

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
«Информатика»
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
А. С. Кузнецов
подпись инициалы, фамилия
« » 2019 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.04 «Программная инженерия»
код — наименование направления

Разработка веб-сервиса для миграции музыкальных плейлистов из социальной сети
«ВКонтакте» в потоковый музыкальный сервис «Deezer»
тема

Руководитель	_____	<u>доцент, канд. техн. наук</u>	<u>А. В. Хныкин</u>
	подпись, дата	должность, ученая степень	инициалы, фамилия
Выпускник	_____		<u>А. К. Мангиров</u>
	подпись, дата		инициалы, фамилия
Нормоконтролер	_____	<u>доцент, канд. техн. наук</u>	<u>О. А. Антамошкин</u>
	подпись, дата	должность, ученая степень	инициалы, фамилия

Красноярск 2019

РЕФЕРАТ

Выпускная квалификационная работа по теме: «Разработка веб-сервиса для миграции музыкальных плейлистов из социальной сети «ВКонтакте» в потоковый музыкальный сервис «Deezer»» содержит 39 страниц текстового документа, 29 иллюстраций, 1 таблицу, 18 использованных источников.

РАЗРАБОТКА ВЕБ-СЕРВИСА, АВТОРИЗАЦИЯ, КЛИЕНТ-СЕРВЕРНАЯ АРХИТЕКТУРА, PYTHON, JAVASCRIPT, FLASK, БАЗА ДАННЫХ, ТОКЕН ДОСТУПА, ЗАПРОСЫ К API.

Целью работы является написание веб-сервиса для переноса аудиотеки пользователя «ВКонтакте» в музыкальный сервис «Deezer».

Для выполнения были поставлены задачи:

- проанализировать доступные решения для переноса структуры треков;
- изучить возможные способы получения данных о музыкальном плейлисте пользователя «ВКонтакте»;
- разработать веб-сайт, на котором от пользователя потребуется вход в его учётные записи «Deezer» и «ВКонтакте» посредством ввода пары логин/пароль.
- разработать серверное приложение, которое, после получения данных от веб-сайта, получит список аудиозаписей и плейлистов пользователя «ВКонтакте», а затем, используя API Deezer, добавит найденные в базе «Deezer» треки в аудиотеку.

Актуальность данной темы подтверждается отсутствием полноценного решения в свободном доступе. Использование данного программного обеспечения минимизирует временные и трудовые затраты со стороны пользователя.

СОДЕРЖАНИЕ

Введение	4
1 Анализ предметной области	6
1.1 Анализ существующих решений	6
1.2 Получение данных о музыкальных дорожках сети «ВКонтакте»	10
1.3 Взаимодействие с музыкальным потоковым сервисом «Deezer»	11
1.4 Выводы	12
2 Проектирование архитектуры веб-сервиса	13
2.1 Выбор средств разработки	14
2.1.1 Организация разработки	14
2.1.2 Разработка серверной части	15
2.1.3 Node.js	16
2.1.4 PHP	16
2.1.5 Ruby on rails	17
2.1.6 Python (Flask)	18
2.1.7 Язык разметки HTML	19
2.1.8 Каскадные таблицы стилей CSS	20
2.1.9 Язык программирования JavaScript	21
2.1.10 Система контроля версий Git	22
2.2 Обоснование выбора средств разработки	24
2.3 Выводы	25
3 Разработка веб-сервиса	26
3.1 Разработка клиентской части	26
3.2 Разработка серверной части	32
3.3 Тестирование	37
3.4 Выводы	37
Заключение	38
Список использованных источников	39

ВВЕДЕНИЕ

В настоящее время использование потоковых музыкальных сервисов становится распространённым явлением среди людей, увлекающихся музыкой. Развитие специализированных сервисов для прослушивания музыки позволило снизить процент нарушения авторских прав, а также помогло многим исполнителям получать прибыль за свой труд.

