МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ» (ФГБОУ ВО «ВГУ»)

Факультет компьютерных наук
Кафедра программирования и информационных технологий

Отчет по курсу «Технологии программирования»

Каталог музыкальных произведений, записанных в электронной форме. Рекомендации для конкретного пользователя на основе его вкусов.

Заказчики: Тарасов В. С., Клейменов И. В.
Исполнители: Яцкова В. Н., Сапелкин А. Г., Тарлыков А. В.

Воронеж, 2022 г.

Содержание

Введение

С развитием компьютерных технологий оцифровке подвергаются многие события и процессы. Музыканты набирают популярность с помощью распространения своих произведений в сети интернет посредством рекламы, а уже известные исполнители получают прибыль продавая авторские права различным сервисам для прослушивания музыкальных треков.

В современном мире большинство музыкантов используют для написания своих треков различные приложения, в которых собраны звуки всевозможных музыкальных инструментов. Это значительно упрощает процесс сочинения, ведь совсем не обязательно иметь под рукой массивные барабанные установки или уметь играть на скрипке, достаточно просто установить подходящее приложение и разобраться с сочетаниями клавиш на клавиатуре.

Советский музыковед и музыкальный социолог Арнольд Наумович Сохор однажды сказал, что музыка, которая, кстати, с греческого переводится как «муза», «отражает действительность и воздействует на человека посредством осмысленных и особым образом организованных по высоте и во времени звуковых последований, состоящих в основном из тонов».

В качестве распорядителей информации в настоящее время все чаще выступают базы данных. Они обеспечивают безопасное хранение структурированных данных и доступ к ним почти в любое время. Любой современный проект, предусматривающий большое количество пользователей, нуждается в базе данных, которая отвечает требованиям проекта, а также удовлетворяет потребности по хранению, управлению и администрированию данных.

С развитием технологий люди получили возможность наслаждаться любимой музыкой в любое удобное время с помощью различных сервисов. Современные системы позволяют не только прослушивать уже известные композиции, но и находить новые треки и новых исполнителей посредством рекомендаций, основанных на плейлисте людей со схожими музыкальными предпочтениями.

Создание единого музыкального пространства позволяет пользователям находить новую для себя музыку и прослушивать уже известные композиции. А для авторов это хороший инструмент визуализации своей музыкальной востребованности, так как, например, по количеству прослушиваний легко можно определить, насколько популярно произведение.

Для сбора подбора новых треков, авторов, жанров, отслеживания предпочтений пользователей необходима база данных, которая будет содержать эту информацию.

Целью данной работы являлась разработка приложения, предназначенного для подбора и рекомендации новых произведений на основе предпочтений пользователя, который он также может задать. Также данное приложение может использоваться для самостоятельного поиска треков в случае необходимости. Это позволит пользователям расширять свой музыкальный кругозор и открывать для себя новых исполнителей, новые жанры и новые композиции.

1. Анализ предметной области

1.1. Обзор аналогов

Начиная разработку нового музыкального приложения, стоит ознакомится с уже существующими и рассмотреть их достоинства и недостатки. На основании полученной информации можно сделать выводы о том, какие недочеты стоит взять на вооружение и не допускать в своей работе, чтобы интерфейс для пользователя был удобен, прост в использовании и управлении.

1.1.1. Spotify

Возможности приложения:

— Прослушивание любых выбранных и доступных произведений;

— Получение личных рекомендаций на основе избранных или уже прослушанных произведений;

— Доступ к скачанным композициям в режиме offline;

— Доступ к различным чартам и хит-парадам;

— Доступ к радиостанциям;

Рассмотрим интерфейс приложения.

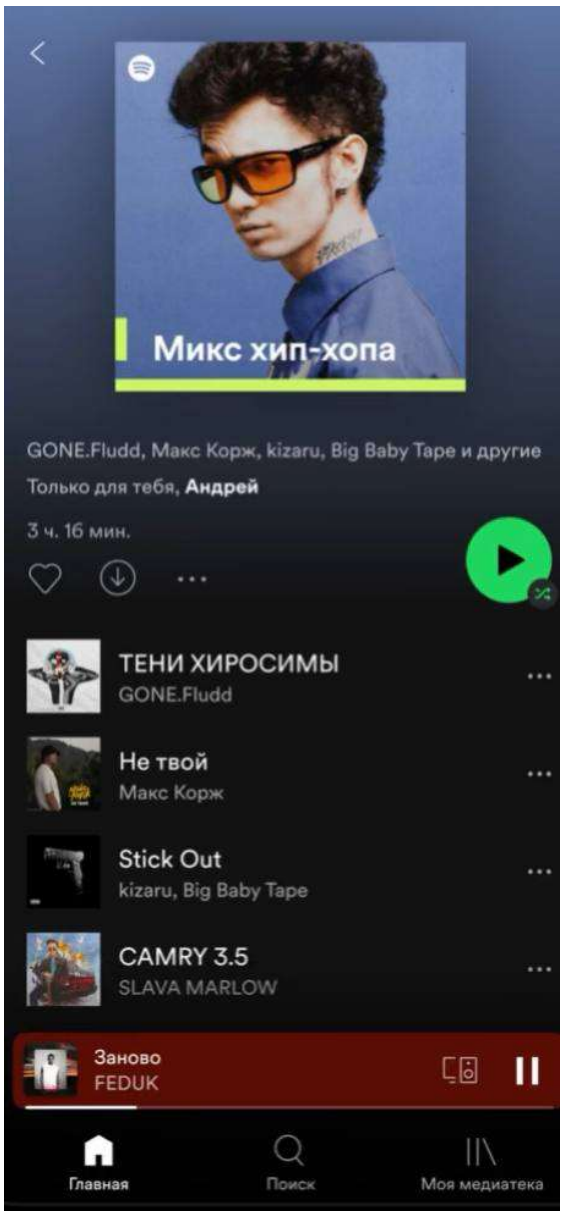

Рисунок 1 — Пример интерфейса Spotify.

