ученых. – Саратов, 2018. - №5(21). – С. 666-670.
2. Доходы от музыкального стриминга в России удвоилось в 2016 году [Электронный ресурс] : статья в журнале «Ведомости». – Режим доступа: https://www.vedomosti.ru/technology/articles/2017/05/03/688474-dohodi-striminga
3. Российский рынок музыкального стриминга в 2022 году сократился до 5-7 млрд рублей – исследование [Электронный ресурс] : статья на сайте новостей про информационные технологии «vc.ru». – Режим доступа: https://vc.ru/services/600606-rossiyskiy-rynok-muzykalnogo-striminga-v-2022-godu-sokratilsya-na-30-50-do-5-7-mlrd-rubley-issledovanie
4. Блох Д. Java: Эффективное программирование / Д. Блох – М. : Вильямс, 2018 – 464с.
5. Spring [Электронный ресурс] : официальный сайт фреймворка Spring. – Режим доступа: <https://spring.io/>
6. React.js [Электронный ресурс] : документация фреймворка React. – Режим доступа: <https://react.dev/>
7. PostgreSQL [Электронный ресурс] : официальный сайт. – Режим доступа: <https://www.postgresql.org/>
8. Liquibase [Электронный ресурс] : официальный сайт. – Режим доступа: <https://www.liquibase.org/>
9. Effector [Электронный ресурс] : документация. – Режим доступа: <https://effector.dev/>
10. Material UI [Электронный ресурс] : официальный сайт. – Режим доступа: <https://mui.com/>
11. Мартин Р. Чистый код: создание, анализ и рефакторинг / Р. Мартин – СПб. : Питер, 2022 – 464с.
12. Prometheus [Электронный ресурс] : официальная документация. – Режим доступа: <https://prometheus.io/docs/introduction/overview/>
13. Grafana [Электронный ресурс] : официальный сайт. – Режим доступа: <https://grafana.com/>
14. Исходный код и документация стримингового сервиса «Musicman» [Электронный ресурс] : репозиторий на веб-хостинге «GitHub». – Режим доступа: https://github.com/TP-6-1-3/StreamingService