МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

Направление специальности 1-40 01 01 10 Программное обеспечение информационных технологий (программирование интернет-изданий)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированные технологии и стандарты проектирования»

Тема «Музыкальный сервис VMusic»

Исполнитель

студент 2 курса группы 6 Валько Сергей Александрович

Руководитель работы ассистент Радиванович Д.А.

Курсовой проект защищён с оценкой

Председатель Пацей Н.В.

Минск 2021

Содержание

[ВВЕДЕНИЕ 3](#_Toc73057116)

[1. Аналитический обзор литературы и прототипов 4](#_Toc73057117)

[2. Анализ требований к курсовому проекту и разработка функциональных требований 6](#_Toc73057118)

[2.1. Постановка задачи 6](#_Toc73057119)

[2.2. Средства разработки 6](#_Toc73057120)

[2.3. Функциональные возможности приложения 7](#_Toc73057121)

[3. Проектирование программного средства 8](#_Toc73057122)

[3.1. Архитектура приложения 8](#_Toc73057123)

[3.2. Проектирование базы данных 11](#_Toc73057124)

[4. Создание (реализация) программного средства 14](#_Toc73057125)

[4.1. Структура пакетов приложения 14](#_Toc73057126)

[4.2. Реализация сущностей приложения 15](#_Toc73057127)

[4.3. Реализация ViewModel 16](#_Toc73057128)

[4.4. Реализация представления 18](#_Toc73057129)

[5. Тестирование, проверка работоспособности и анализ полученых результатов 20](#_Toc73057130)

[6. Руководство по использованию 25](#_Toc73057131)

[6.1. Руководство пользователя 25](#_Toc73057132)

[6.2. Руководство администратора 32](#_Toc73057133)

[ЗАКЛЮЧЕНИЕ 35](#_Toc73057134)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 36](#_Toc73057135)

[Приложение А 37](#_Toc73057136)

[Приложение Б 38](#_Toc73057137)

[Приложение В 39](#_Toc73057138)

[Приложение Г 40](#_Toc73057139)

[Приложение Д 41](#_Toc73057140)

[Приложение Е 42](#_Toc73057141)

# ВВЕДЕНИЕ

Независимо, что люди делают: отдыхают, занимаются спортом, работают – они слушает музыку. Прослушивание музыки прочно вжилось в повседневную жизнь человека. Она помогает: сосредоточится, отдохнуть, получить заряд необходимый эмоций или направить поток мыслей в необходимое русло.

Существует уже немалое количество программных средств и сервисов, обеспечивающих возможность прослушивать треки, в пример можно привести такие сервисы, как: Spotify, Яндекс.Музыка, Youtube Music и др. Также в пример можно привести аудиоплееры: AIMP, KMPlayer и т.д.

Программное средство «Музыкальный сервис VMusic» - это приложение, позволяющее прослушивать предоставляемую коллекцию музыки, создавать плейлисты и прослушивать их, а также оценивать треки и добавлять их в избранные.

В приложении имеется 2 типа пользователей: администратор и пользователь. Администратор формирует и корректирует коллекцию треков, обновляет список популярный треков, а также производит управление пользователями. Пользователь имеет возможность пользоваться основными возможностями приложения, которые были описаны ранее.

Прежде чем преступить к использованию приложения необходимо авторизоваться, а не зарегистрированным пользователем пройти стадию регистрации.

В основы приложения положено применение принципов ООП, архитектурного паттерна MVVM, паттернов проектирования и работы с БД.

Целью данного проекта является получение опыта в разработке настольных приложений под Windows, использовании принципов ООП и паттернов проектирования, а также предоставить удобное средство для прослушивания музыки в повседневном использовании.

# Аналитический обзор литературы и прототипов

За основу моего приложения был взят музыкальный сервис Spotify. Данное приложение представляет из себя интернет-сервис потокового аудио и предоставляет возможности прослушивания музыки, аудиокниг и подкастов, не скачивая их на устройство. Доступ к нему можно получить через сайт, приложения для различных ОС, а также через приложения для мобильных устройств и медиа-систем автомобилей. Естественно из всего выше предложенного многообразия средств за шаблон было взято настольное приложения под Windows.

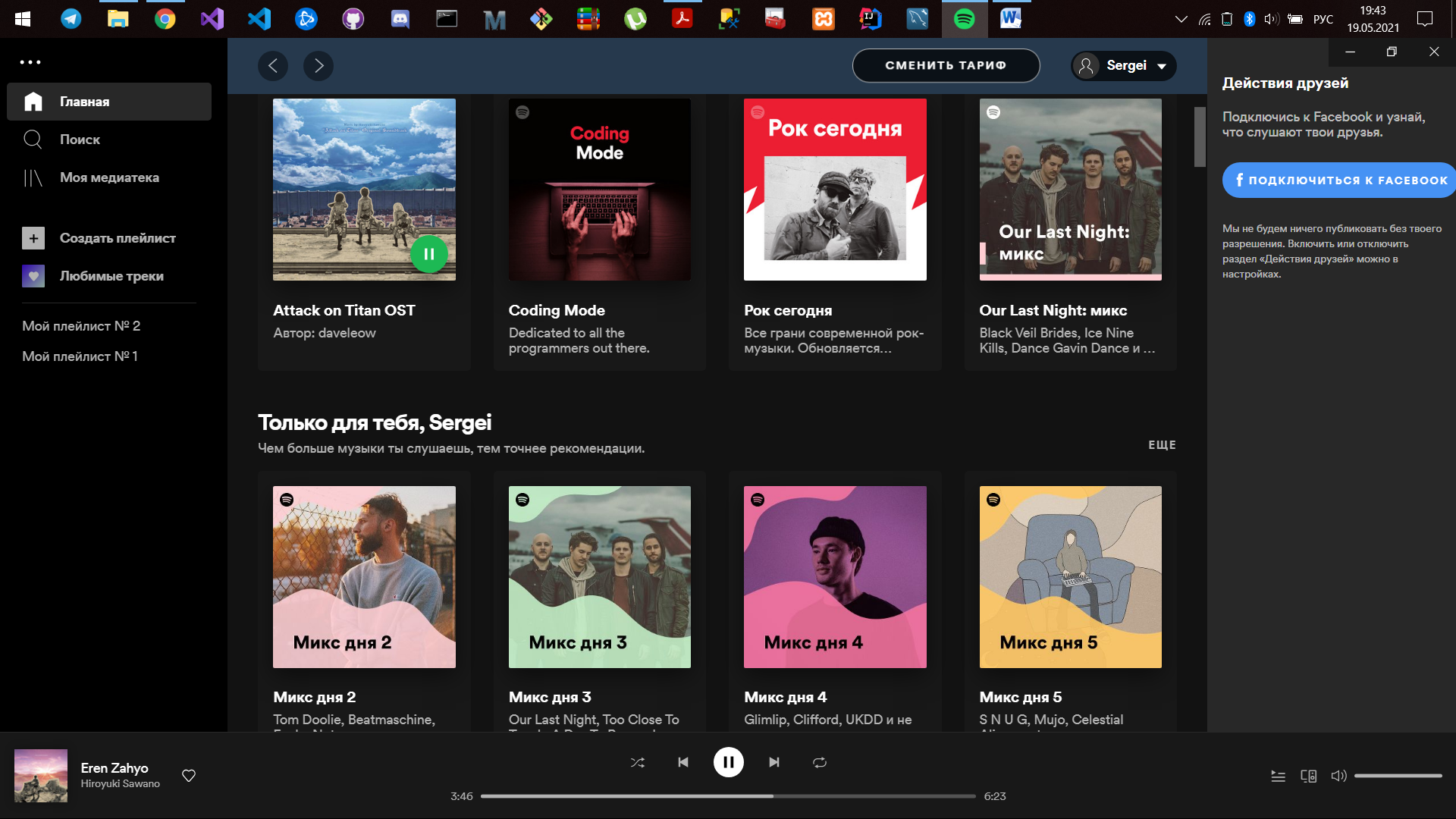


Рисунок 1.1 – Окно приложения «Spotify»

Снизу расположен плеер приложения при помощи которого пользователь имеет возможность управлять прослушиванием трека: изменять громкость, ставить на паузу, перематывать. Слева расположена панель меня, через которое производится навигация по приложению. В центральной части окно происходит отображения основного контента: треков, плэйлистов и т.д.

Имеется возможность как пользовательского поиска так и по жанрам.

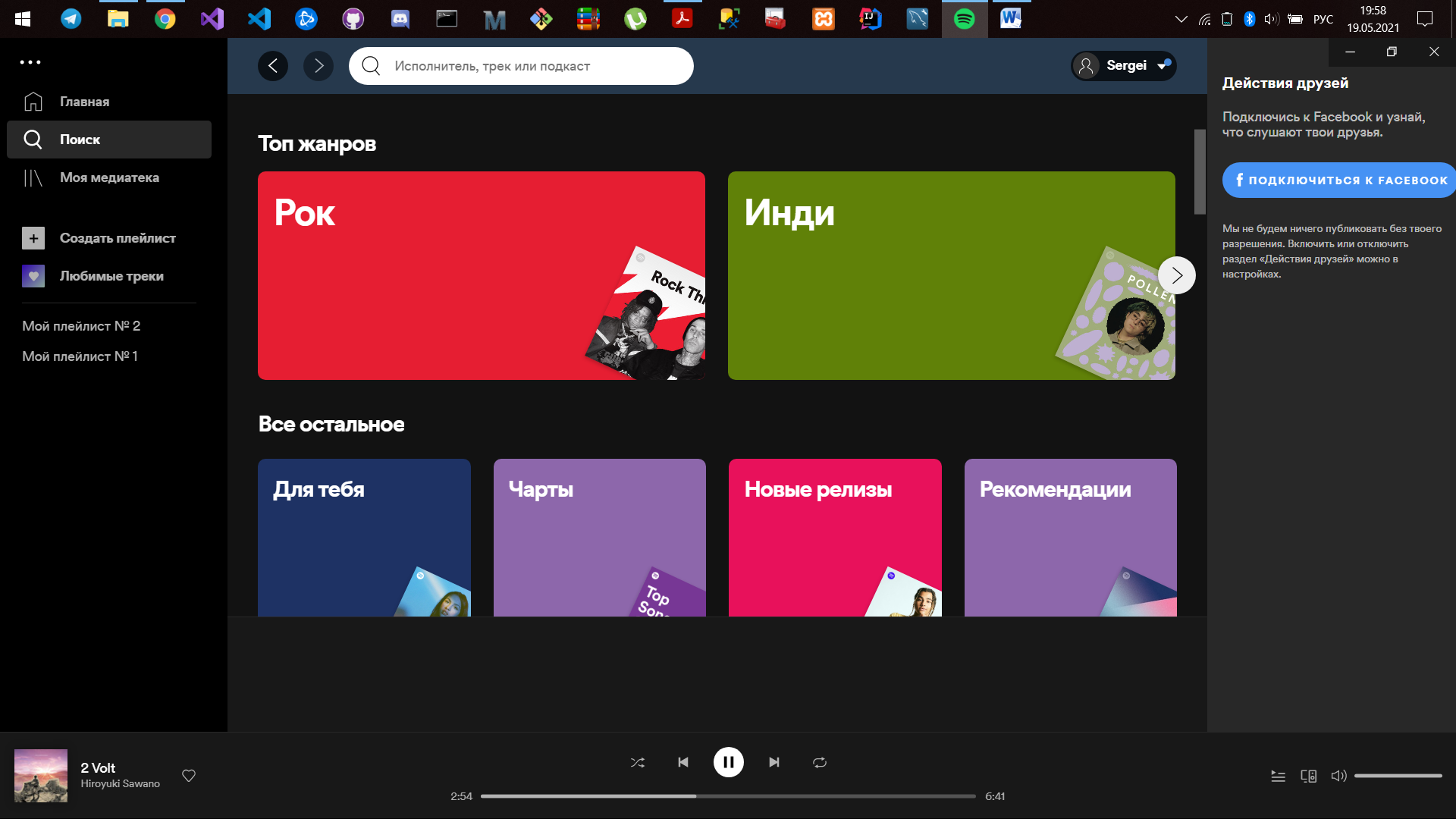


Рисунок 1.2 – Поиск в приложении «Spotify»

По статистике 70% процентов контента воспроизводится из плейлистов. И в приложении предусмотрена пользовательская медиатека, которая предоставляет возможности: управления плэйлистами, создания пользовательских, а также подписываться на понравившихся исполнителей или добавлять любимые альбомы.

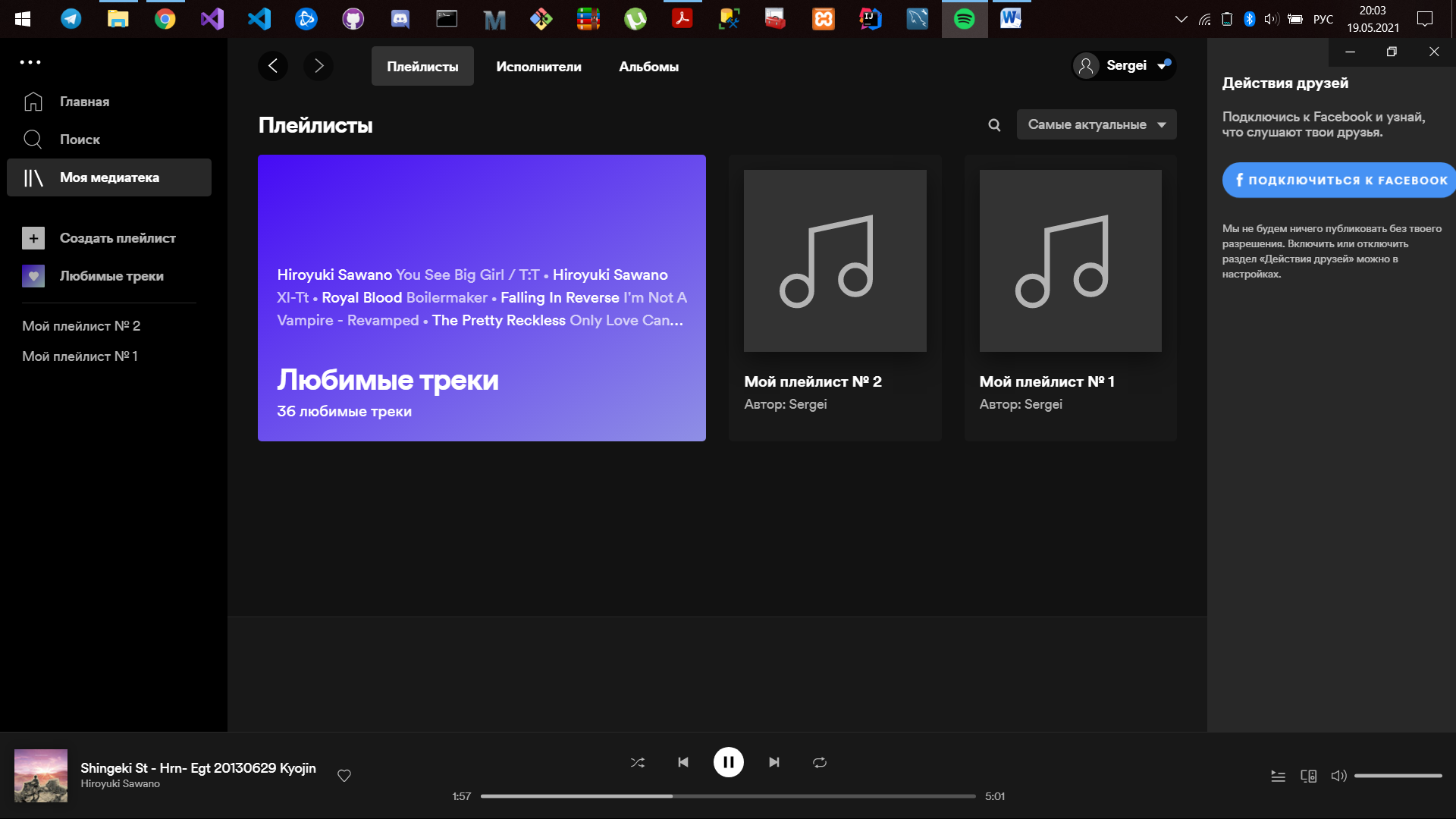


Рисунок 1.3 – Пользовательская медиатека

Данное приложение обладает платной подпиской которая позволяет избавится от назойливой рекламы, убрать ограничение скипов, получить возможность скачивать треки и слушать их оффлайн.