Интерфейс минималистичный и понятный, экран не перегружен лишними элементами, все нужные функции в быстром доступе.

Достоинства:

— Простой интерфейс;

— Доступ с различных платформ;

— Удобно в использовании;

— Легко делиться музыкой с друзьями;

— Большая база музыкальных произведений.

Недостатки:

— Обязательная регистрация для доступа к функционалу приложения;

— Ограниченные возможности бесплатной версии;

— Отсутствуют тексты песен;

— Часто всплывающая реклама.

На основе вышеописанного можно сделать следующий вывод:

Удобное и интуитивно понятное приложение с широкими возможностями, которое, однако, требует регистрации для доступа к функционалу, а в бесплатной версии ухудшает впечатления пользователя при использовании приложения.

1.1.2. VK Music

Возможности приложения:

— Прослушивание любых доступных композиций;

— Просмотр и воспроизведение музыки, которая нравится пользователям в списке друзей;

— Просмотр текста некоторых композиций;

— Доступ к скачанным трекам в режиме offline;

— Доступ к редактированию текста композиции при выявлении ошибок;

— Рекомендации на основе сохраненных треков;

— Доступ к различным подборкам, чартам, плейлистам, сборникам;

Рассмотрим интерфейс приложения.

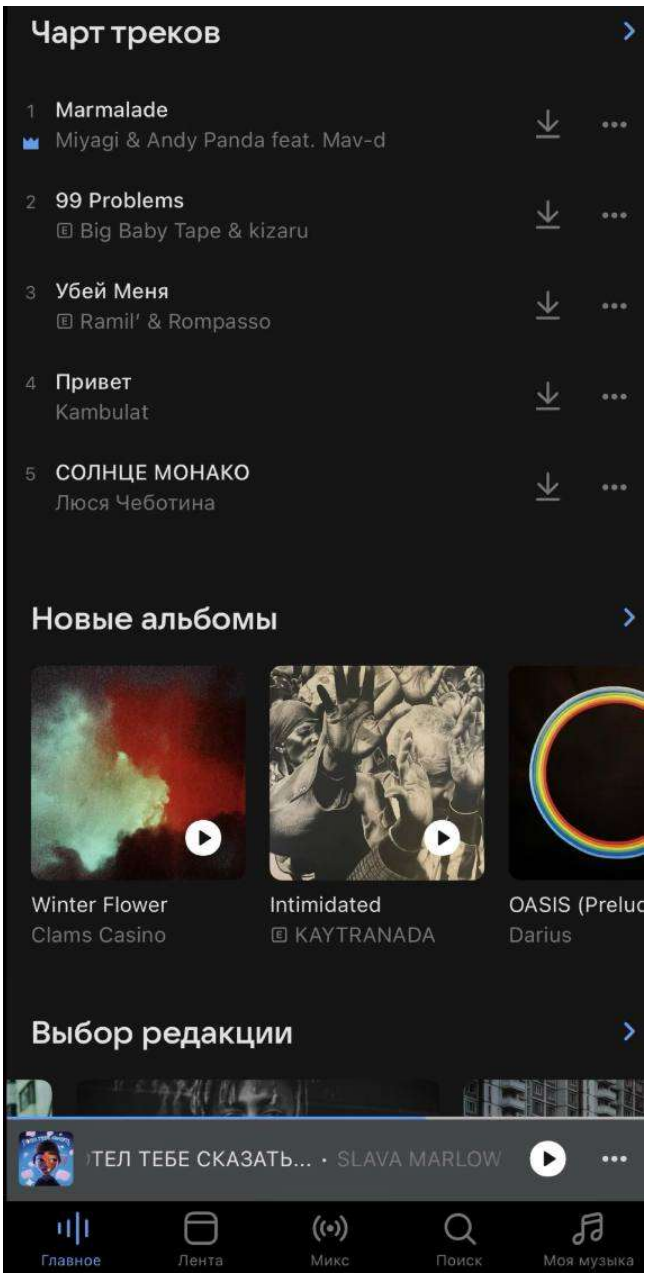

Рисунок 2 — Пример интерфейса VK Music

Достаточно понятный интерфейс, с выделенными на странице разворачивающимися блоками, несколько загроможден.

Достоинства:

— Большая музыкальная библиотека;

— Регулярное добавление новинок и обновление чартов;

— Наличие возможности создавать плейлисты;

— Легко делиться музыкой с друзьями.

Недостатки:

— Огромное количество повторяющихся композиций;

— Часто всплывающая реклама;

— Отсутствует возможность сортировки.

Можем сделать вывод:

Приложение обладает приятным интерфейсом и не менее приятным функционалом, однако композиции не отсортированы, а реклама портит общее впечатление о приложении.

1.1.3. Deezer

Возможности приложения:

— Воспроизведение любых находящихся в библиотеке треков;

— Воспроизведение сохраненных композиций в режиме offline;

— Формирование своего плейлиста;

— Получение рекомендаций, на основе избранных композиций;

— Доступ к составленных плейлистам других пользователей;

— Поиск нужных композиций;

— Прослушивание интересующих подкастов;

— Просмотр текстов песен.

Теперь рассмотрим интерфейс приложения.

Рисунок 3 — Интерфейс приложения Deezer.

Достоинства:

— Большая музыкальная библиотека;

— Наличие подкастов;

— Рекомендации не только по трекам, но и по исполнителям, и по плейлистам;

— Расширенный поиск, включающий не только поиск по автору и названию, но и по тексту композиции;

Недостатки:

— Серьезные ограничения для бесплатной версии;

— Обязательная регистрация/авторизация для использования;

— Отсутствие возможности редактирования тестов песен;

— Отсутствие чартов и хит-парадов, а также радиостанций;

Сделаем вывод:

Приятный и интуитивный интерфейс, широкие возможности, в том числе по рекомендациям по разным критериям, однако все это значительно ограничено в бесплатной версии.