На ранних этапах развития социальной сети «ВКонтакте» в 2007 году появился раздел «аудиозаписи», который позволял пользователям загружать свои аудиодорожки, а также добавлять в свой плейлист чужие. До создания собственного музыкального сервиса, в социальной сети отсутствовала какая-либо модерация добавленных композиций, вследствие чего музыкальная библиотека данного сервиса превратилась в неструктурированную «свалку» аудиофайлов с огромным количеством повторений и отсутствием полноты информации об аудиодорожках.

Весной 2017 года социальная сеть «ВКонтакте» запустила собственный музыкальный сервис «Boom», который даёт пользователю возможность оформить платную подписку для неограниченного доступа к разделу «музыка» социальной сети «ВКонтакте». Также он позволяет слушать музыкальные композиции в офлайн-режиме, пользоваться подборками и рекомендациями от сервиса.

Многим пользователям «музыки ВКонтакте» долгое время не хватало структурированности их музыкальной библиотеки, а также оригинальной и полной информации о той или иной композиции, исполнителе или альбоме. Музыкальный сервис «Boom» был призван решить данные проблемы, но даже через 2 года после его запуска музыкальная библиотека «ВКонтакте» имеет некоторые недостатки.

С приходом на российский рынок других, в том числе зарубежных, потоковых музыкальных сервисов с большим объёмом функционала и возможностей, у пользователя появился выбор между различными вариантами. Рынок специализированных аудиосервисов за рубежом довольно широк, и, как следствие, у социальной сети «ВКонтакте» появились серьёзные конкуренты.

При переходе на другой музыкальный сервис пользователю необходимо заново создавать свою аудиотеку, а это может занять больше количество времени. И так как у пользователя социальной сети «ВКонтакте» уже собрана собственная библиотека треков, появляется необходимость в программном продукте, который позволит перенести большую часть библиотеки в другой сервис.

В качестве альтернативного музыкального сервиса в данном проекте выступает французский сервис «Deezer», который имеет обширную структурированную аудиотеку и официально работает на территории России по принципу платной подписки.

Целью выпускной квалификационной работы является создание веб-сервиса для переноса плейлистов пользователя «ВКонтакте» в музыкальный сервис «Deezer».

Для достижения цели были определены следующие задачи:

- проанализировать доступные решения для переноса структуры треков;
- изучить возможные способы получения данных о музыкальном плейлисте пользователя «ВКонтакте»;
- разработать веб-сайт, на котором от пользователя потребуется вход в его учётные записи «Deezer» и «ВКонтакте» посредством ввода пары логин/пароль.
- разработать серверное приложение, которое, после получения данных от веб-сайта, получит список аудиозаписей и плейлистов пользователя «ВКонтакте», а затем, используя API Deezer, добавит найденные в базе «Deezer» треки в аудиотеку.

1 Анализ предметной области

1.1 Анализ существующих решений

В данной главе рассмотрены существующие решения и их недостатки. На основе анализа сделаны выводы и сформулированы основные принципы относительно разработки приложения.

При изучении существующих аналогов были найдены следующие решения данной проблемы:

- англоязычный сервис soundiiz.com;
- способ переноса посредством скачивания списка треков с помощью сайта vkpls.boutnew.ru и последующей загрузкой полученного списка на сайт playlist-converter.net.

Рассмотрим сервис soundiiz.com (рисунок 1). Данный программный продукт позволяет переносить аудиотеку пользователя между различными официальными музыкальными площадками.