Я решил взять за основу данное приложение так, как оно по сути наиболее близко к моему: обладает функциями прослушивания музыки и составления/прослушивания плэйлистов. Также оно обладает довольно приятным и в своё время простым дизайном.

# Анализ требований к курсовому проекту и разработка функциональных требований

### Постановка задачи

Задачей курсового проекта является разработка приложения, предназначенного для прослушивания музыки с дополнительными возможностями: создание и прослушивание плейлистов, оценивание треков и добавление их в избранные. Коллекция треков пополняется и редактируется администратором. Администратор также производит управление пользователями и обновляет коллекцию популярных треков.

Для разработанного продукта необходимо предусмотреть следующие задачи:

* графический интерфейс пользователя;
* механизм авторизации;
* отображение коллекции треков;
* механизм управления контентом для администратора;
* механизм поиска треков;
* механизм создания и прослушивания плейлистов для пользователя.

Приложение необходимо реализовать с использованием шаблона проектирования MVVM и с организацией взаимодействия с базой данных. В качестве клиентской части необходимо реализовать оконное приложение с использованием стандартных и иных библиотек.

### Средства разработки

Программный продукт должен быть реализовать на объектно-ориентированном языке программирования C# с использованием технологии WPF. Windows Presentation Foundation (WPF) — система для построения клиентских приложений Windows, с визуально привлекательными возможностями взаимодействия с пользователем; графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML.

База данных была разработана в Microsoft SQL Server.

SQL Server является одной из наиболее популярных систем управления базами данных (СУБД) в мире.

SQL Server характеризуется такими особенностями как:

* производительность, SQL Server работает очень быстро;
* надежность и безопасность, SQL Server предоставляет шифрование данных;
* простота, с данной СУБД относительно легко работать и вести администрирование.

Для организации баз данных MS SQL Server использует реляционную модель, которая предполагает хранение данных в виде таблиц, каждая из которых состоит из строк и столбцов. Каждая строка хранит отдельный объект, а в столбцах размещаются атрибуты этого объекта.

Для взаимодействия с базой данных применяется язык SQL (Structured Query Language). Клиент (например, внешняя программа) отправляет запрос на языке SQL должным образом интерпретирует и выполняет запрос, а затем посылает клиенту результат выполнения.

Основной используемый язык запросов —Transact-SQL — реализован на структурированном языке запросов (SQL) с расширениями.

Для осуществления связи между базой данных и приложением на C# необходим посредник, которым будет являться технология Entity Framework. Она основана на платформе .NET Framework и предназначена для работы с данными.

Entity Framework предполагает три возможных способа взаимодействия с базой данных:

* database first: Entity Framework создает набор классов, которые отражают модель конкретной базы данных;
* model first: сначала разработчик создает модель базы данных, по которой затем Entity Framework создает реальную базу данных на сервере;
* code first: разработчик создает класс модели данных, которые будут храниться в базе данных, а затем Entity Framework по этой модели генерирует базу данных и ее таблицы.

В курсовом проектировании для разработки был выбран третий подход.

### Функциональные возможности приложения

Программное средство предоставляет пользователю следующие функциональные возможности:

* прослушивание треков;
* поиск треков;
* создание плейлистов;
* прослушивание плейлистов;
* редактирование плейлистов;
* редактирование приватной информации пользователя(смена пароля);
* возможность оценки треков;
* добавление треков в избранные.

Возможности администратора:

* добавление и удаление треков;
* редактирование информации о треках;
* обновление коллекции популярных треков;
* удаление и блокировка пользователей;
* передача прав администратора.

Наглядно это видно на UML-схеме возможностей (Приложение А).

# Проектирование программного средства

### Архитектура приложения

В разрабатываемом приложении используется архитектурный шаблон Model-View-ViewModel (MVVM).

На рисунке 3.1 представлена диаграмма, которая показывает общую структуру приложения в рамках шаблона MVVM.

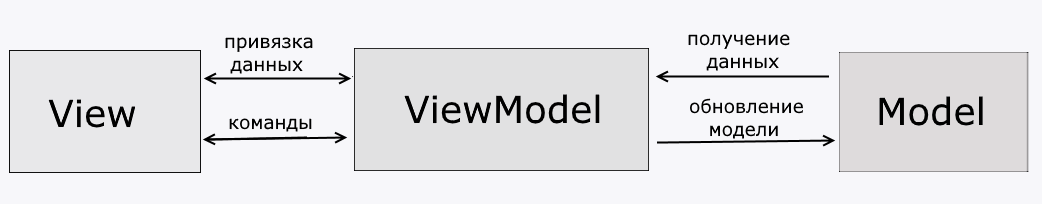


Рисунок 3.1 – Структура шаблона MVVM

Шаблон MVVM имеет три основных слоя: модель, которая представляет бизнес-логику приложения, представление пользовательского интерфейса, и представление-модель, в котором содержится вся логика построения графического интерфейса и ссылка на модель, поэтому он выступает в качестве модели для представления.

Пользовательский интерфейс и качество его реализации играет далеко не последнее место в конечном результате, разработка эффективного интерфейса, приятного и удобного для конечного пользователя, является важной задачей. Поэтому для хорошего проектирования View необходимо понять, как пользователь будет взаимодействовать с приложением. Для этого была составлена схема на рисунке 3.2, на которой представлен принцип работы приложения с точки зрения пользователя.

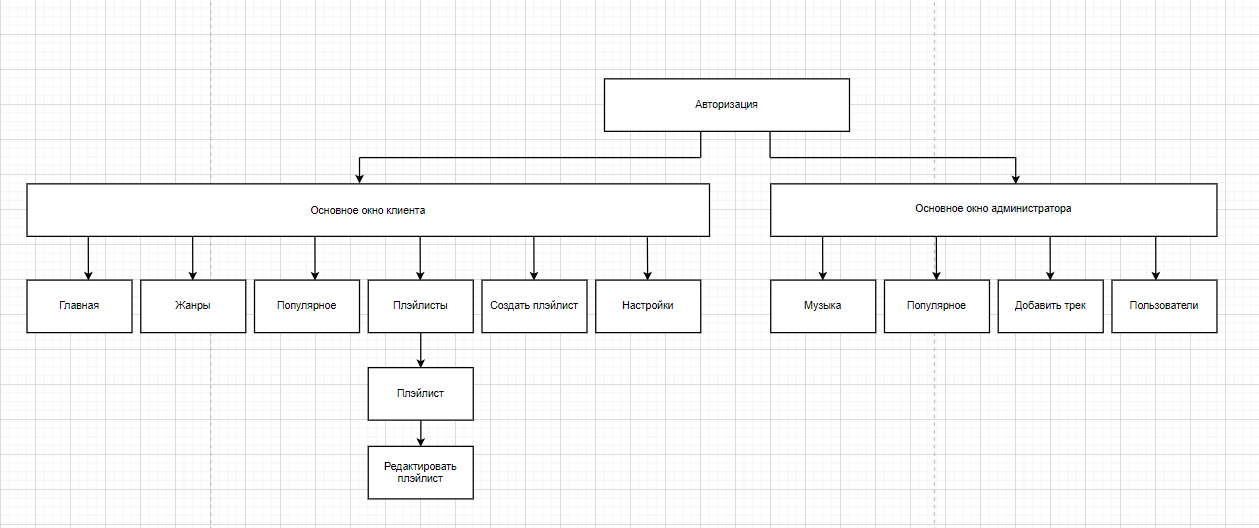


Рисунок 3.2 – Общая схема работы приложения

При запуске приложения появляется окно авторизации, которое требует ввода логина и пароля для начала работы.

В случае, если вас нету своей учётной записи, её можно создать, нажав на кнопку «Регистрация» в окне авторизации. Вы будете направлены в окно регистрации, где вам нужно будет заполнить все поля. После успешной регистрации, вас перенаправит в окно авторизации для ввода ваших данных.

Также есть кнопка перехода на вход в качестве администратора, где вы сможете, если ваш пользователь обладает должными правами, войти в окно управления приложения и воспользоваться возможностями администратора.

После ввода логина и пароля происходит их проверка и в случае успеха вам откроется основное окно программы, иначе будет выведено сообщение об ошибке.

Главное окно приложения предоставляет возможность навигации по составным компонентам текущего окна (страницам).

Рассмотрим главное окно пользователя в деталях. Оно содержит мини-плеер для управления текущим проигрыванием трека, элемент управления для поиска треков, панель меню для навигации по странице и основную часть, в которой собственно отображаются страницы.

При первоначальном переходе в данное окно, отображается главная страница. Далее уже в зависимости от действий пользователя может происходить переход на другие страницы, через панель меню. Пользователь также может искать треки через поисковую строку, результат будет отображаться на главной странице приложения.

Также пользователь обладает возможностью контролировать процесс воспроизведения треков с помощью проигрывателя. Проигрыватель воспроизводит треки из текущего набора, имеется возможность перехода на следующий или предыдущий трек, регулировки звука. Пользователь может изменить проигрываемый трек и собственно проигрываемый набор, кликнув по треку из набора в главной области программы.

Панель меню содержит кнопки перехода на следующие страницы:

* главная;
* жанры;
* популярное;
* плэйлисты;
* создать плэйлист;
* настройки.

Рассмотрим каждую страницу по отдельности.

Главная страница отображает список всех треков из коллекции.

Страница «Жанры» отображает список имеющихся жанров и имеется возможность получения списков треков по определённому жанру.

Страница «Популярное» отображает формируемый администратором список популярных треков.

Страница «Плэйлисты» отображает набор плэйлистов соответствующий данному пользователю (в том числе и избранные треки). Отсюда пользователь может перейти к конкретному плэйлисту, запустить его прослушивание, а также перейти на страницу редактирования.

Со страницы редактирования пользователь может изменить информацию о плэйлисте, удалить его или изменить его содержимое.

На странице создания плэйлиста пользователь заполняет все необходимые данные, ищет треки и добавляет их в плэйлист. После создания плэйлиста произойдёт переадресация на страницу отображающую все плэйлисты.

Через страницу настроек пользователь имеет возможность изменить пароль, удалить аккаунт или выйти из текущего аккаунта.

Рассмотрим главное окно администратора. Оно состоит из панели меню и области отображения. С помощью панели меню происходит страничная навигация. Наглядно работа с данным окном продемонстрирована на UML-схеме последовательности (Приложение Б).

Панель меню содержит следующие страницы:

* музыка;
* популярное;
* добавить трек;
* пользователи.

Рассмотрим каждую страницу отдельно.

На данной странице отображается список всех треков. Администратор может удалять треки и списка, а также редактировать, для этого выделена отдельная страница. На странице редактирования администратор может изменить информацию о треке.

На странице «Популярное» отображается список 10 популярных треков и соответственно администратор может изменять его при устаревании.

Страница добавления трека позволяет заполнить необходимые данные и добавить трек в коллекцию.

На странице «Пользователи» предоставляется возможность просмотра списка пользователей, удалять их, блокировать и разблокировать, передавать права администратора другим пользователям.

В качестве сервиса для хранения данных была выбрана СУБД Microsoft SQL Server. На диаграмме размещение на рисунке 3. показано графическое представление инфраструктуры приложения.

Часть хранимых данных, а именно аудио-файлы, сохраняются в локальном файловом хранилище, так как их размер велик.

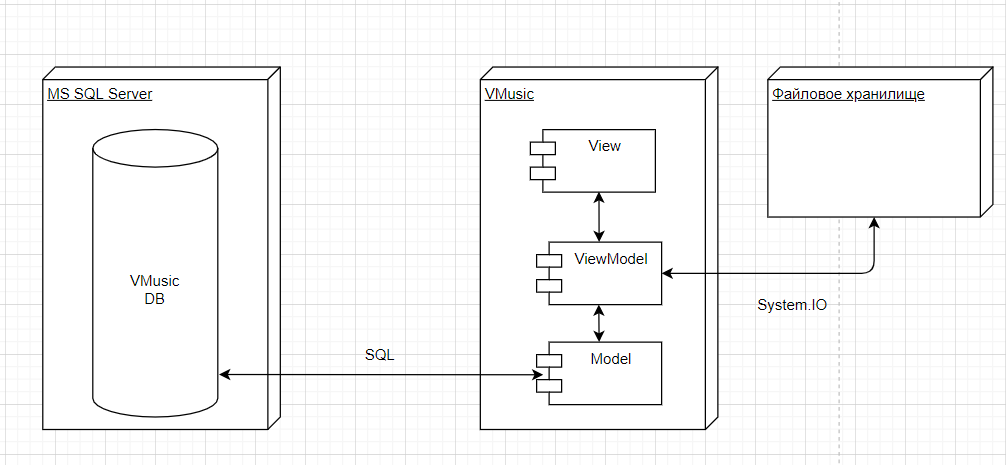


Рисунок 3.3 – Диаграмма размещения

Связь приложения с базой данных Microsoft SQL Server происходит с помощью SQL-команд.

### Проектирование базы данных

Проектирование баз данных — процесс создания схемы базы данных и определения необходимых ограничений целостности.

Основные задачи проектирования базы данных:

* обеспечение хранения в БД всей необходимой информации;
* обеспечение возможности получения данных по всем необходимым запросам;
* сокращение избыточности и дублирования данных;
* обеспечение целостности базы данных.

Проектирование базы данных проводится в два этапа: концептуальное и логическое проектирование.

Концептуальное (инфологическое) проектирование — построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. В результате этого этапа создаётся ER-модель. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных.

Основными понятиями ER-модели являются: сущность, связь и атрибут.

Сущность – это реальный или представляемый объект, информация о котором должна сохраняться и быть доступна.

Связь – это графически изображаемая ассоциация, устанавливаемая между двумя сущностями. Эта ассоциация обычно является бинарной и может существовать между двумя разными сущностями или между сущностью и ей же самой (рекурсивная связь).

Атрибут сущности это любая детaль, которая служит для уточнения, идентификации, классификации, числовой характеристики или выражения состояния сущности.

Логическое проектирование — создание схемы базы данных на основе конкретной модели данных, например, реляционной модели данных. Для реляционной модели данных логическая модель — набор схем отношений, обычно с указанием первичных ключей, а также «связей» между отношениями, представляющих собой внешние ключи. На этапе логического проектирования учитывается специфика конкретной модели данных, но может не учитываться специфика конкретной СУБД.

Логическая модель базы данных представлена на рисунке 3.4.

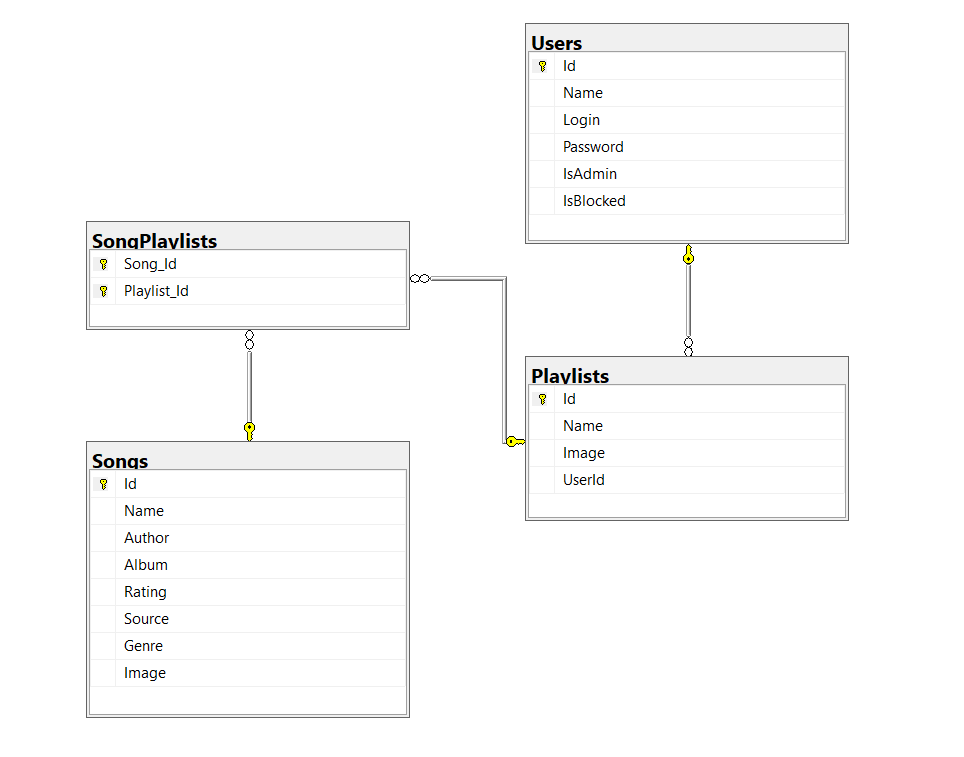


Рисунок 3.4 – Логическая модель базы данных

В базе данных содержится 4 таблицы:

* Users;
* Playlists;
* SongPlaylists;
* Songs.