1.2. Постановка задачи

Целью данного проекта является разработка Android приложения «Музкат», которое, предназначено для составления индивидуального списка музыкальных предпочтений для каждого пользователя, на основании которого можно получить рекомендации музыкальных произведений, что поможет пользователям этого приложения расширить свой кругозор в музыке, найти новые музыкальные произведения.

1.2.1. Функциональные требования

— Возможность входа в аккаунт и регистрации неавторизированного пользователя с использованием логина и пароля;

— Возможность выхода из аккаунта авторизированного пользователя;

— Возможность просмотра информации о музыкальных записях: название, автор и жанр;

— Возможность задания авторизированным пользователем своих предпочтений: предпочитаемые авторы и жанры;

— Возможность авторизированного пользователя пополнять список доступных музыкальных записей: название произведения, автор и жанр.

— Возможность авторизированного пользователя получения списка музыкальных произведений, с учетом заданных предпочтений.

1.2.2. Технические требования

— Все запросы клиента серверу и ответы сервера клиенту осуществляются по протоколу HTTP, если явно не указано иное.

— Вход в аккаунт или регистрация осуществляется посредством отправки клиента серверу запроса о регистрации или входе с включенными в тело запроса введенного пользователем логина и пароля, где сервер проверяет и

одобряет или отклоняет вход или регистрацию, отправляя соответствующие ответы пользователю;

— Для хранения данных должна использоваться реляционная база данных;

— Приложение должно быть реализовано под Android версии 6 или старше, с использованием языка Java;

— При добавлении новой информации о музыкальных записях или изменениях предпочтений авторизированного пользователя производится отсылка запроса серверу, проверяющему на дубликаты записей в соответствии с ограничениями, заданными в используемой базе данных.

1.2.3. Требования к интерфейсу

— Возможность пропуска входа или регистрации неавторизированного пользователя с дальнейшим переходом к списку информации о музыкальных произведениях;

— Возможность просмотреть причину неудачного входа в аккаунт или его регистрацию при, соответственно, попытке входа или регистрации;

— При выходе из аккаунта интерфейс обновляется в соответствии с функциональными требованиями, а список музыкальных произведений сбрасывается на клиентской части с последующим запросом нового списка у сервера.

1.2.4. Макет интерфейса

В качестве предлагаемого для реализации интерфейса приложения был разработан макет экранов приложения, однако реализация может отличаться по усмотрению исполнителя, но с соответствием ранее описанным требованиям.

При входе в приложение пользователь видит экран личного кабинета для авторизации, для которой нужно ввести логин и пароль, нажать «Login» или «Logon» для входа или регистрации соответственно.

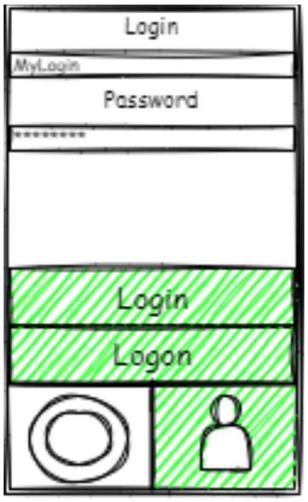

Рисунок 4 — Начальный экран – вход или регистрация

При неудачной попытке входа или регистрации пользователь будет уведомлен об этом с указанной причиной провала.

Рисунок 5 — Попытка входа провалена

Не выполнив авторизацию, пользователь может перейти к списку информации о музыкальных произведениях сразу, нажав по левой нижней кнопке с изображением диска.

Здесь каждая запись отображает в себе автора, название произведения и жанр.

Рисунок 6 — Неавторизированный пользователь пропустил авторизацию и перешел к списку информации о музыкальных произведениях.

Однако, если пользователь авторизовался, экран личного кабинета меняется – появляются возможности выхода из аккаунта (кнопка Logout), добавления (кнопка «+») или удаления (кнопка «-») предпочтений по авторам и жанрам.

Рисунок 7 — Личный кабинет авторизированного пользователя.

При добавлении новой записи в список своих предпочтений пользователю необходимо нажать «+», ввести предпочтение и нажать кнопку «Ок». Для краткости макета добавление автора и жанра объединены, а название соответствующей функции помечено как [DataTypeName].

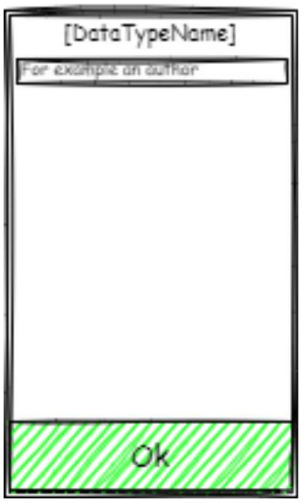

Рисунок 8 — Экран добавления новой записи в список предпочтений авторизированного пользователя.

Экран списка информации о музыкальных произведениях при авторизации также меняется: появляется возможность добавления новых записей, а также все записи теперь отсортированы по соответствию предпочтений.

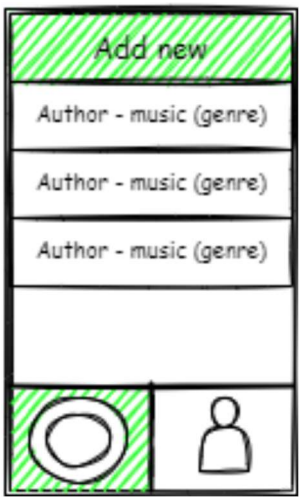

Рисунок 9 — Экран списка музыкальных произведений для авторизированного пользователя.

При добавлении новой записи авторизированным пользователем (кнопка «Add new») пользователю нужно ввести название записи, ее автора и жанр, после чего нажать кнопку «Ок».