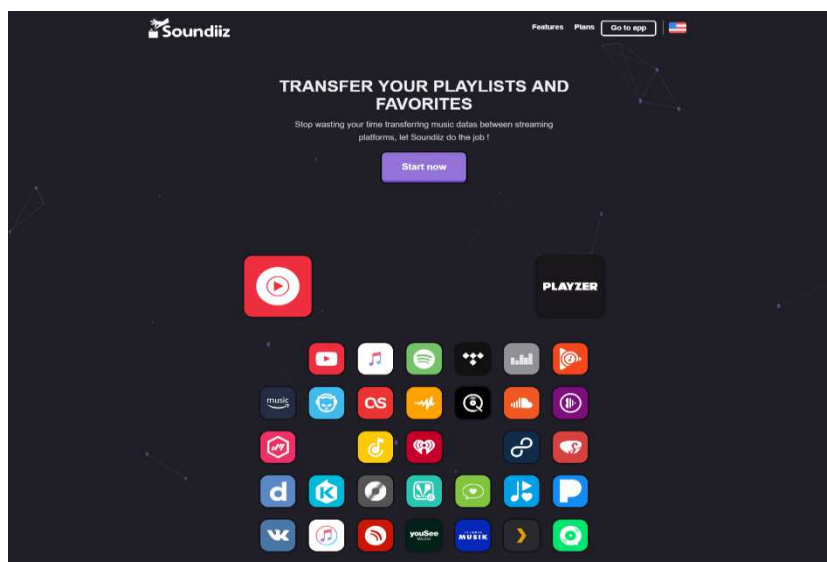


Рисунок 1 – Сервис soundiiz.com

Данное веб-приложение включает возможность переноса треков между 34 музыкальными стриминговыми сервисами (рисунок 2). В список включены площадки «ВКонтакте» и «Deezer».

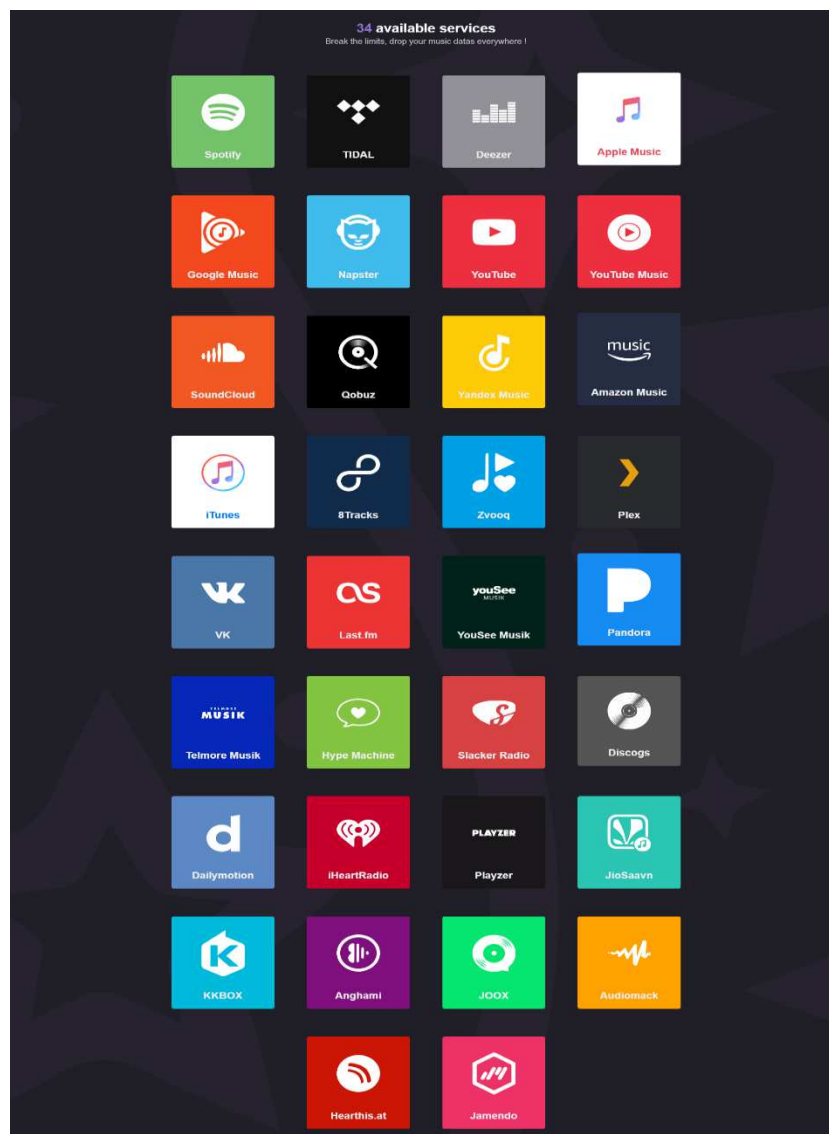


Рисунок 2 – Список музыкальных площадок

Функционал Soundiiz позволяет взаимодействовать со всеми представленными площадками для синхронизации музыкальных плейлистов. Недостатками данного решения являются:

- ограниченность функционала бесплатной версии приложения в виде лимита на перенос треков в количестве 300 шт.;
- использование персональных данных пользователя для подключения к социальной сети «ВКонтакте».

Чтобы открыть доступ к своей аудиотеке «ВКонтакте», пользователю необходимо ввести свои данные в неофициальную форму (рисунок 3). Данный метод авторизации может повлечь за собой утечку персональных данных.

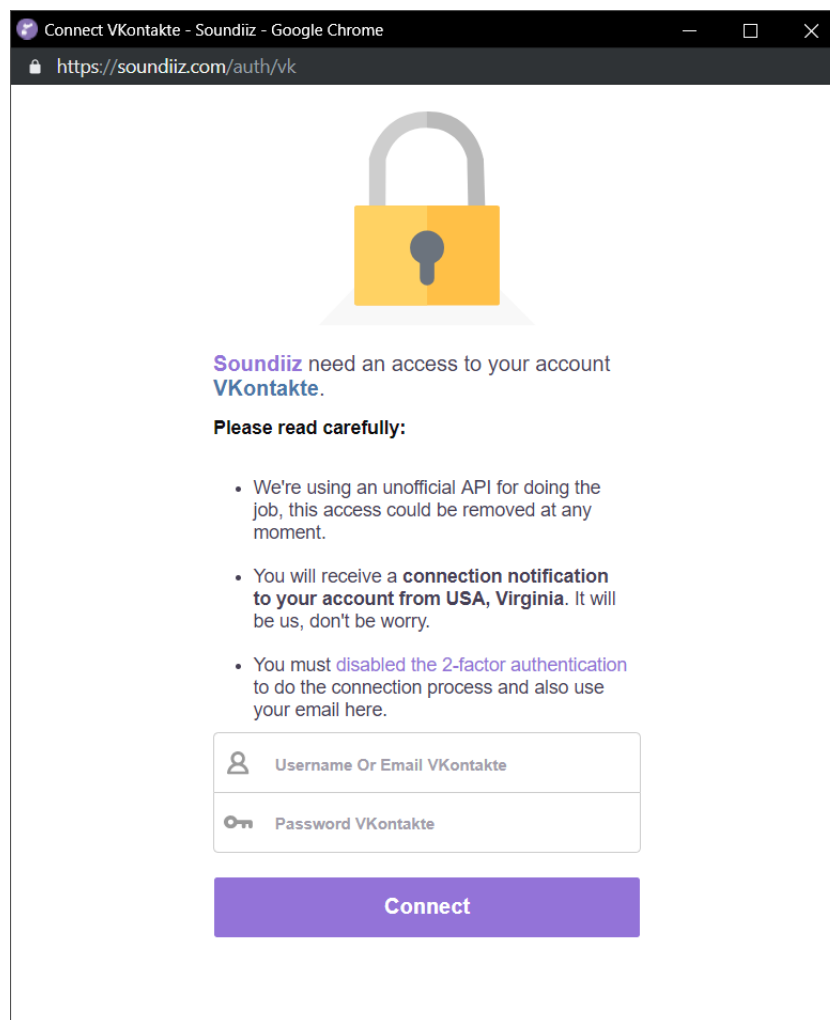


Рисунок 3 – Форма авторизации soundiiz.com

Второй способ базируется на использовании двух веб-приложений: vkpls.bouthnew.ru и playlist-converter.net. При использовании первого веб-приложения необходимо ввести ID страницы «ВКонтакте» и количество аудиозаписей для переноса (рисунок 4).