Таблица Users хранит информацию о пользователях приложения. Она содержит столбцы:

* Id;
* Name;
* Login;
* Password;
* IsAdmin;
* IsBlocked.

Столбец Name хранит имя пользователя, Password и Login данные необходимые для авторизации. Столбец IsAdmin определяет является ли данный пользователь администратором, IsBlocked – заблокирован ли пользователь.

Таблица Songs хранит информацию о треках. Она содержит столбцы:

* Id;
* Name;
* Author;
* Album;
* Rating;
* Source;
* Genre;
* Image.

Столбец Name хранит название трека, Album – название альбома, Author – лицо или группа лиц исполняющее музыкальное произведение, Rating – рейтинг треков (сколько раз данный трек был добавлен в избранное), Source – относительный путь к файлу в локальном хранилище, Genre – жанр произведения, Image – изображение-заставка трека.

Таблица Playlists содержит информацию о плэйлистах пользователей. Она представлена столбцами:

* Id;
* Name;
* Image;
* UserId.

Столбец Name хранит название плэйлиста, Image – изображение-заставку, UserId – id пользователя, который является его владельцем.

Таблица SongPlaylists – промежуточная таблица, которая служит для реализации отношения многие ко многим и служит для хранения информации: какие треки относятся к каким плэйлистам. Она содержит столбцы:

* Song\_Id;
* Playlist\_Id.

Данные столбцы хранят id трека и плэйлиста.

# Создание (реализация) программного средства

### Структура пакетов приложения

При создании приложения использовался паттерн проектирование MVVM. Он заключается в разделении представления от бизнес логики. Это достигается за счёт ввода новой логической конструкции ViewModel. Она связывает представление и бизнес логику приложения.

Пакетная структура проекта показана на рисунке 4.1

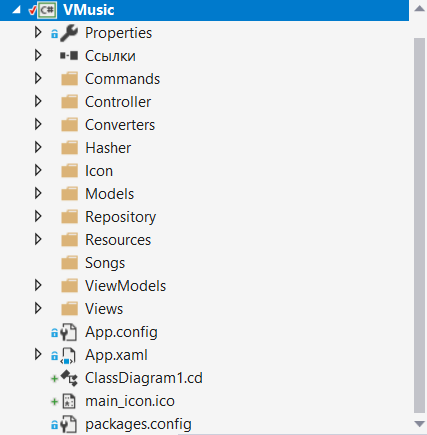


Рисунок 4.1 – Пакетная структура проекта

Более подробно структура проекта описана в таблице 4.

Таблица 4.1 – Описание структурных пакетов проекта

|  |  |
| --- | --- |
| **Имя пакета** | **Описание** |
| Models | Описание сущностей |
| Commands | Реализация Command |
| Controller | Бизнес-логика, вынесенная из ViewModel |
| Converters | Вспомогательные классы-конвертеры |
| Hasher | Содержит класс реализующий алгоритм хеширования md5 |
| Icon | Содержит иконку для окон приложения |
| Repository | Реализация UnitOfWork и Repository |
| Resources | Файлы стилей |
| Views | Представления |
| ViewModels | Основная бизнес-логика |
| App.config | Файл конфигурации приложения |
| App.xaml | Общие ресурсы приложения |

В этой таблице приведены основные логические составляющие пакеты, которые используются в работе приложения.

UML диаграмма классов показана в приложении (Приложение В).

### Реализация сущностей приложения

В данном программном решении был использован подход Code-First. При данном подходе модель EDMX не используется. Создание базы данных происходит из созданной вручную модели объектов C#.

Диаграмма классов UML для сущностных классов представлена на рисунке 4.2.

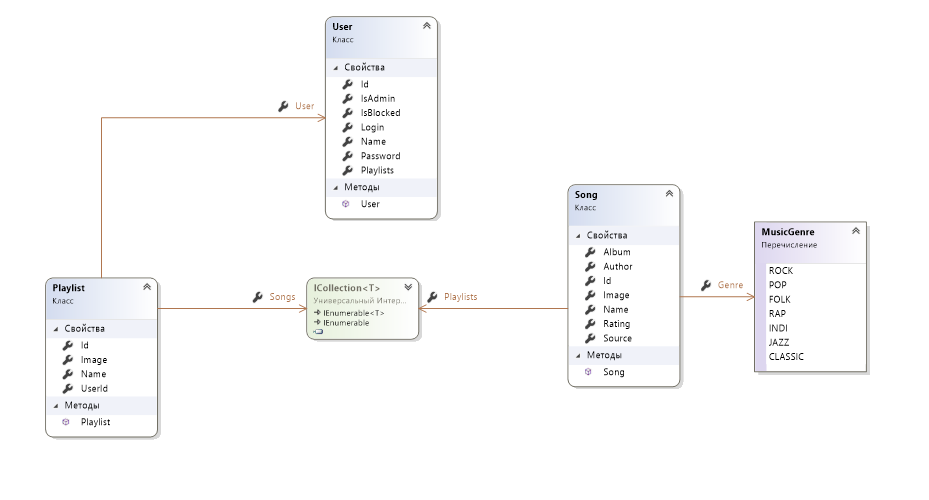


Рисунок 4.2 – UML диаграмма классов-сущностей.

Из схемы видно, что классы User, Playlist, Song и перечисление MusicGenre связаны отношением ассоциации.

### Реализация ViewModel

Слой view-model – промежуточное звено, которое связывает представление и данные. Пакетная реализация данного звена представлена в приложении (Приложение Г).

Для разгрузки view-model часть логики вынесена в controller. Пакетная структура данного звена представлена на рисунке 4.4.

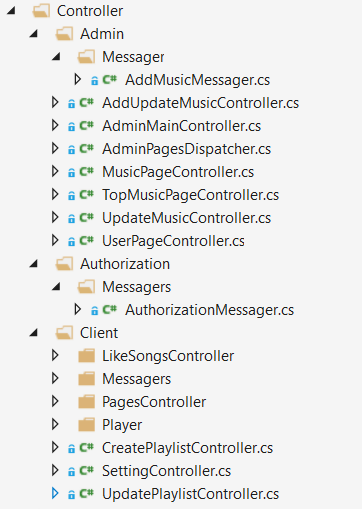


Рисунок 4.4 – Пакетная реализация controller

Диаграмма классов UML для классов view-model представлена на рисунке 4.5.

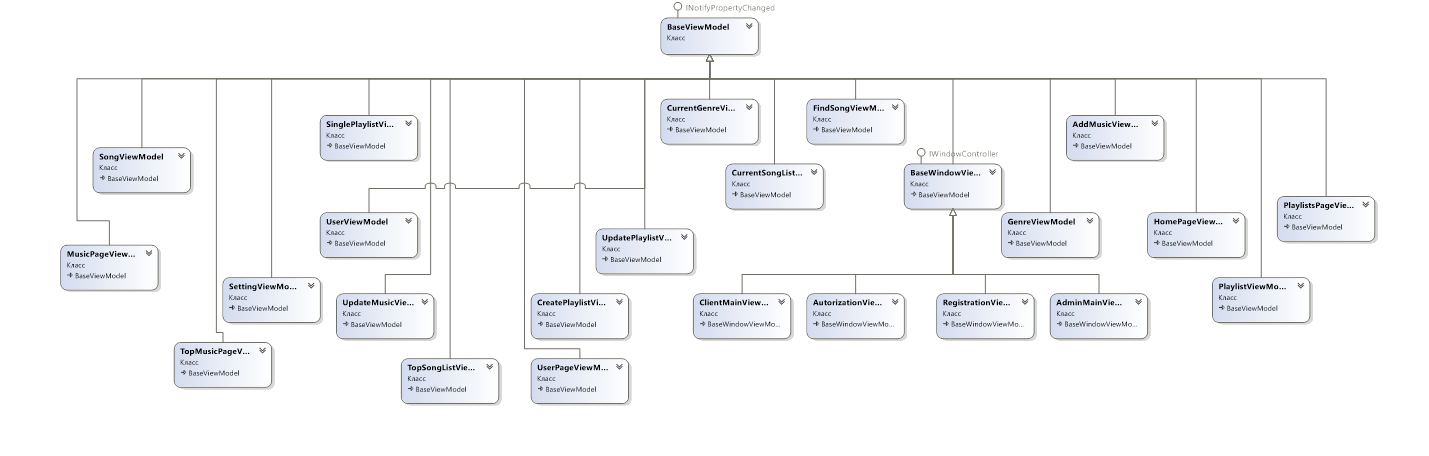


Рисунок 4.5 – UML диаграмма классов view-model

Для возможности связать данные из view-model через привязки с представлением – был реализован интерфейс INotifyPropertyChanged, в родительском классе BaseViewModel. Для его реализации, нужно объявить свойство PropertyChanged и метод OnPropertyChanged().Для того, чтобы начать отслеживание свойства, необходимо вызывать метод OnPropertyChanged(“Имя свойства”). Код реализации предоставлен в приложении (Приложение Д).

Для удобной работы с данными, используется паттерн Repository. Repository - позволяет абстрагироваться от конкретных подключений к источникам данных, с которыми работает программа, и является промежуточным звеном между классами, непосредственно взаимодействующими с данными, и остальной программой. Так как работа с данными, ведётся через один контекст, был реализован паттерн UnitOfWork. Который реализует интерфейс IDisposable – за счёт этого происходит подчистка данных после работы с контекстом. Код реализации UnitOfWork представлен в приложении (Приложение Е).

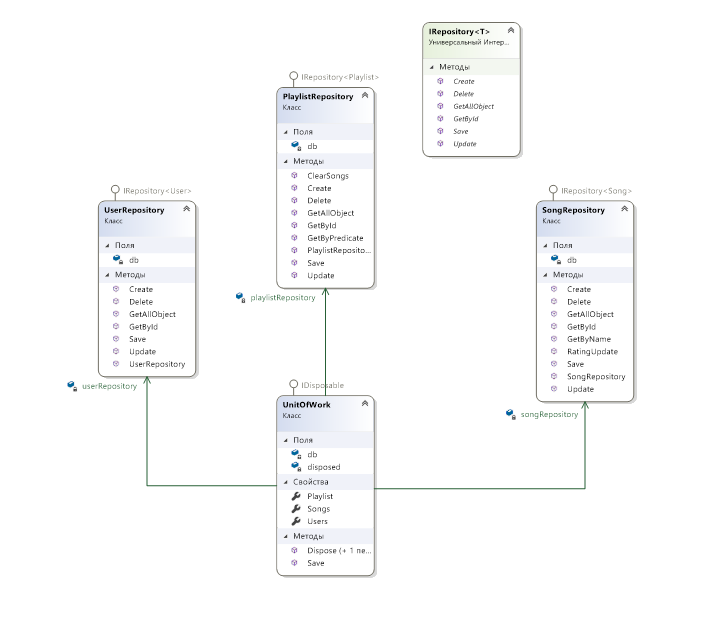


Рисунок 4.6 – Реализация паттернов Repository и UnitOfWork

### Реализация представления

Структура пакета View, в котором хранятся файлы представлений на языке разметки xaml, а также соответствующие им файлы на языке C#, показана на изображении 4.7.

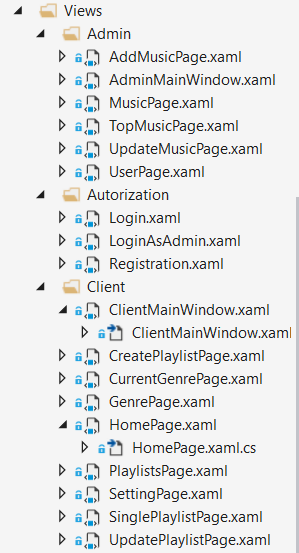


Рисунок 4.7 – Структура пакета View

В этом пакете находятся все «окна» и «страницы», которые мы видим, используя приложение. Назначение каждого файла описано в таблице ниже (таблица 4.1)

Таблица 4.1 – «Описание файлов View»

|  |  |
| --- | --- |
| **Название** | **Краткое описание** |
| AddMusicPage.xaml | Страница добавления треков |
| AdminMainWindow.xaml | Главное окно администратора |
| MusicPage.xaml | Страница со списком треков |
| TopMusicPage.xaml | Страница с плэйлистом топ-треков |
| UpdateMusicPage.xaml | Страница обновления информации о треке |
| UserPage.xaml | Страница со списком пользователей |
| Login.xaml | Окно входа пользователя |
| LoginAsAdmin.xaml | Окно входа администратора |
| Registration.xaml | Окно регистрации |
| ClientMainWindow.xaml | Главное окно пользователя |

Продолжение таблицы 4.1

|  |  |
| --- | --- |
| **Название** | **Краткое описание** |
| CreatePlaylistPage.xaml | Страница создания плэйлиста |
| CurrentGenrePage.xaml | Страница со списком треков определённого жанра |
| GenrePage.xaml | Страница со списком жанров |
| HomePage.xaml | Домашняя страница со списком всех треков |
| PlaylistsPage.xaml | Страница со списком плэйлистов текущего пользователя |
| SettingPage.xaml | Страница настроек |
| SinglePlaylistPage.xaml | Страница определённого плэйлиста |
| UpdatePlaylistPage.xaml | Страница редактирования плэйлиста |

# Тестирование, проверка работоспособности и анализ полученых результатов

При тестировании данного приложения, были применены сценарии, которые могли бы привести к ошибке. В этой главе мы рассмотрим некоторые такие сценарии и посмотрим на их обработку.

В момент регистрации, возможна такая ситуация, в которой пользователь ничего не ввёл. Обработка данного сценария приведена на рисунке 5.1.

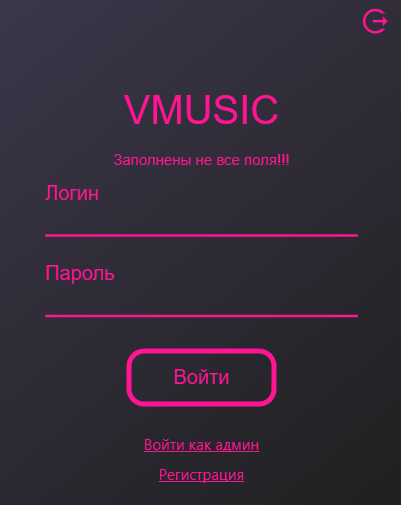


Рисунок 5.1 – Ошибка «пустого» ввода

Обработка ситуации ввода не корректных или несуществующих данных, представлена на изображении 5.2.

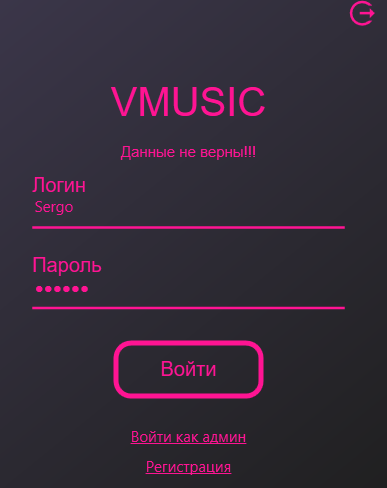


Рисунок 5.2 – Ввод неверных данных

Попытка создать пользователя с повторяющимися данными также обрабатывается в приложении, демонстрация на рисунке 5.3.

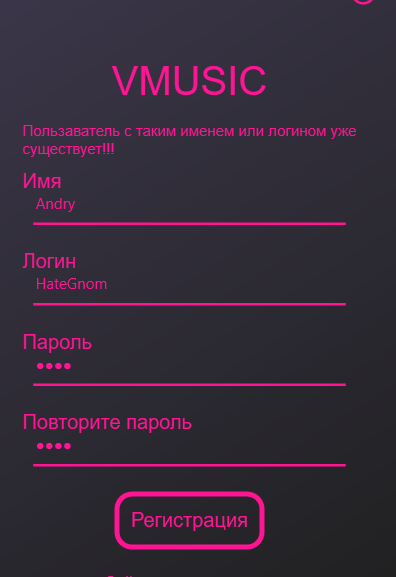


Рисунок 5.3 – Регистрация пользователя с повторяющимся логином

В приложении минимальный размер пароля – 4 символа, собственно ситуация ввода пароля меньшей длины проходит валидацию. Демонстрация на рисунке 5.4.