1.3. Средства реализации

В качестве средств реализации для разработки мобильного приложения использованы:

— Java 11 (клиент и сервер)

— Spring Framework (сервер)

— Spring Boot (сервер)

— Hibernate (сервер)

— Java Persistence API (сервер)

— Lombok (сервер)

— XML при создании экранов приложения и стилей (клиент)

— IntelliJ IDEA Community Edition (сервер)

— Android Studio (клиент)

Перед очевидным аналогом – Kotlin – Java как язык программирования для андроид-приложений обладает преимуществом в размере сообщества разработчиков и информации по разработке приложений на этом языке, так как является более старым, что позволяет быстрее реализовывать решения и находить проблемы. Одиннадцатая версия была выбрана в связи с тем, что обладает долгосрочной поддержкой, и, в отличие от 8 или 17 версии, не является слишком старой или слишком новой, что могло бы привести к потере долгосрочной поддержки по мере существования приложения или к ошибкам со стороны языка по причине новизны версии.

Языком для сервера была выбрана Java в связи с использованием Spring Framework, поэтому будет лучше озвучить его преимущества:

— Spring Framework может быть задействован на всех архитектурных слоях, применяемых при разработке клиент-серверных приложений;

— Использует модель POJO при написании классов;

— Позволяет свободно связывать модули и легко их тестировать;

— Поддерживает декларативное программирование;

— Избавляет от самостоятельного создания фабричных и синглтон-классов;

— Поддерживает различные способы конфигурации;

— Предоставляет сервис уровня middleware.

Spring Boot обладает следующими преимуществами:

— Не требует развертывания WAR-файлов;

— Создает автономные приложения;

— Помогает напрямую встроить в приложение Tomcat, Jetty или Undertow;

— Не требует XML-конфигурации;

— Направлен на уменьшение объема исходного кода;

— Имеет дополнительную функциональность «из коробки»;

— Простота запуска;

— Простая настройка и управление.

Java Persistence API обладает следующими преимуществами:

— Разделяет SQL логику от бизнес логики, что позволяет думать более абстрактно и ориентироваться на свои классы, а это, в свою очередь, также облегчает сопровождение;

— Нет необходимости контролировать количество «?» в SQL-подобных запросах;

— Есть совместимость с библиотеками валидации данных;

Главное преимущество Lombok в том, что он значительно сокращает код, делает его более читаемым при помощи аннотаций, добавляя соответствующие им методы, контролируя уровни доступа. Это снимает рутинную работу с разработчика и, поэтому, ускоряет создание приложения.

Преимущество XML-разметки для создания экранов приложения перед ручным созданием элементов для экранов:

— Легкая портируемость и мобильность без особой привязки к какому-либо коду;

— Более краткая и удобная настройка элементов за счёт тегов и атрибутов;

IntelliJ IDEA Community Edition была выбрана как среда разработки, так как обладает следующими преимуществами:

— Умное автодополнение кода;

— Качественная и удобная отладка;

— Поддержка системы контроля версий;

— Интуитивный интерфейс;

— Есть инструменты для безопасного рефакторинга и автогенерации кода;

И, наконец, Android Studio обладает такими преимуществами:

— Удобный дизайнер пользовательских интерфейсов, позволяющий облегчить визуальное проектирование приложения;

— XML редактор с подсветкой синтаксиса и умным автодополнением;

— Поддержка системы контроля версий;

— Поддержка эмуляции устройств;

— Возможность проводить тестирование и анализ кода;

— Высокая скорость сборки приложения;

— Поддержка рендера средствами GPU.

2. Реализация

2.1. Архитектура приложения

2.1.1. База данных

В качестве базы данных используется реляционная на основе СУБД PostgreSQL.

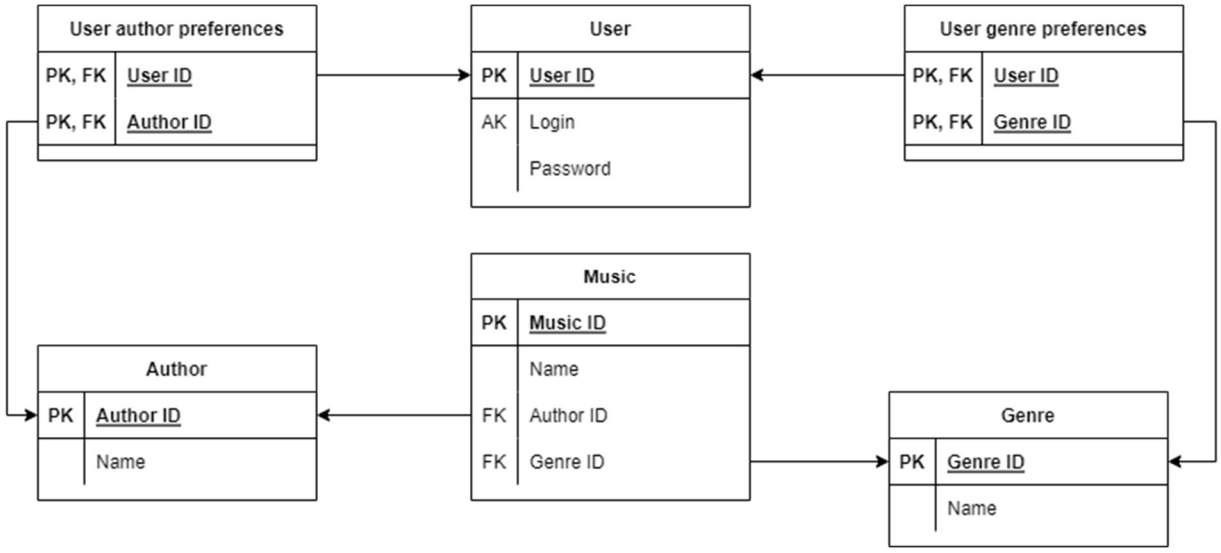

Рисунок 10 — Схема базы данных серверной части приложения.

Таблица User предназначена для хранения данных для авторизации пользователей, однажды успешно создавших аккаунт.

Она имеет такие атрибуты как:

— User ID. Главный ключ. Домен: целые числа. Уникальный идентификатор пользователя. Назначается автоматически при помощи СУБД;

— Login. Альтернативный ключ. Домен: строки. Логин пользователя;

— Password. Домен: строки. Пароль пользователя.