Рисунок 4 – Процесс загрузки списка треков с помощью vkpls.bouthnew.ru

Затем пользователь выбирает формат файла для загрузки (рисунок 5).



Рисунок 5 – Выбор формата плейлиста

Через некоторое время пользователю будет доступен для скачивания файл со списком треков его аудиотеки «ВКонтакте».

Следующим этапом данного метода является загрузка скачанного файла в сервис playlist-converter.net (рисунок 6).



Рисунок 6 – Загрузка списка треков на playlist-converter.net

После загрузки списка композиций пользователь выбирает сервис, в который он желает добавить свою аудиотеку и происходит добавление аудиодорожек в библиотеку на музыкальной площадке (рисунок 7).



Рисунок 7 – Выбор целевого сервиса для будущего плейлиста

Недостатками рассмотренного метода являются:

- большое количество затраченного на перенос времени (для 1000 треков это время составляет порядка 15 минут);
- необходимость совершать дополнительные действия помимо входа в учётные записи;
- перебои с доступом к веб-сайту vkpls.bouthnew.ru, что может повлечь за собой полное отсутствие возможности переноса треков.

При анализе доступных способов решения поставленной задачи были рассмотрены самые популярные методы. При рассмотрении недостатков каждого из методов становится ясно, что разрабатываемый программный продукт может быть востребован потребителями.

1.2 Получение данных о музыкальных дорожках сети «ВКонтакте»

Возможность получения данных из музыкального раздела социальной сети «ВКонтакте» является одной из основных проблем в разработке веб-сервиса.

Стандартный способ получения данных от того или иного веб-сервиса заключается во взаимодействии с API (Application Programming Interface), методы которого позволяют получать необходимые структурированные данные для дальнейшей обработки. Использование данного интерфейса предполагает авторизацию приложения с необходимыми параметрами доступа. Причиной этого является взаимодействие с API из программного кода, без непосредственного участия пользователя. При авторизации пользователь даёт разрешение на доступ приложения к необходимым разделам сайта. После авторизации пользователя приложению предоставляется access token (токен доступа), который используется в запросах к серверу для подтверждения права на получение тех или иных данных.

При изучении официального API «ВКонтакте» выяснилось, что необходимого метода доступа к разделу музыка «ВКонтакте» не существует. После запуска своего музыкального сервиса «Boom» разработчики социальной сети закрыли доступ к методам audio своего API. При обращении в техническую

поддержку социальной сети стало ясно, что разработчики в ближайшее время не собираются открывать доступ к разделу «музыка».

Дальнейшее изучение данного вопроса привело к решению использовать неофициальное API «ВКонтакте» для взаимодействия с разделом «музыка» под видом пользователя. Самыми популярными решениями в данной области являются EasyVK и vk_api.

EasyVK – это библиотека для Node.js, созданная сторонним разработчиком, которая позволяет взаимодействовать с API «ВКонтакте» двумя способами:

- взаимодействие с доступными методами официального API;
- взаимодействие с социальной сетью с помощью http-запросов под видом пользователя для извлечения данных с последующей их обработкой для представления в формате json.

Основным недостатком данной библиотеки является нестабильная работа необходимых для работы веб-сервиса методов. Методы получения списка треков срабатывают не всегда, а также получают неполный список треков. Приведённый недостаток послужил причиной для изучения библиотеки vk_api.

vk_api – это библиотека для Python от сторонних разработчиков, которая предоставляет подобный с EasyVK функционал, но с некоторыми отличиями. При сравнении данных библиотек было выявлено, что vk_api не имеет проблем со стабильностью работы необходимых методов, а также предоставляет нужные данные в удобном виде.

1.3 Взаимодействие с музыкальным потоковым сервисом «Deezer»

API музыкального сервиса Deezer имеет все необходимые для работы разрабатываемого продукта методы. Как отмечено выше, API предполагает авторизацию со стороны пользователя.