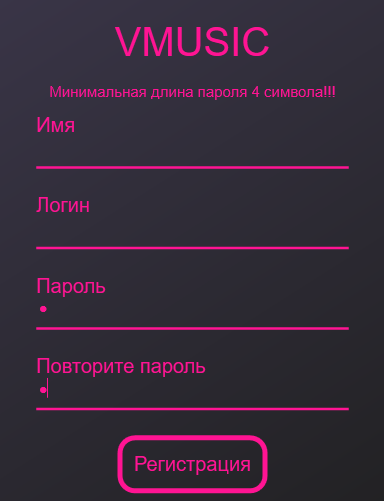


Рисунок 5.4 – Ввод слишком маленького пароля

Также было проведено тестирование работы валидации внутри приложения. Например: попытка загрузить изображение при создание плэйлиста размером больше 1МБ.

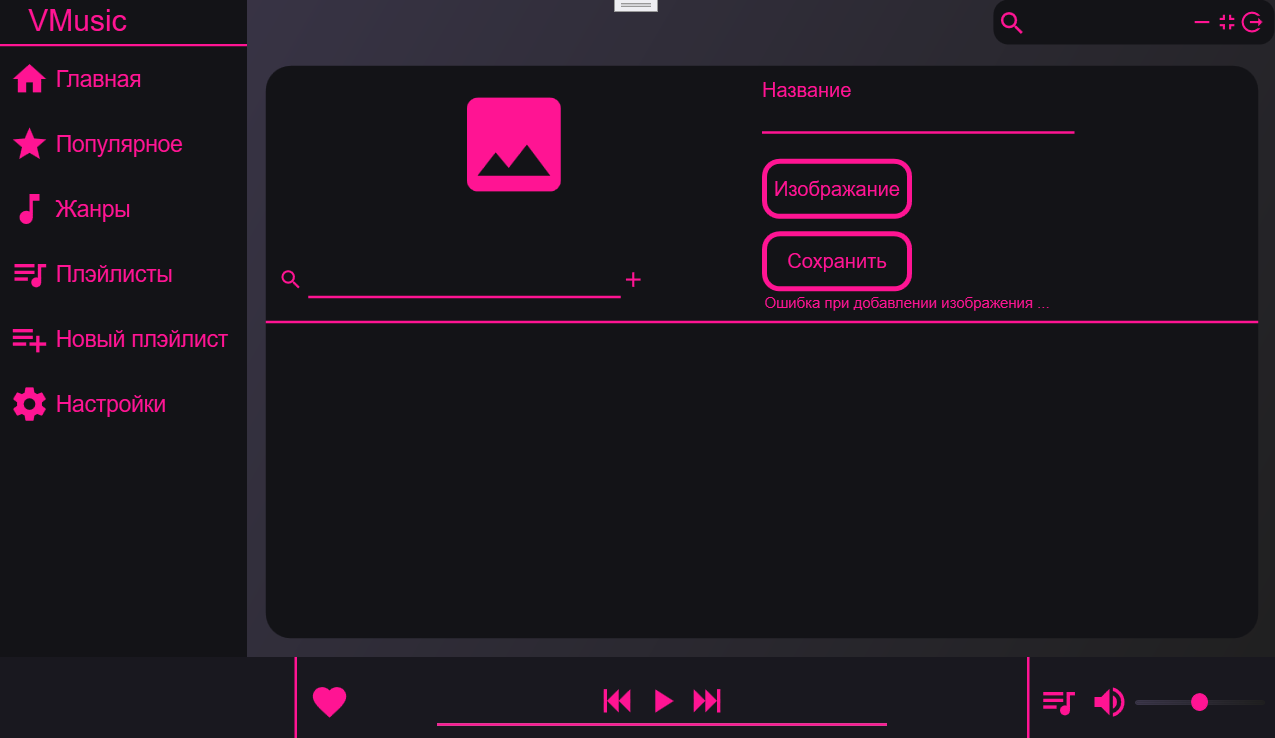


Рисунок 5.5 – Добавление изображения большого размера

Также возможно ситуация добавления трека при пустых полях формы, пример обработки ситуации предоставлен на рисунке 5.6.

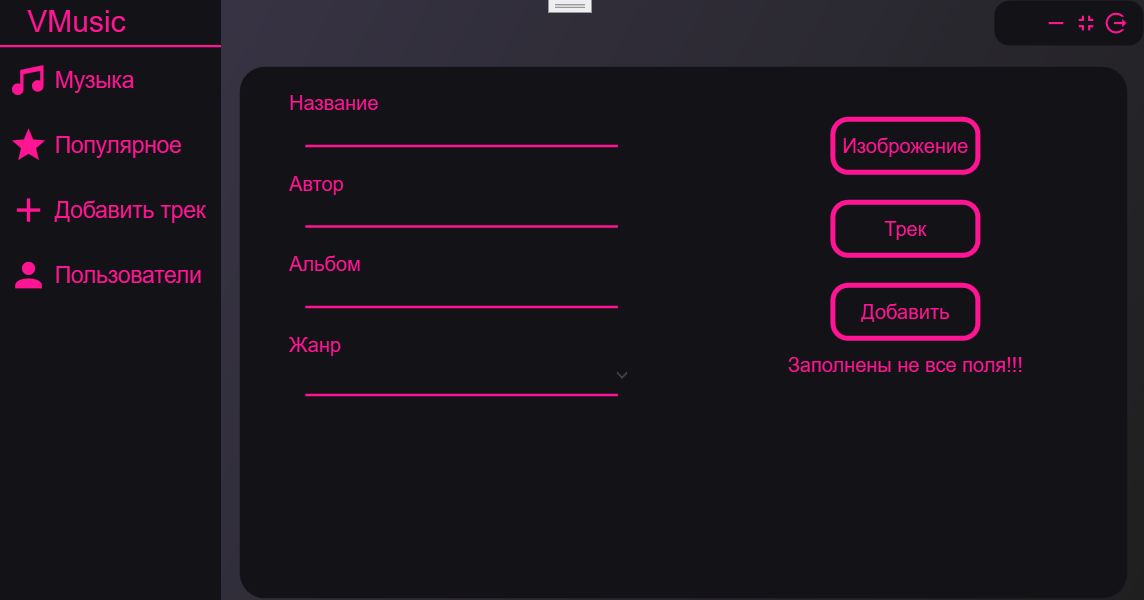


Рисунок 5.6 – Пустая форма при добавлении трека

При добавлении треков возможна ситуация, когда он уже добавлен. Демонстрация обработки на рисунке 5.7.

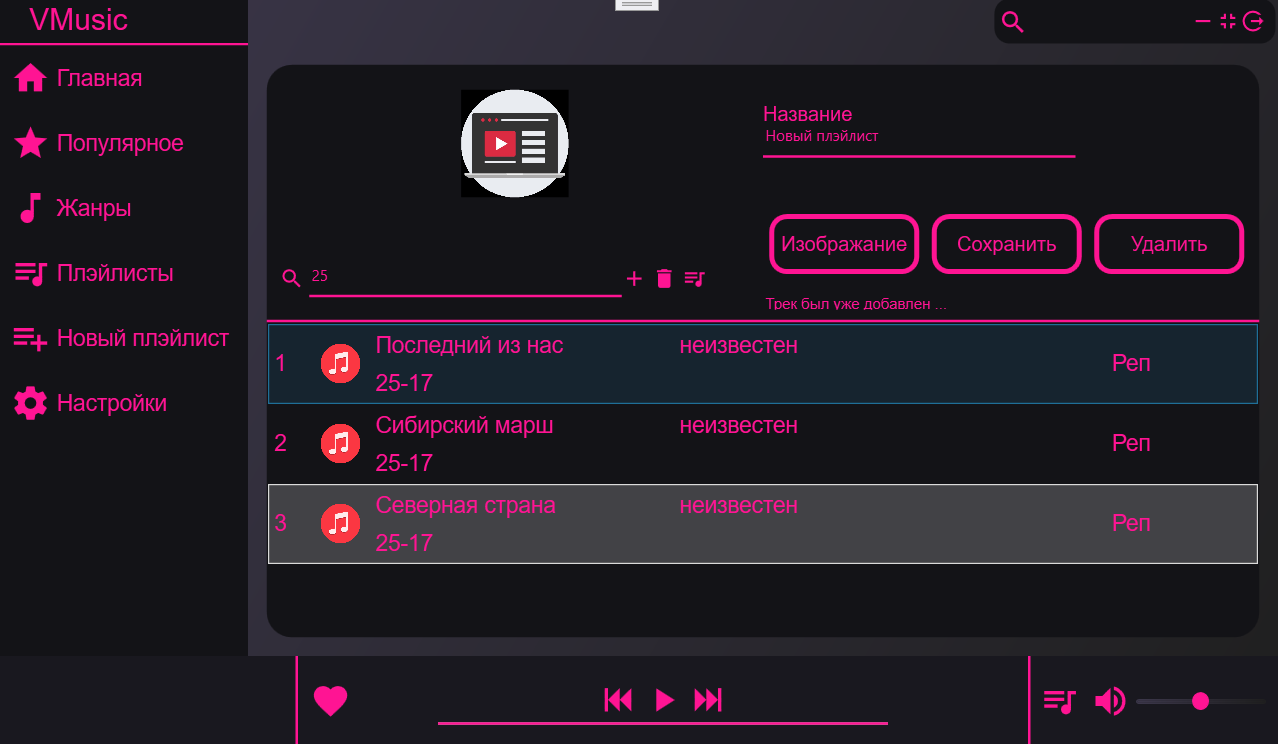


Рисунок 5.7 – Добавление повторяющегося трека в плэйлист

При смене пароля необходимо повторить новый пароль. Обработка ситуации на рисунке 5.8.

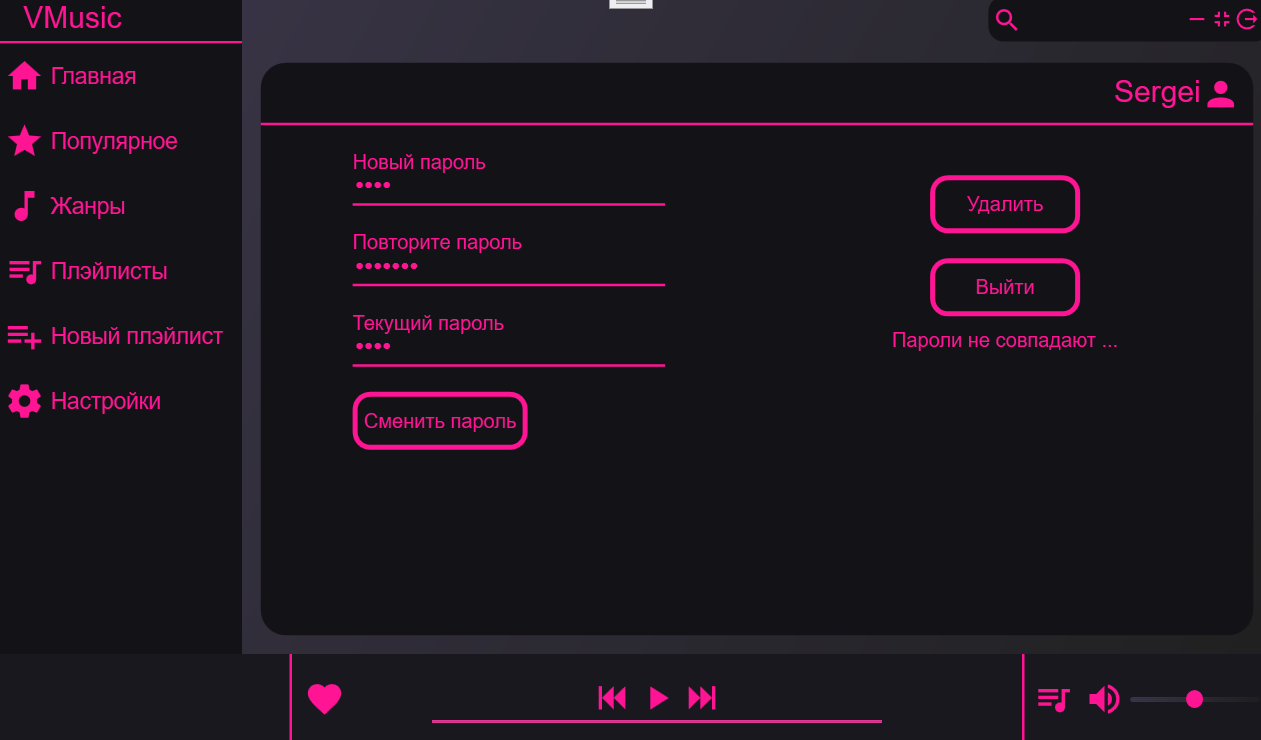


Рисунок 5.8 – Несовпадение паролей

На данном этапе были выполнены тесты на проверку работоспособности приложения, а именно на проверку валидации.

# Руководство по использованию

### Руководство пользователя

После запуска приложения открывается окно входа, здесь вы можете войти как пользователь. Так же можно перейти на окно входа для админа и если вы обладаете должными данными войти как администратор, либо перейти на страницу регистрации. После регистрации вас автоматически перебросит на окно входа пользователя.

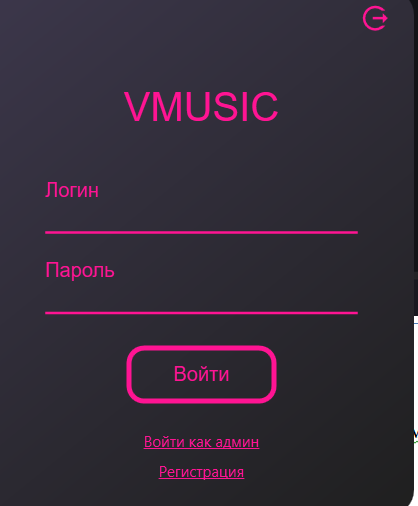


Рисунок 6.1 – Окно входа

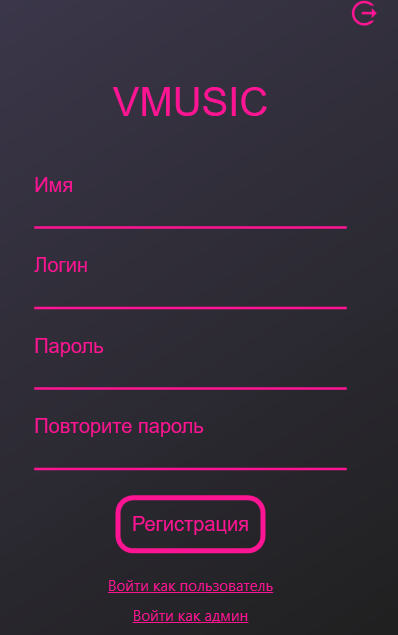


Рисунок 6.2 – Окно регистрации

После успешного входа вам откроется главное окно пользователя. Оно содержит боковое меню навигации, строку поиска, мини-проигрыватель и область отображения данных.