Таблица Genre предназначена для хранения информации о всех существующих в приложении жанрах.

Она имеет такие атрибуты как:

— Genre ID. Главный ключ. Домен: целые числа. Уникальный идентификатор жанра. Назначается автоматически при помощи СУБД;

— Name. Домен: строки. Название жанра.

Таблица Author предназначена для хранения информации о всех существующих в приложении авторах, причем не в буквальном смысле, то есть автором может считаться исполнитель или группа.

Она имеет такие атрибуты как:

— Author ID. Первичный ключ. Домен: целые числа. Уникальный идентификатор автора. Назначается автоматически при помощи СУБД;

— Name. Домен: строки. Имя автора не в буквальном смысле, то есть здесь может быть фамилия, имя, отчество, псевдоним или их комбинация.

Таблица Music предназначена для хранения информации о всех музыкальных произведениях, существующих в приложении.

Она имеет такие атрибуты как:

— Music ID. Первичный ключ. Домен: целые числа. Уникальный идентификатор музыкального произведения. Назначается автоматически при помощи СУБД;

— Name. Домен: строки. Название музыкального произведения;

— Author ID. Внешний ключ, ссылающийся на Author ID таблицы Author. Домен: целые числа. Указывает на уникальный идентификатор автора данного музыкального произведения;

— Genre ID. Внешний ключ, ссылающийся на Genre ID таблицы Genre. Домен: целые числа. Указывает на уникальный идентификатор жанра данного музыкального произведения.

Таблица User author preferences

Данная таблица предназначена для хранения информации о предпочитаемых пользователями авторах.

Она имеет такие атрибуты как:

— User ID. Первичный, внешний и составной в паре с Author ID ключ. Домен: целые числа. Уникальный идентификатор пользователя, имеющего в предпочтениях автора с уникальным идентификатором Author ID данной таблицы;

— Author ID. Первичный, внешний и составной в паре с User ID ключ. Домен: целые числа. Уникальный идентификатор автора, которого в качестве одного из предпочитаемых выбрал пользователь с уникальным идентификатором User ID данной таблицы.

Таблица User genre preferences предназначена для хранения информации о предпочитаемых пользователями жанрах.

Она имеет такие атрибуты как:

— User ID. Первичный, внешний и составной в паре с Genre ID ключ. Домен: целые числа. Уникальный идентификатор пользователя, имеющего в предпочтениях жанр с уникальным идентификатором Genre ID данной таблицы;

— Genre ID. Первичный, внешний и составной в паре с User ID ключ. Домен: целые числа. Уникальный идентификатор жанра, которого в качестве одного из предпочитаемых выбрал пользователь с уникальным идентификатором User ID данной таблицы.

2.1.2. Варианты использования

Рассмотрим сначала диаграмма вариантов использования неавторизованного пользователя.

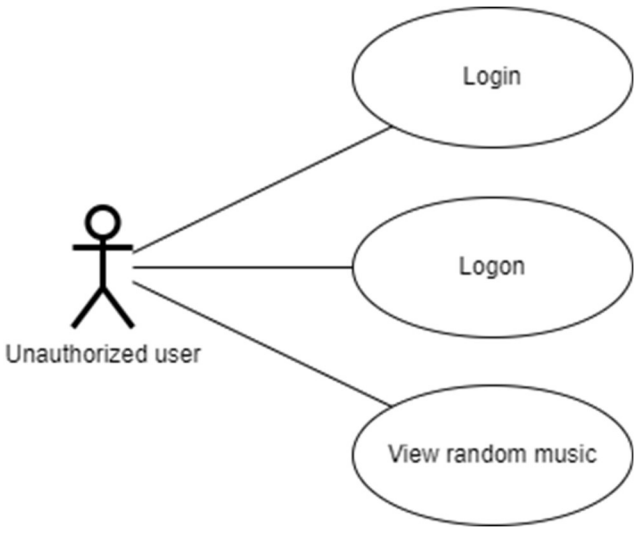

Рисунок 11 — Use-case диаграмма для неавторизованного пользователя

Также рассмотрим диаграмму использования для авторизированного пользователя.

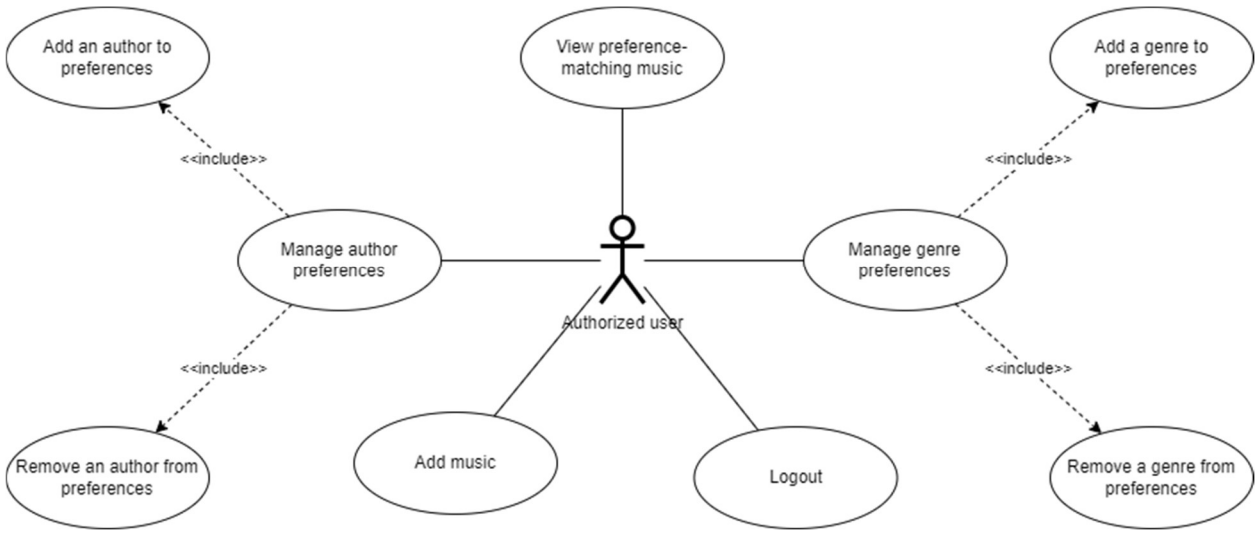

Рисунок 12 — Use-case диаграмма для авторизованного пользователя

2.1.3. Диаграммы классов

В данном пункте будут рассмотрены диаграммы только основных классов клиентской части приложения.