Стандартным для большинства сервисов является протокол авторизации OAuth 2.0, разработанный IETF OAuth Working Group. Данный протокол упрощает взаимодействие с API посредством предоставления механизмов авторизации для различных типов приложений. Использование протокола

OAuth 2.0 требует предварительной регистрации приложения, которому нужен доступ к API, в личном кабинете разработчика. После создания приложения и добавления всех необходимых данных разработчик получает ID своего приложения и секретный ключ от него, которые затем использует для формирования запроса на получение access token при авторизации пользователя. Данный механизм позволяет встраивать официальные формы авторизации на страницах сторонних приложений, а также предотвращает утечку авторизационных данных.

1.4 Выводы

В данной главе были рассмотрены и проанализированы доступные на данный момент решения. Итогом анализа являются следующие выводы:

- необходимо создание бесплатного веб-приложения для переноса аудиотеки;
- необходимо использование официальных методов авторизации в социальной сети и на музыкальной площадке;
- необходимо совмещение всех нужных функций в одном месте.

2 Проектирование архитектуры веб-сервиса

При проектировании архитектуры приложения использовались выводы, сделанные при анализе существующих решений. Так как разрабатывается веб-приложение, суть работы которого заключена во взаимодействии с различными API, то здесь подойдёт архитектура «клиент – сервер». Данный тип архитектуры позволяет отделить логику приложения от его внешней составляющей. Чтобы не нагружать пользовательское устройство, все запросы к «ВКонтакте» и «Deezer» во время процесса переноса отправляются с сервера, где и сосредоточена вся основная логика приложения (рисунок 8).

Для взаимодействия пользователя с разрабатываемым веб-сервисом в качестве интерфейса используется одностраничный сайт. Клиентская сторона отвечает за авторизацию серверного приложения на площадках, а также отображение результатов переноса. Пользователь входит на сайт, авторизуется на площадках и нажимает кнопку «Перенести треки», после чего в дело вступает серверная часть приложения.