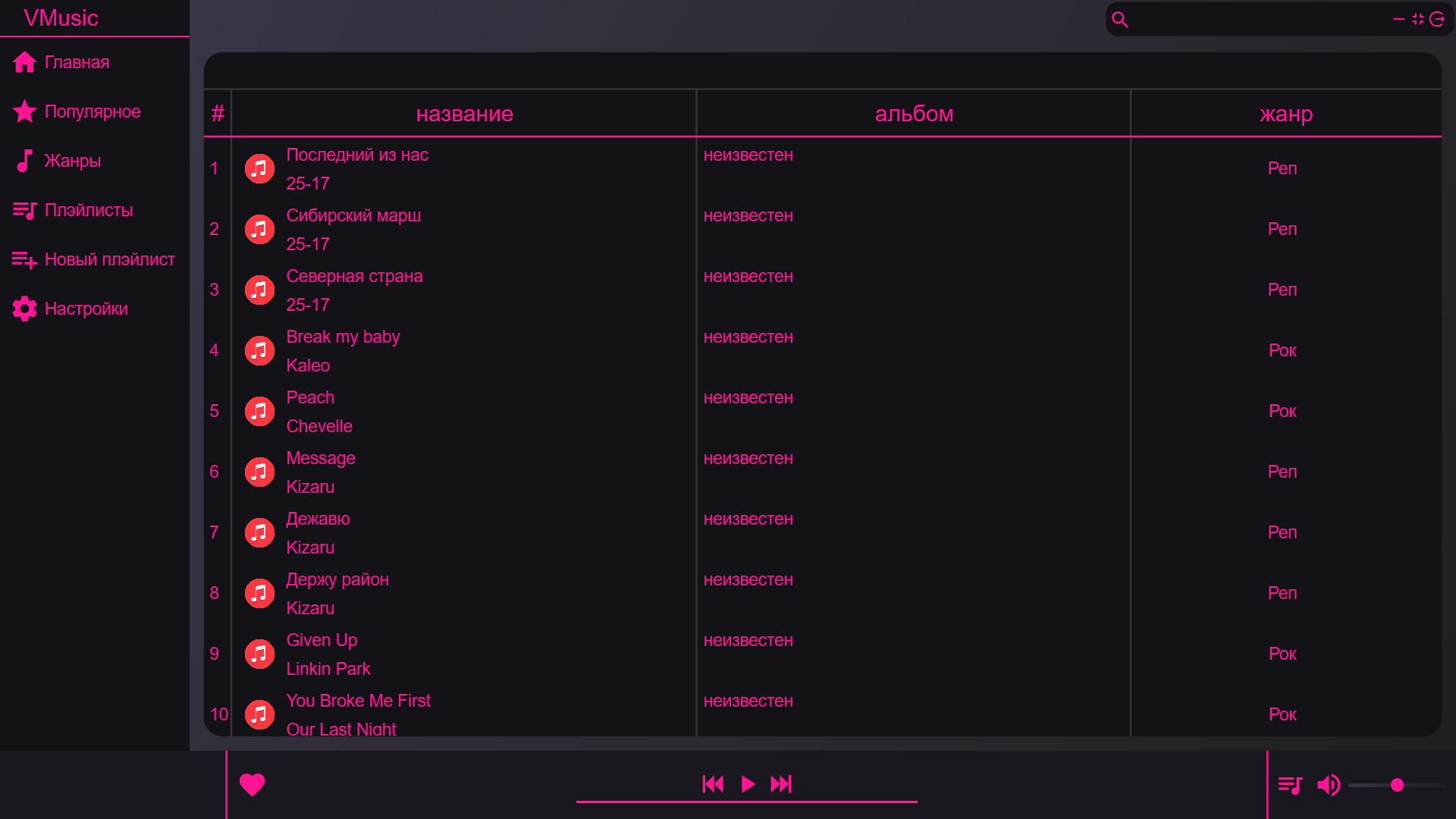


Рисунок 6.3 – Главное окно пользователя

Изначально будет открыта главная страница со списком всех треков. Для проигрывания трека необходимо щёлкнуть по выбранному из списка левой кнопкой мыши.

Снизу окна размещён мини-проигрыватель с помощью которого вы можете управлять проигрыванием треков:

* с помощью слайдера и кнопки регулировать громкость;
* ставить трек на паузу и воспроизводить обратно;
* переключаться на следующий или предыдущий треки;
* просмотреть текущий проигрываемый список треков;
* с помощью кнопки в виде сердца добавлять трек в избранные.

При переходе на страницу «Популярное» будет отображён список популярных треков.

При переходе на страницу «Жанры» будет отображён список жанров. И нажатием по выбранному можно будет перейти к списку треков конкретного жанра.

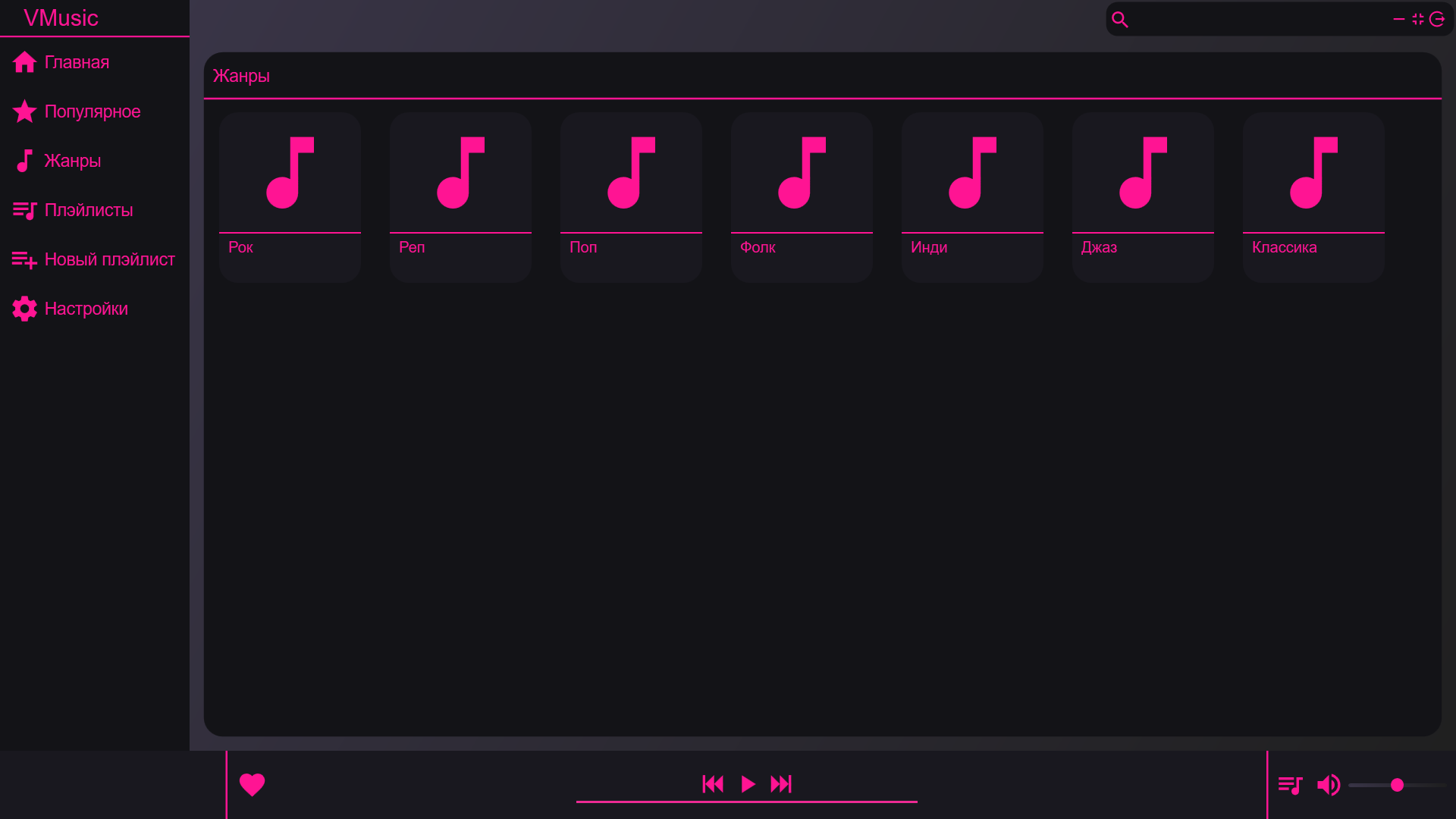


Рисунок 6.4 – Список жанров

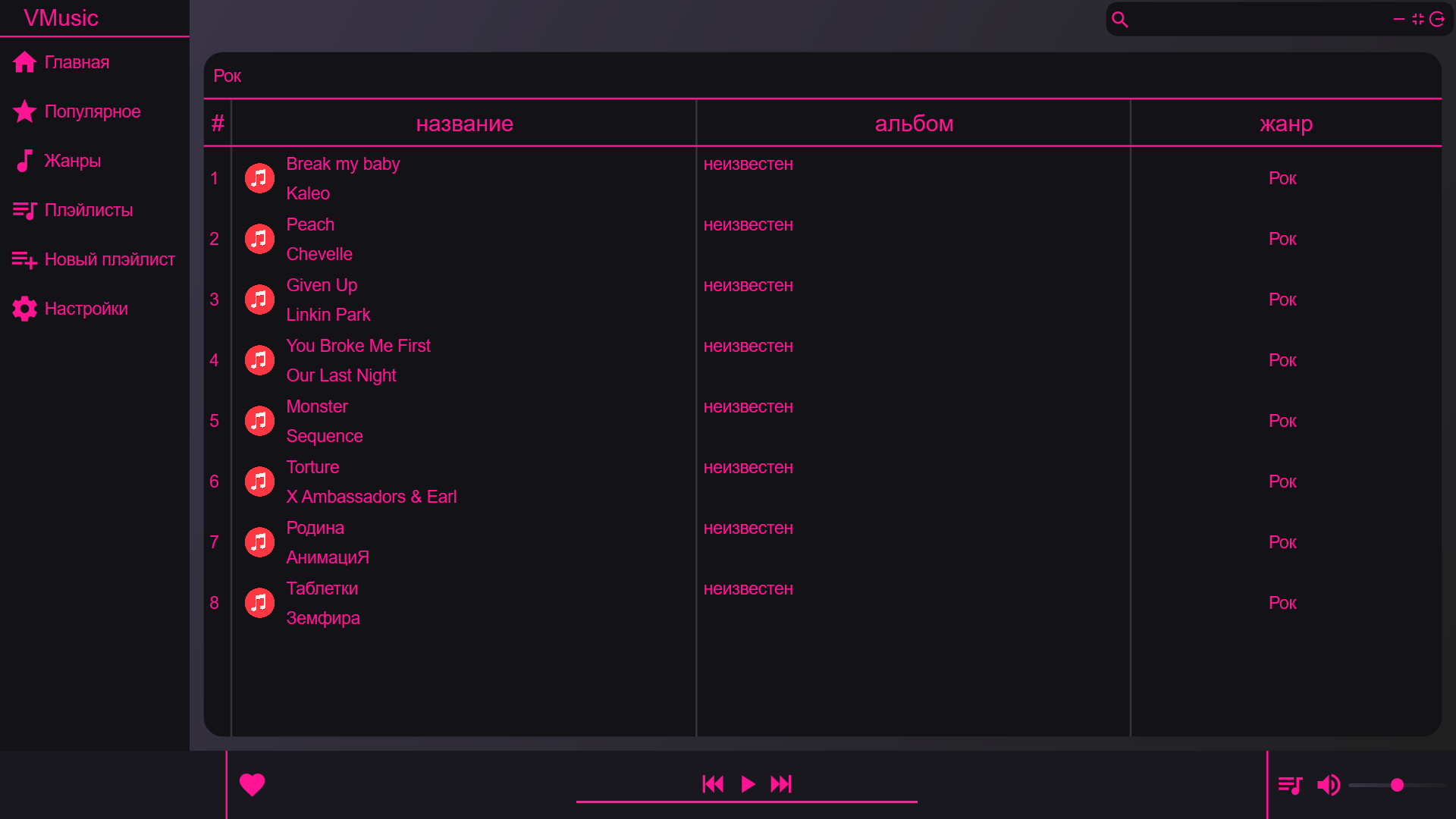


Рисунок 6.5 – Список треков определённого жанра

При переходе на страницу «Плэйлисты» можно будет увидеть список всех плэйлистов пользователя, а также перейти к конкретному.

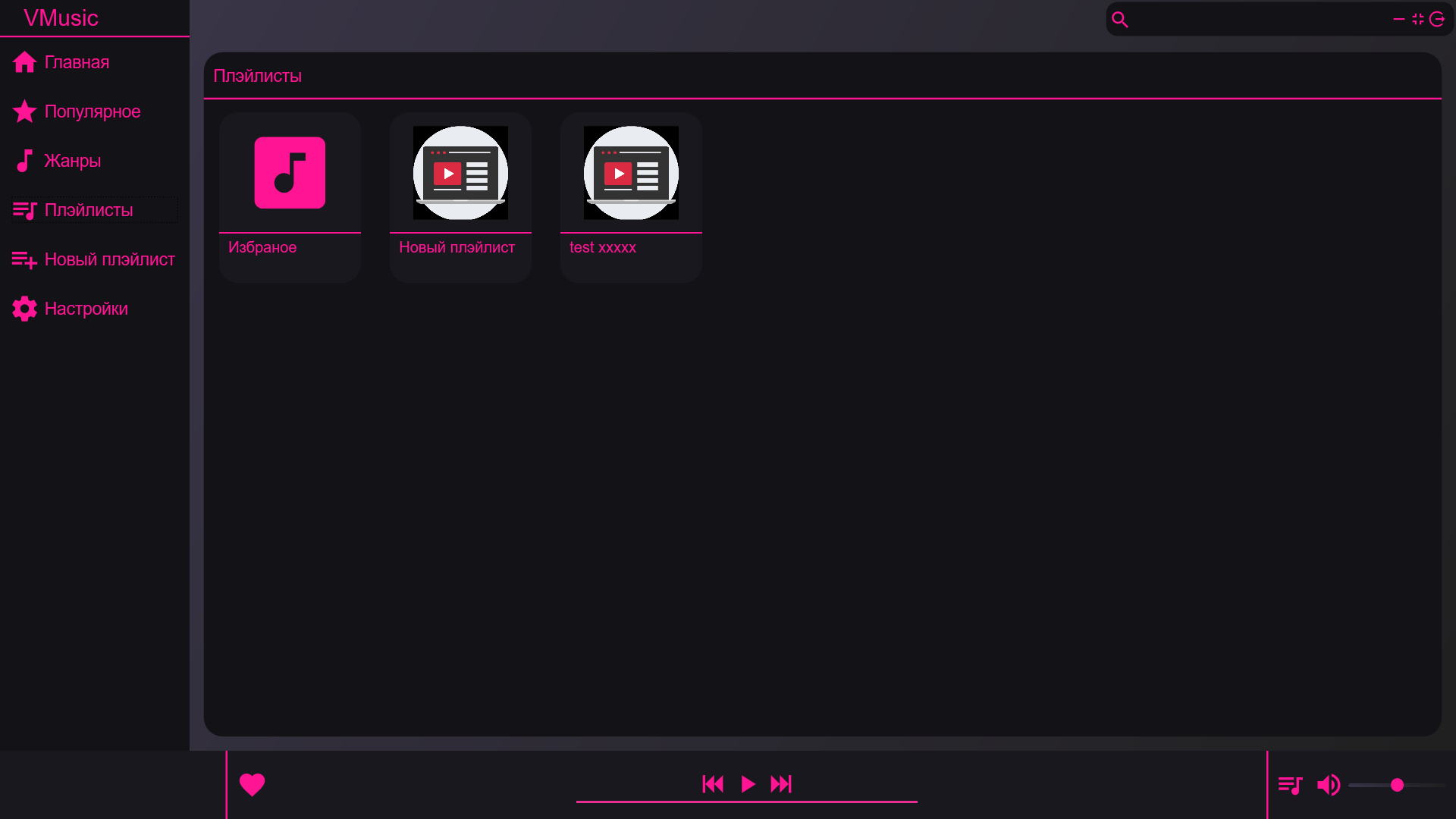


Рисунок 6.6 – Страница «Плэйлисты»

При переходе к конкретному плэйлисту можно будет начать его прослушивание, щелчком кнопки либо выбором трека из списка, или перейти к его редактированию.