Одними из таких классов являются сущности, в которых хранятся полученные от сервера данные из базы данных.

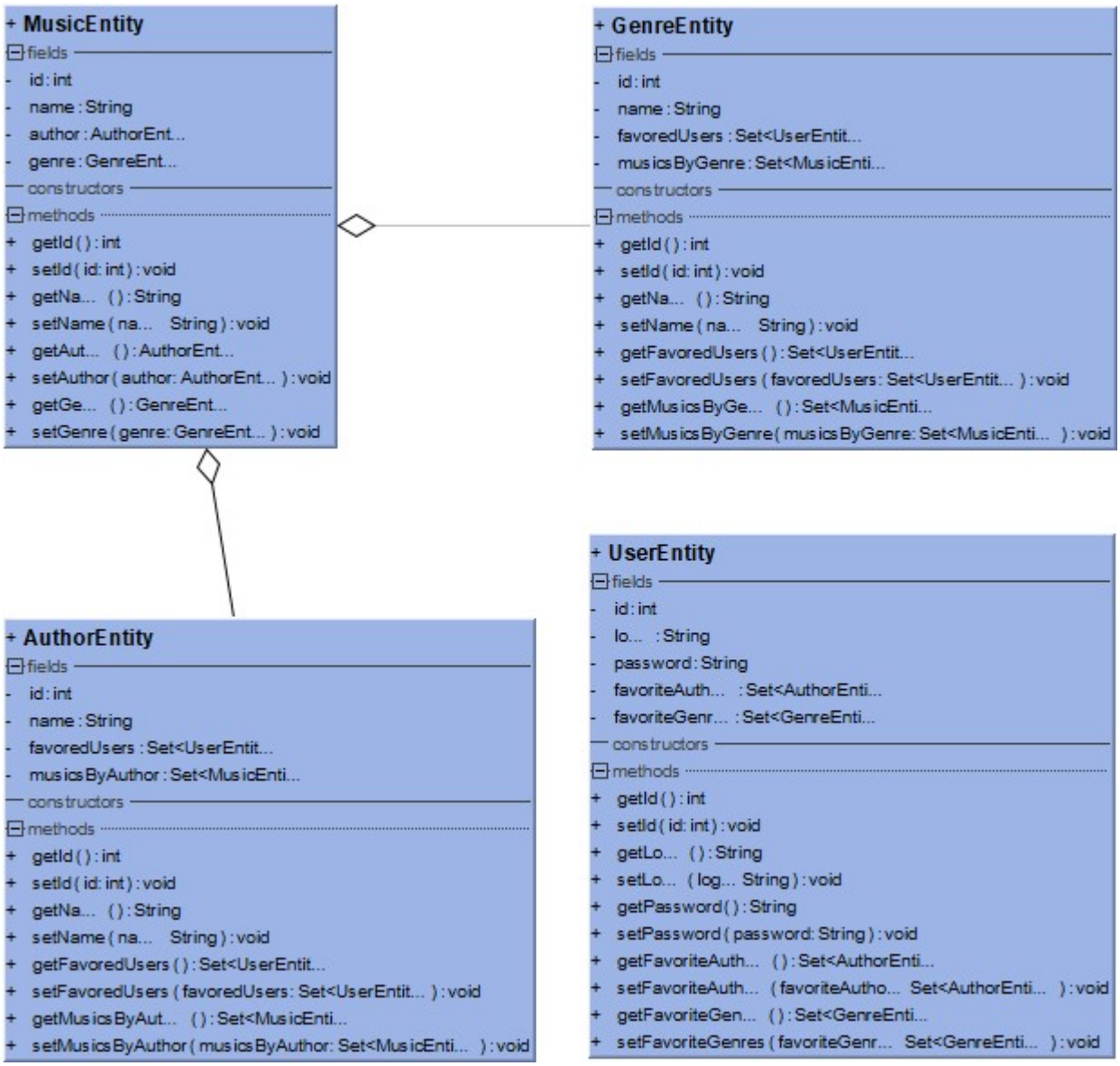

Рисунок 13 — Диаграмма классов сущностей клиентской части приложения.

Также стоит рассмотреть диаграмму классов API, которые играют важную роль в клиент-серверном взаимодействии на стороне клиента.

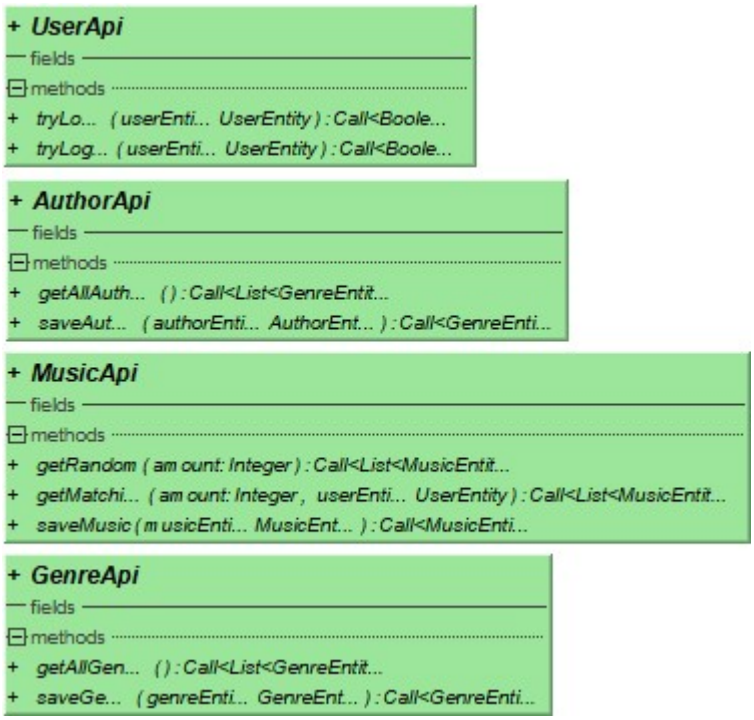

Рисунок 14 — Диаграмма классов API, взаимодействующих с серверной частью приложения.