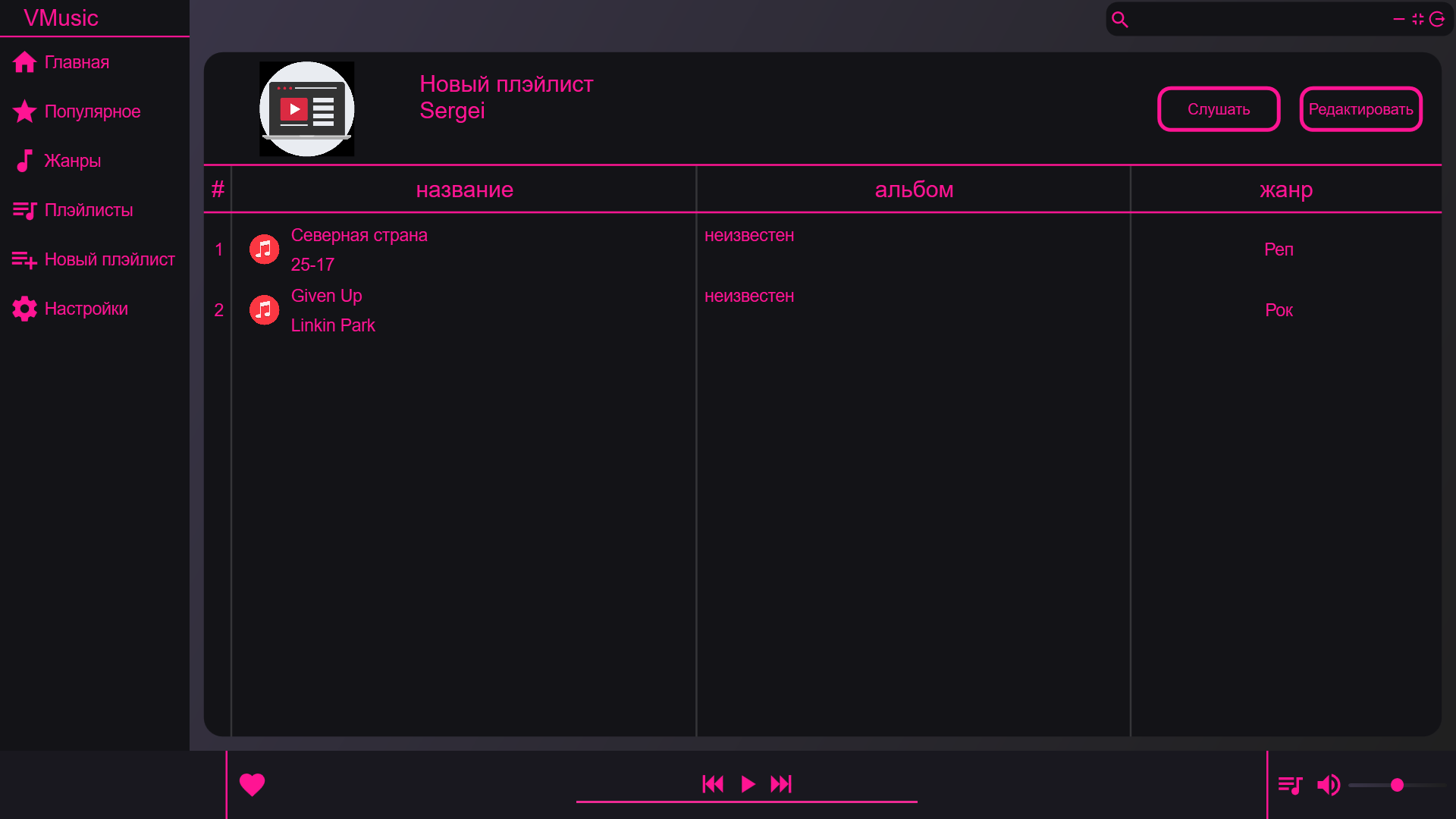


Рисунок 6.7 – Страница конкретного плэйлиста

На странице редактирования можно изменить название плэйлиста, сменить картинку, удалить его или изменять коллекцию треков с помощью поисковой строки на странице и вспомогательных кнопок.

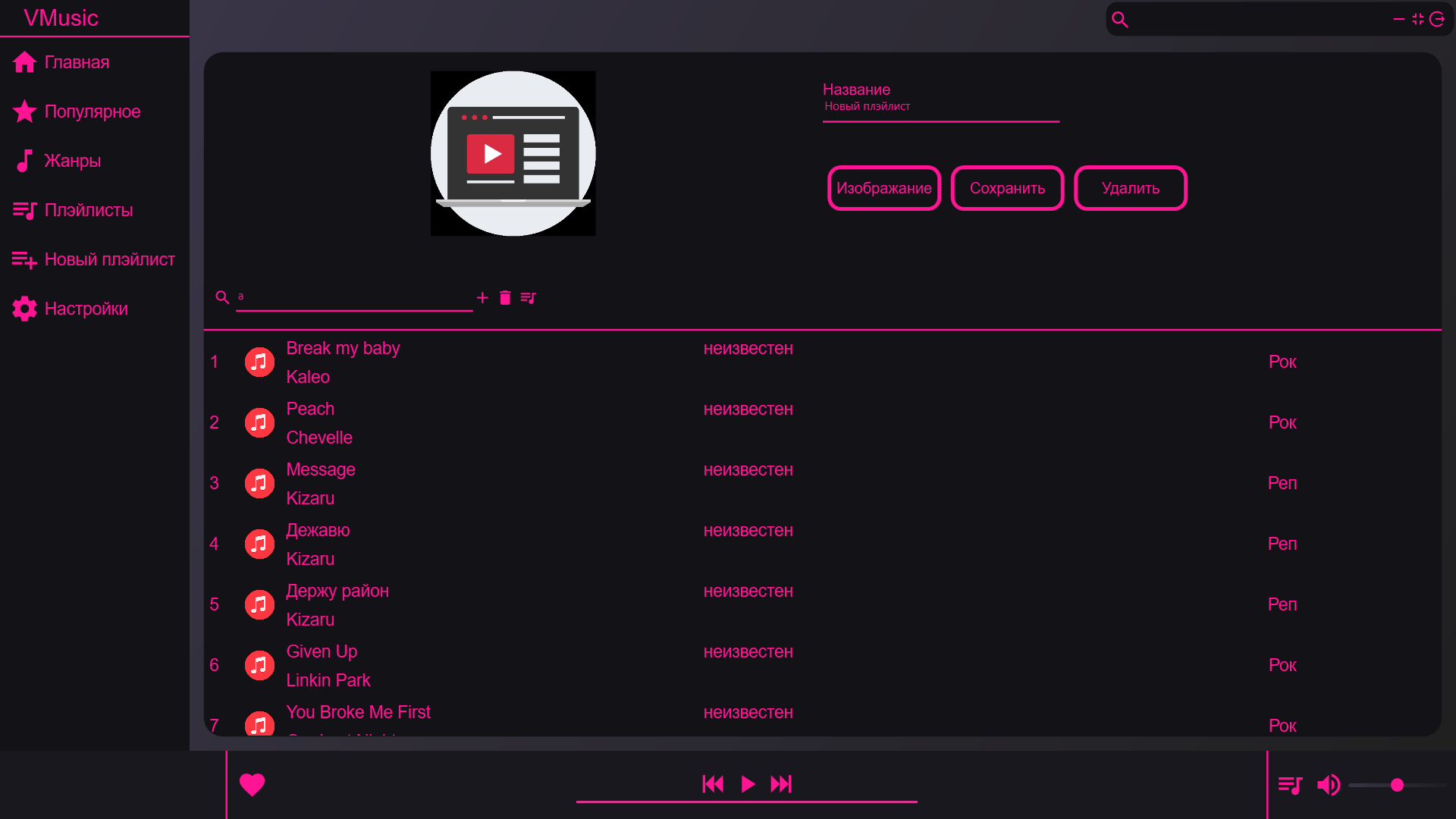


Рисунок 6.8 – Редактирование плэйлиста

Страница «Новый плэйлист» предоставляет возможность создания плэйлиста. В текстовое поле вводится название, есть кнопка для добавления изображения, поисковая строка для поиска треков и кнопка для их добавления в плэйлист.

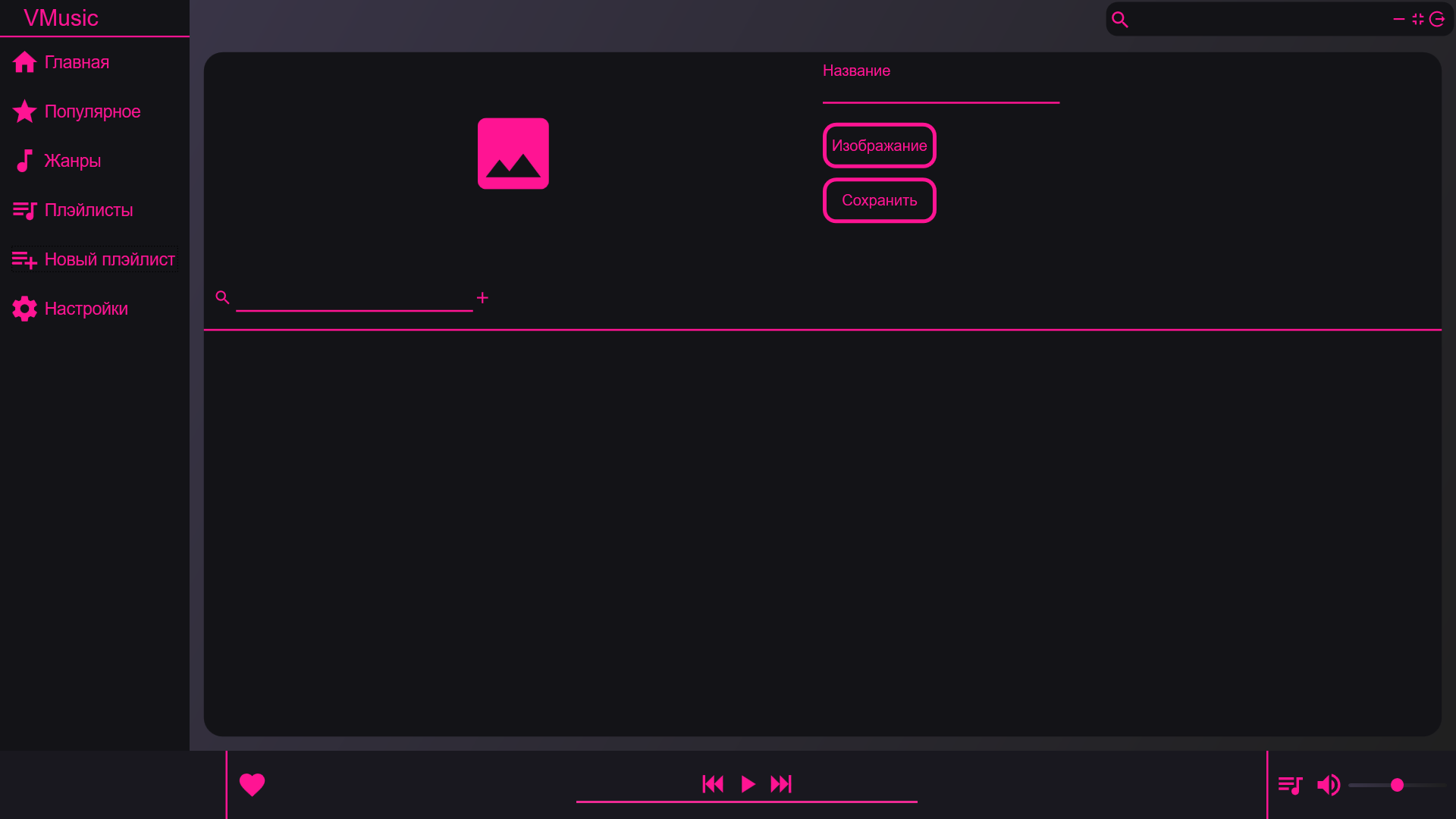


Рисунок 6.9 –Новый плэйлист

Страница настройки содержит поля и кнопку для смены пароля, кнопку для удаления аккаунта и кнопку для выхода из него.

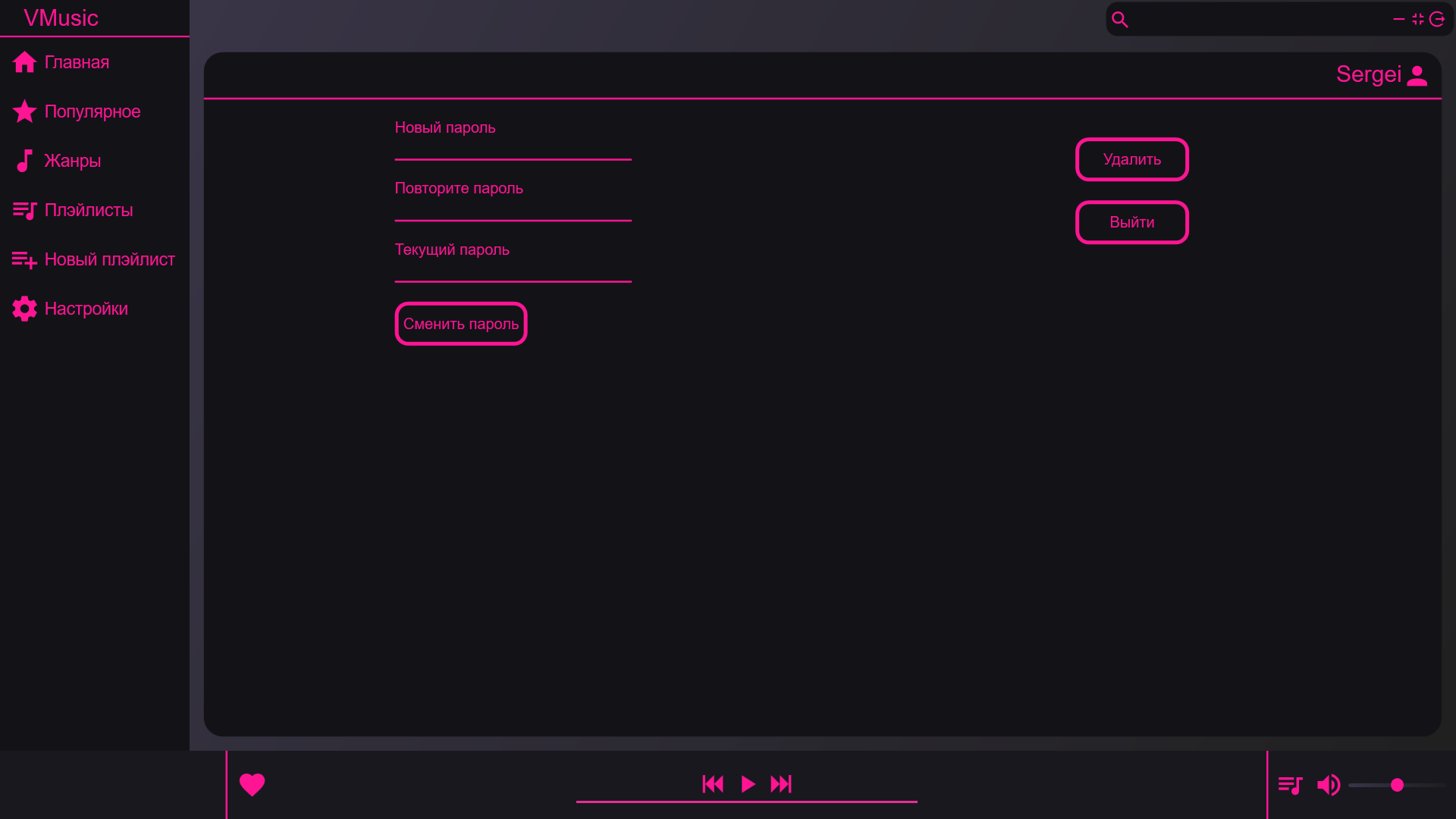


Рисунок 6.10 – Настройки

Вверху главного окна содержаться стандартные кнопки управления окном, а также строка поиска. Для поиска необходимо ввести данные и нажать кнопку и в центральной области будет отображён результат.