2.1.4. Диаграммы последовательностей

Рассмотрение диаграмм последовательностей или их фрагментов из сторонней библиотеки (например, логика взаимодействия Spring Framework и JPA с базой данных, отправка ответа сервера клиенту) рассматриваться не будет, так как выходит за рамки данного отчета.

В связи с этим, на данный момент на клиентской части приложения большинство возможных диаграмм являлись бы тривиальным запросом у сервера необходимых данных, поэтому будут рассмотрены только некоторые диаграммы, связанные с графическим интерфейсом.

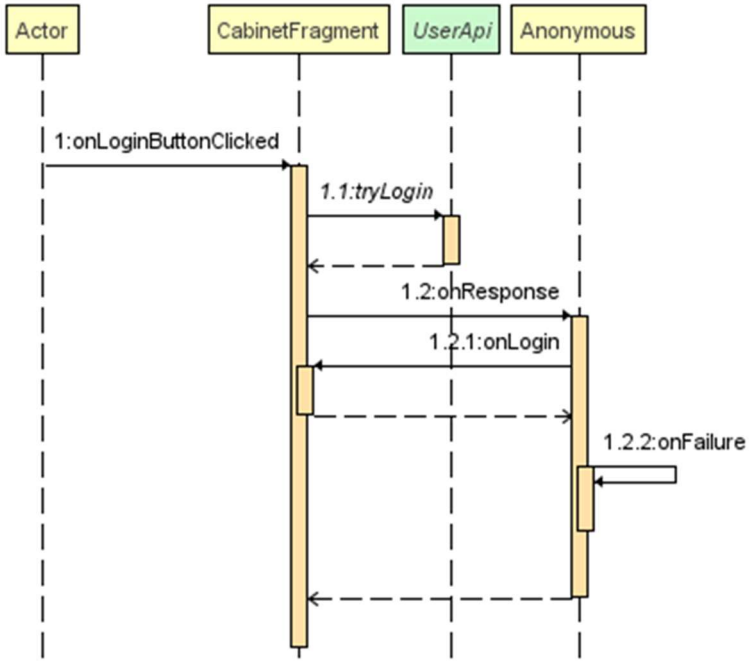

Рисунок 15 — Диаграмма последовательностей для логики нажатия на кнопку входа в аккаунт.

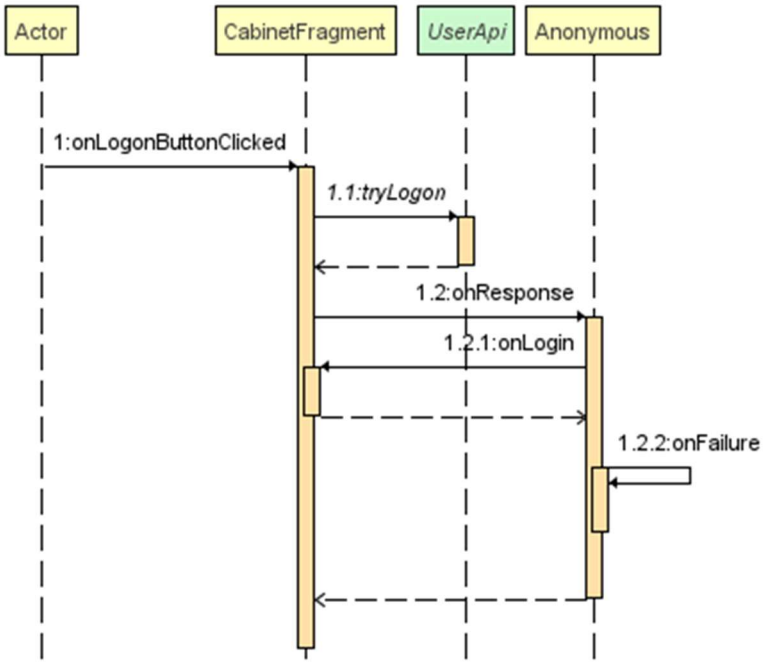

Рисунок 16 — Диаграмма последовательностей для логики нажатия на кнопку регистрации.