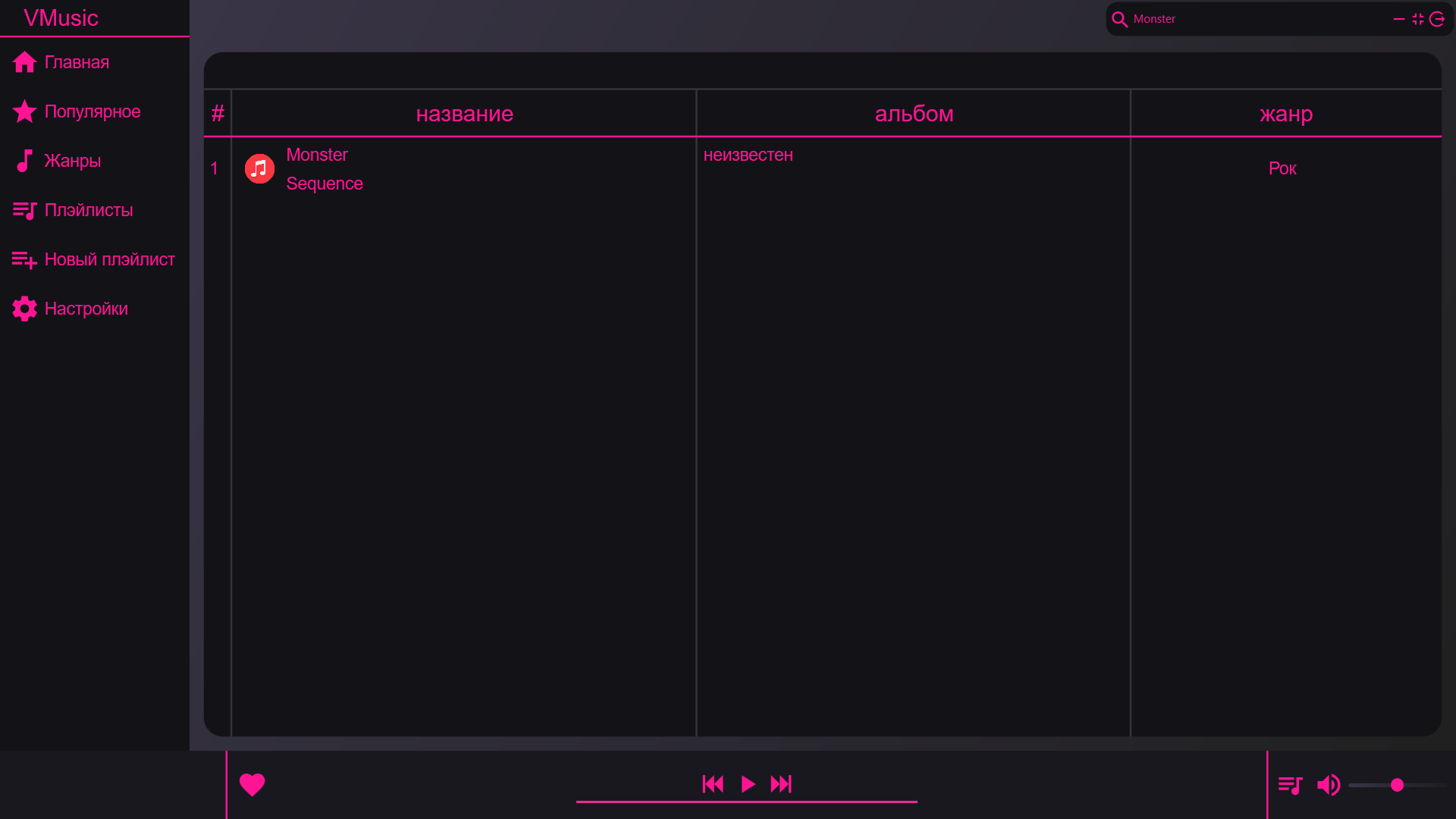


Рисунок 6.11 – Поиск трека

### Руководство администратора

После перехода в окно авторизации администратора предоставляется ввести данные и после проверки их корректности открывается главное окно администратора. В центре окно находится основная область с отображаемыми данными, а слева меню для навигации по страницам.

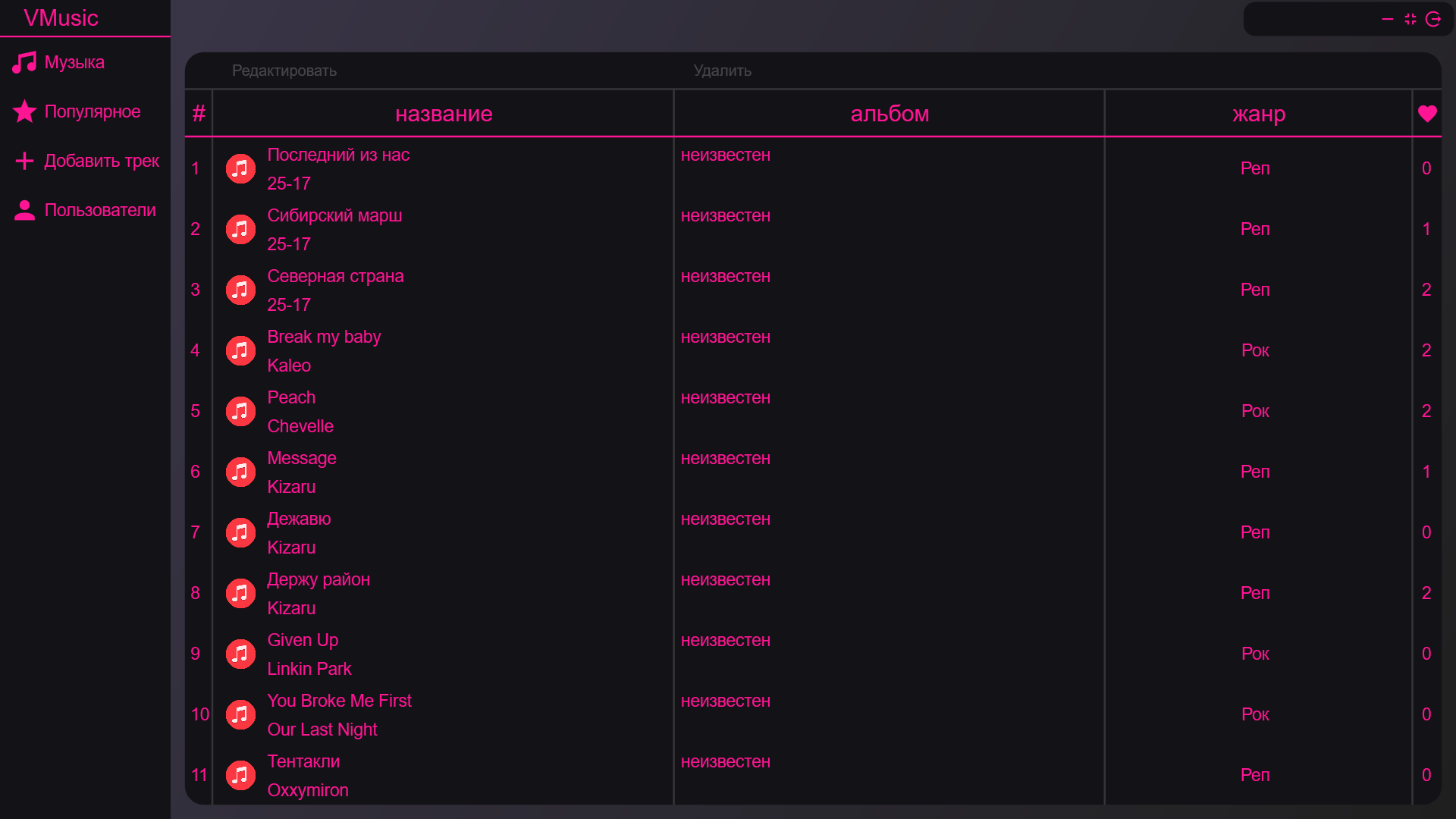


Рисунок 6.12 – Главное окно администратора

Изначально открывается страница «Музыка» на которой предоставлен список всех треков. Над списком размещены кнопки редактирования и удаления. Для удаления или редактирования необходимо выбрать трек из списка, а затем нажать кнопку. При нажатии кнопки «Редактировать» произойдёт переход на страницу редактирования трека.

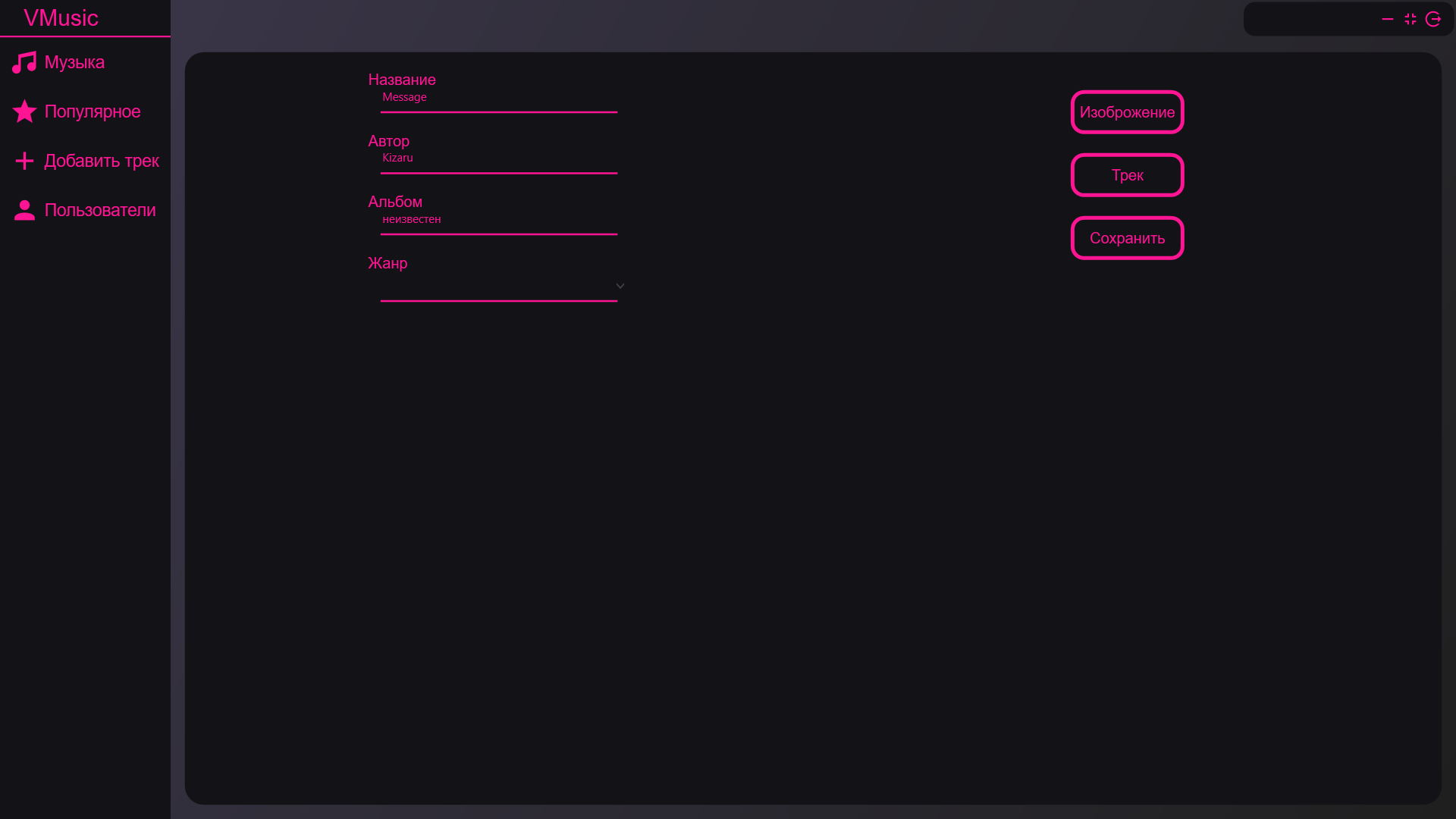


Рисунок 6.13 – Страница редактирования трека

На ней есть возможность изменить информацию о треке, сменить изображение, заменить ссылку на аудио-файл. После внесённых изменений необходимо нажать кнопку «Сохранить».

При переходе на страницу «Популярное» отображается список 10 популярных треков. Для его обновления над списком размещена кнопка «Обновить». После её нажатия будет проведена сортировка треков по рейтингу и выбор 10 популярнейших из них.

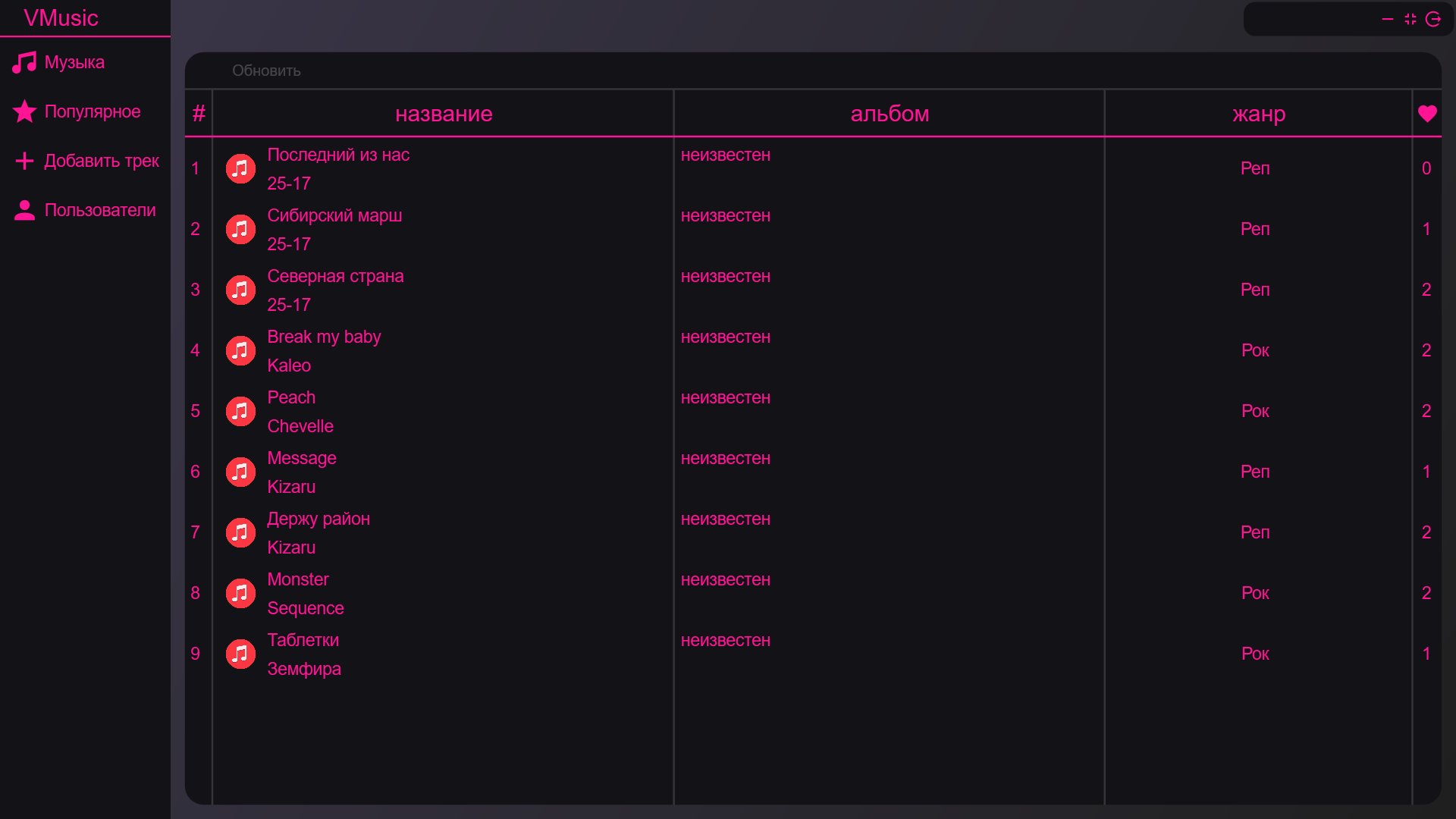


Рисунок 6.14 – Страница популярных треков

При переходе на страницу «Добавить трек» предоставляется возможность заполнить поля данными, выбрать изображение-заставку и ссылку на аудио-файл, для чего есть соответствующие кнопки. После заполнения всех данных нажатием кнопки «Добавить» информация сохраняется в БД.

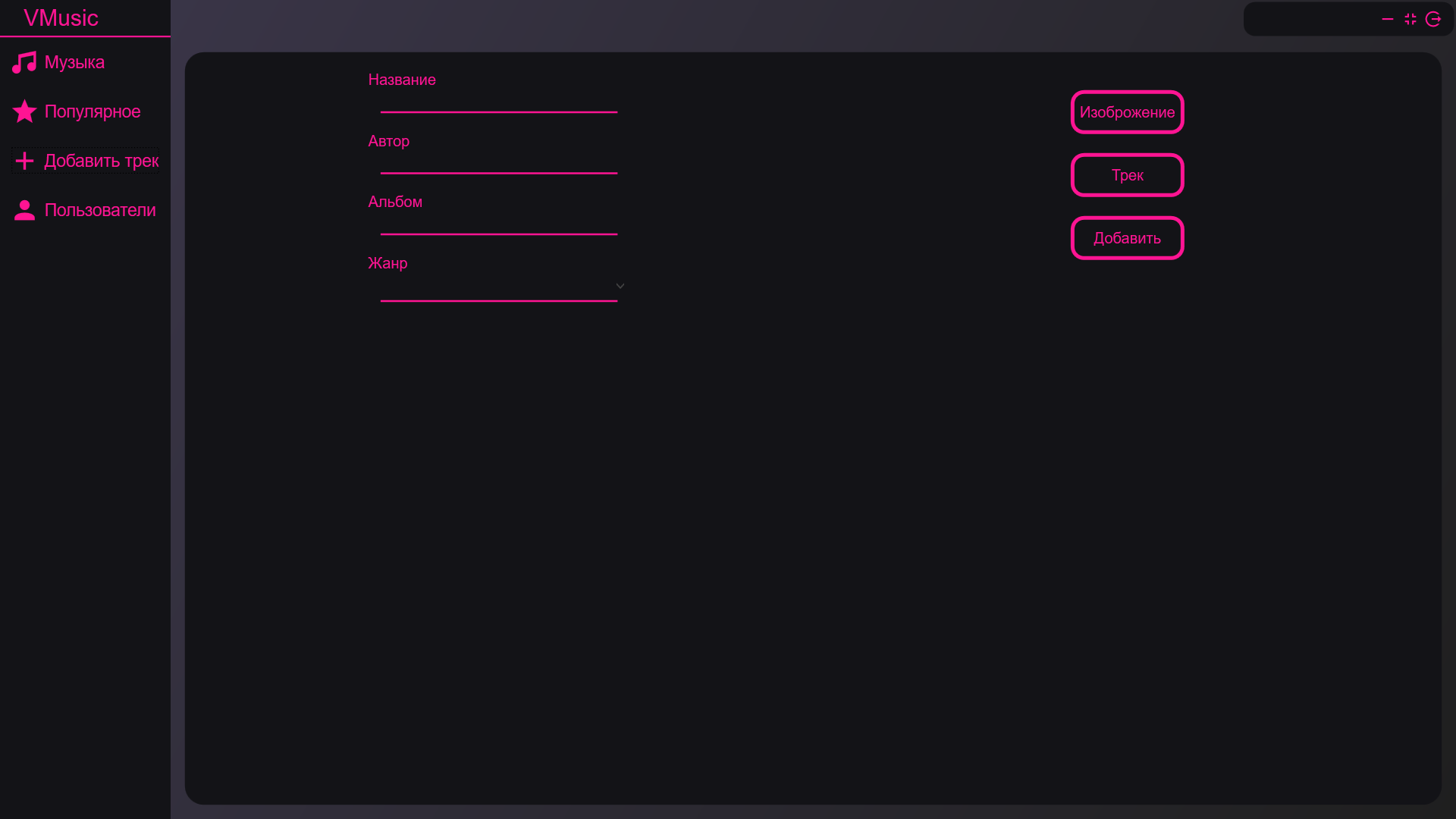


Рисунок 6.14 – Добавление трека

Страница «Пользователи» содержит список всех пользователей, а также кнопки для блокировки, удаления или передачи прав администратора другому пользователю. Для выполнения какого-либо действия необходимо выбрать пользователя из списка, а затем нажать соответствующую кнопку.

Сверху главного окна содержатся стандартные кнопки управления окном.

# ЗАКЛЮЧЕНИЕ

В данном курсовом проекте было разработано программное средство «Музыкальный сервис VMusic» при помощи языка программирования C#, API-интерфейса Windows Presentation Foundation, технологии Entity Framework и базы данных Microsoft SQL Server. При выполнении курсового проекта использовались принципы и приемы ООП.

Программное средство предоставляет пользователю следующие функциональные возможности:

* прослушивание треков;
* поиск треков;
* создание плэйлистов;
* прослушивание плэйлистов;
* редактирование плэйлистов;
* редактирование приватной информации пользователя(смена пароля);
* возможность оценки треков;
* добавление треков в избранные.

Возможности администратора:

* добавление и удаление треков;
* редактирование информации о треках;
* обновление коллекции популярных треков;
* удаление и блокировка пользователей;
* передача прав администратора.

Проектирование осуществлялось по паттерну проектирования MVVM.

Для удобной работы с данными, был реализован паттерн Repository и UnitOfWork.

Приложение хорошо работает с базой данных. В ходе тестирования, не было выявлено нарушений в работе приложения с базой данных.

Приложение было успешно протестировано, что означает пригодность этого приложения для своей цели.

Для хранения исходного кода и удобства контроля версий проекта использовался GitHub. Ссылка на проект: (https://github.com/SergeiWal/VMusic).

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

[1]. Пацей, Н.В. Курс лекций по языку программирования С# / Н.В. Пацей. – Минск: БГТУ, 2021. – 175 с.

[2]. Пацей, Н.В. Технология разработки программного обеспечения / Н.В. Пацей. – Минск: БГТУ, 2016. – 129 с. [3]. MSDN сеть разработчиков в Microsoft [Электронный ресурс] / Режим доступа: http://msdn.microsoft.com/library/

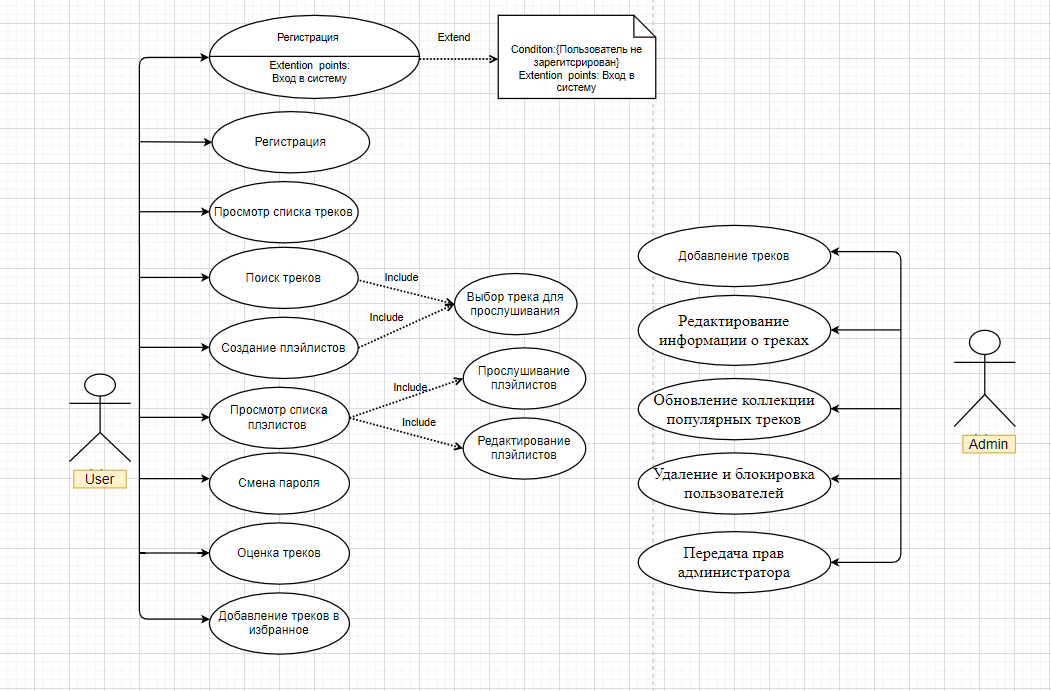
[4]. METANIT.COM Сайт о программировании [Электронный ресурс] / Режим доступа: https://metanit.com

[5]. ProfessorWeb .NET & Web Programming [Электронный ресурс] / Режим доступа: https://professorweb.ru

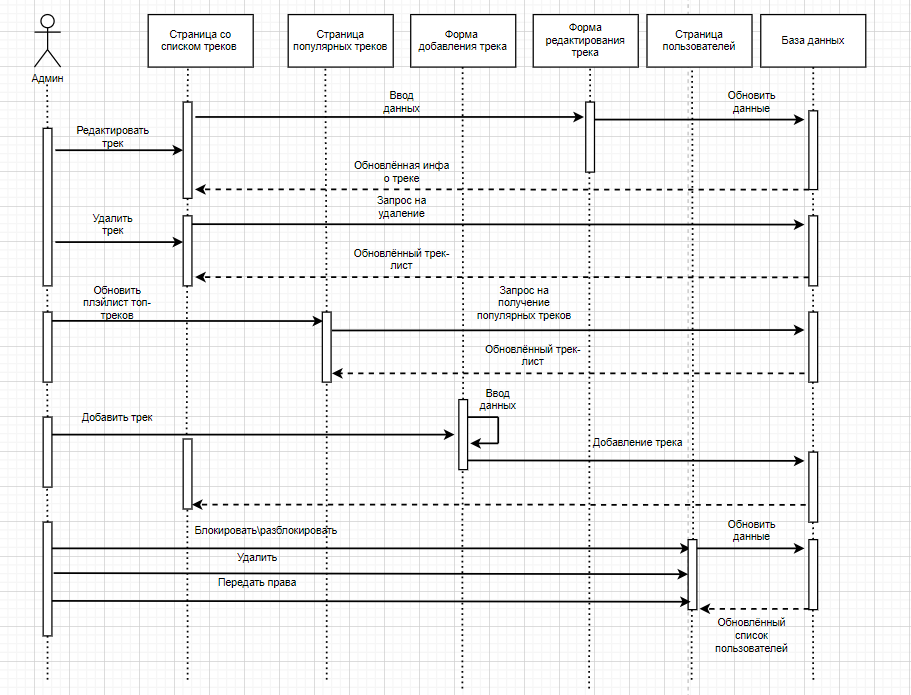
[6]. StackOverflow [Электронный ресурс] / Режим доступа: <https://stackoverflow.com>

[7]. Хабр [Электронный ресурс] / Режим доступа: <https://habr.com>.

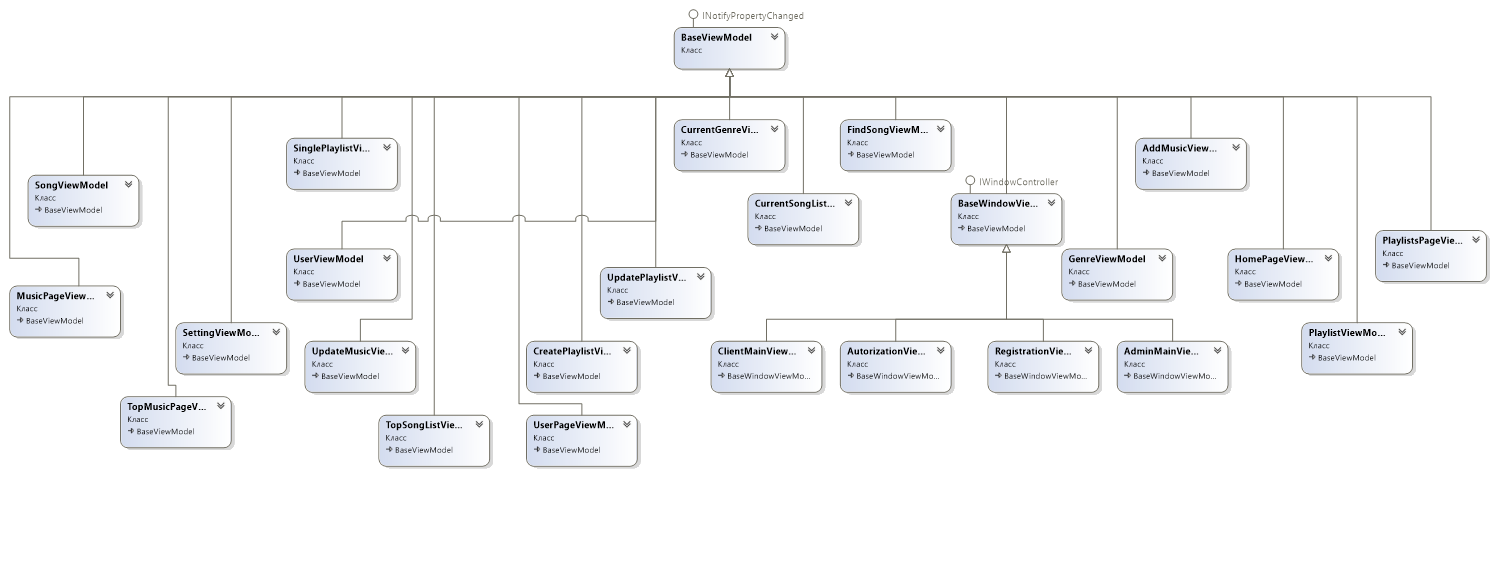
# Приложение А

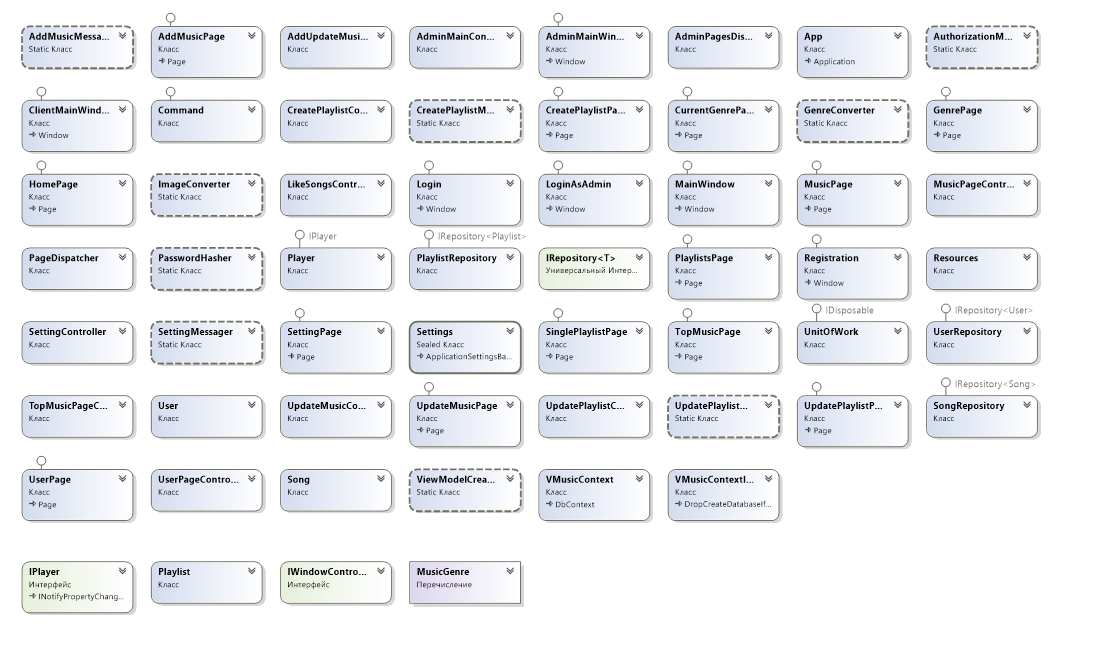


# Приложение Б

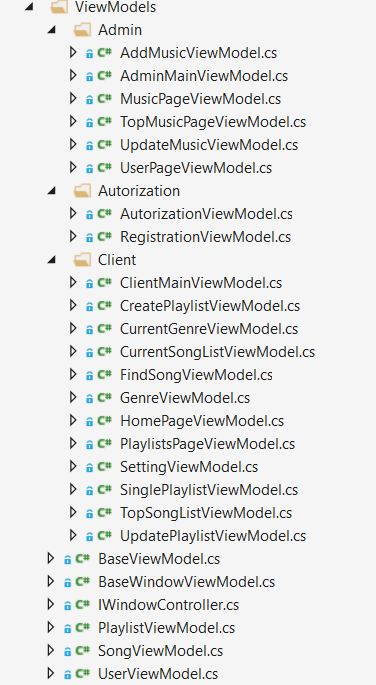


# Приложение В

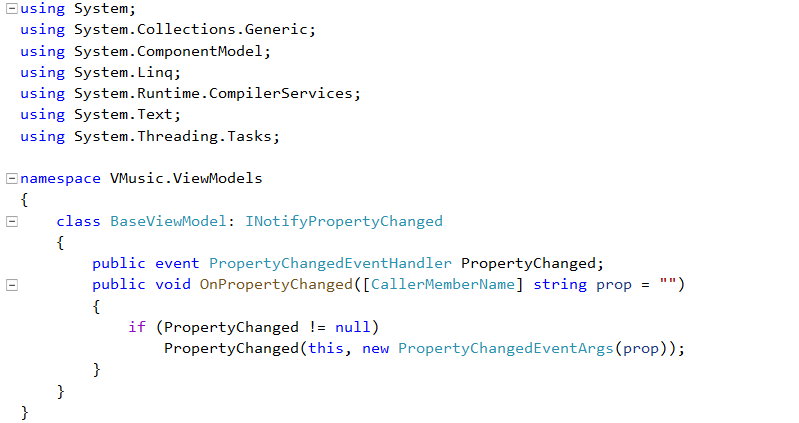




# Приложение Г



# Приложение Д



# Приложение Е