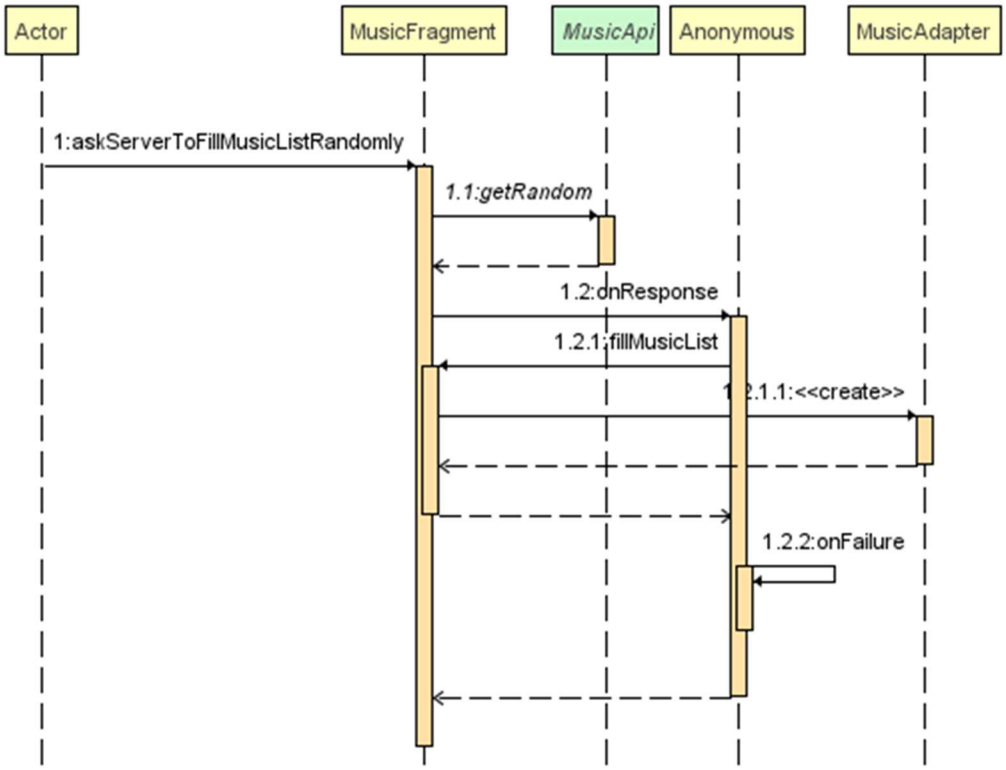

Рисунок 17 — Диаграмма последовательностей для логики заполнения экрана со списком музыки случайно для неавторизированного пользователя (при переходе на экран с музыкой в первый раз).

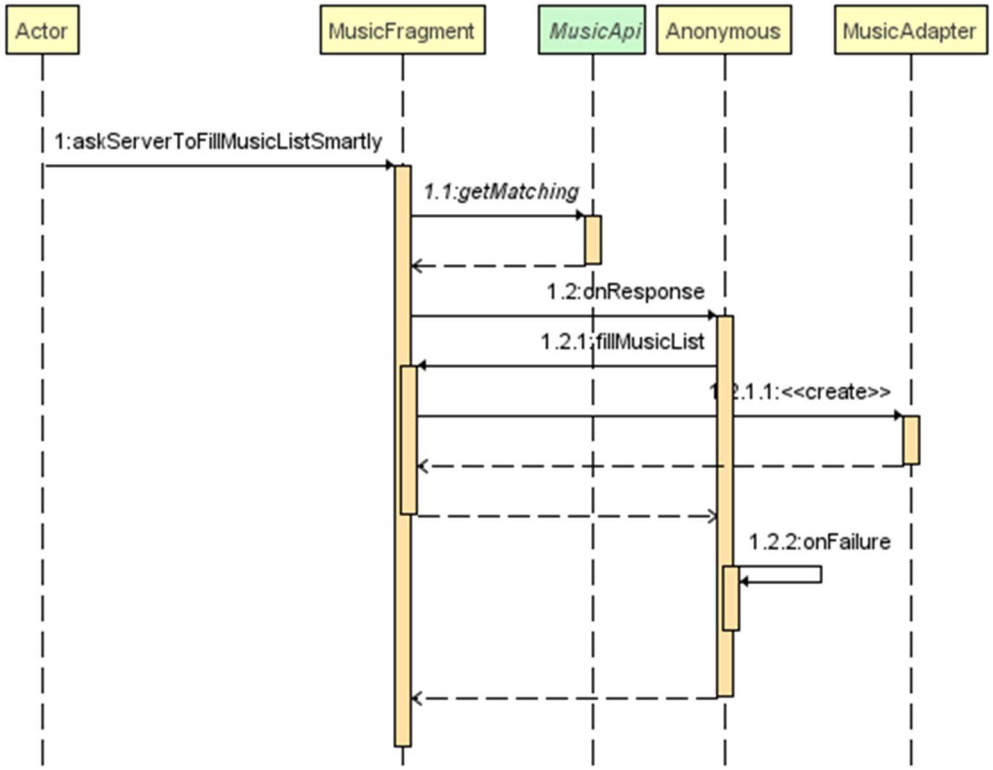

Рисунок 18 — Диаграмма последовательностей для логики заполнения экрана со списком музыки в соответствии с заданными предпочтениями для

авторизированного пользователя (при переходе на экран с музыкой в первый раз).

Диаграммы последовательностей сервера без учета логики сторонних библиотек также являются тривиальными и не представляют интереса, так как представляют из себя обращение к базе данных с некоторым запросом.

3. Приложение

3.1. Календарный план

Так как часть описанных выше требований удовлетворена, на календарном плане присутствуют только неудовлетворенные требования.

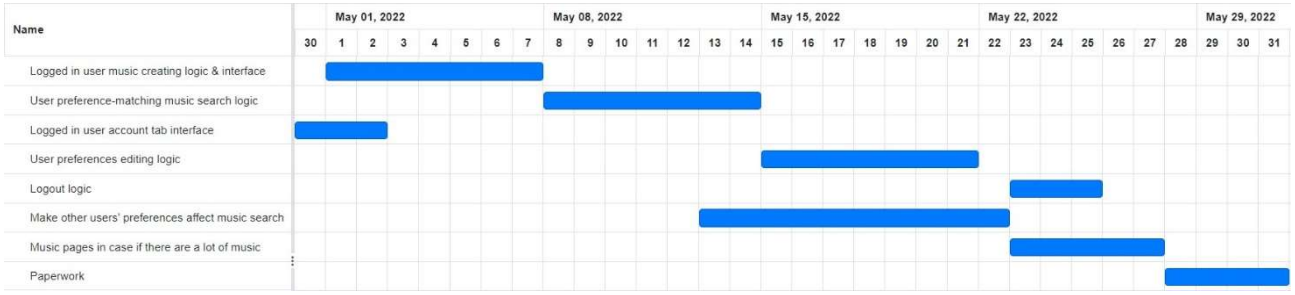

Рисунок 19 — Календарный план разработки приложения на момент начала мая.

Исполнитель оставляет за собой право отклонения от данного плана с сохранением срока приемки и сдачи, обозначенном в техническом задании.