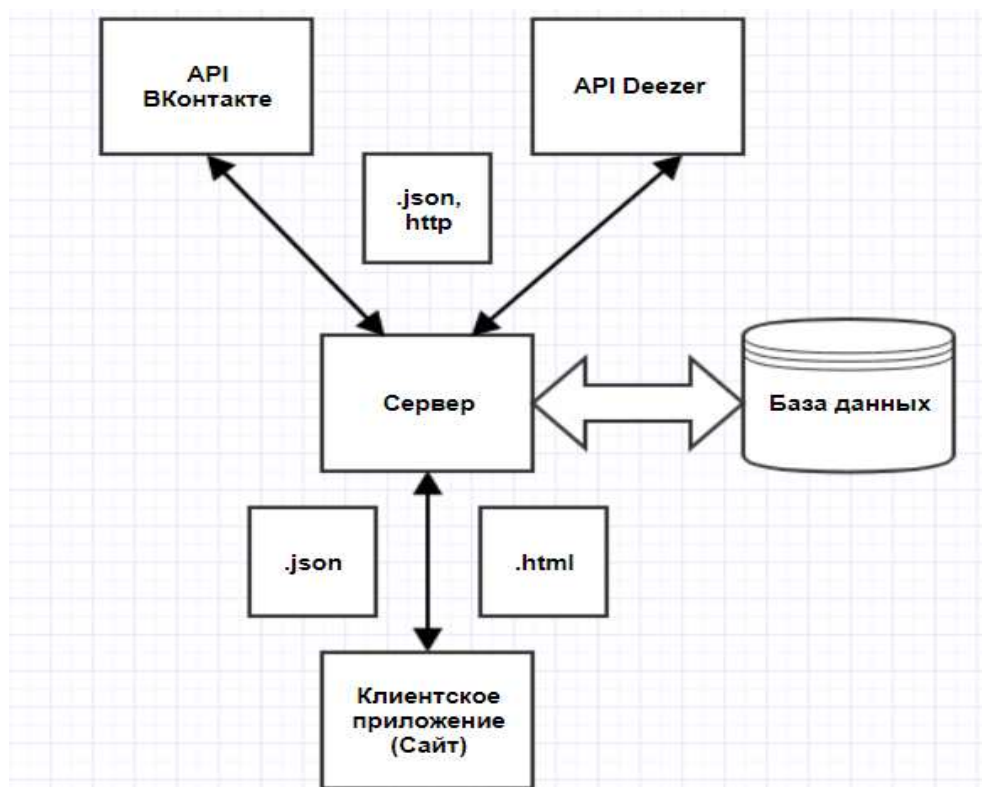


Рисунок 8 – Общая архитектура приложения

Так как сервисом в один момент времени могут пользоваться несколько человек, на сервере должен использоваться сессионный механизм работы. Данные, относящиеся к каждому пользователю, хранятся отдельно. Количество используемых данных имеет малый объём, в связи с этим база данных должна иметь минимальный набор функций для временного хранения данных, которые будут удалены после завершения работы конкретной сессии. Структура базы данных приведена на рисунке 9 и включает в себя 2 таблицы, в которых хранится необходимая для переноса треков информация.

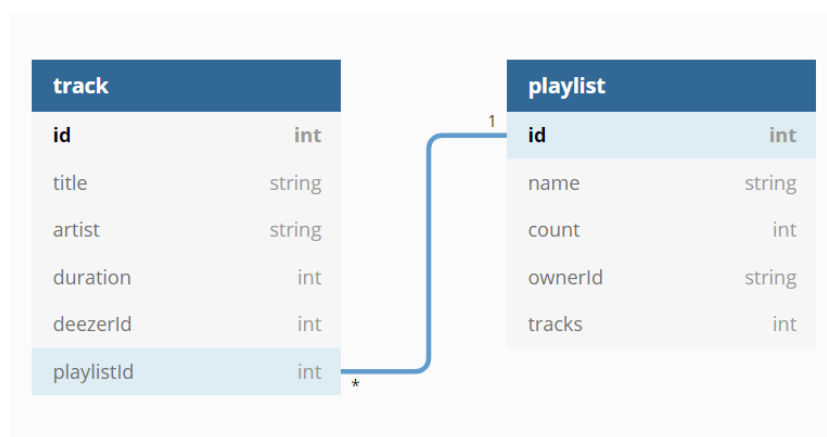


Рисунок 9 – Структура базы данных

2.1 Выбор средств разработки

Важную роль в разработке программного обеспечения играют правильно подобранные инструменты. Результатом использования соответствующих программных продуктов является прогнозируемый и простой процесс разработки.

2.1.1 Организация разработки

Для организации и планирования процесса разработки, выделения ключевых моментов, определения сроков и распределения задач по сложности был использован онлайн сервис Trello.

Trello – это одна из самых популярных систем управления проектами в режиме онлайн, которая пользуется особым спросом среди небольших

компаний и стартапов. Она позволяет эффективно организовывать работу по японской методологии канбан-досок [1].

Система создана компанией Fog Creek Software в 2011 году на базе MongoDB, Backbone.js и Node.js. Главные достоинства, которые позволили Trello добиться популярности – это:

- простой интерфейс;
- почти неограниченный бесплатный доступ;
- удобство в работе и возможность интеграции с другими популярными инструментами для онлайн-работы.

Freemium-модель – второе после простоты самое существенное преимущество Trello перед аналогами. В Trello можно полноценно работать совершенно бесплатно. Платная версия этого сервиса отличается большим количеством интеграций, объёмом загружаемых файлов и возможностью смены фона досок на кастомные. Но и это не беда: сейчас Trello оформил партнерство с Unsplash, и обои можно выбрать из коллекции известного фотостока.

2.1.2 Разработка серверной части

В данном проекте серверная часть приложения является основной, и в ней сосредоточена вся логика и суть программного продукта. В виду этого выбранный инструмент для разработки сервера должен обеспечивать разработчика необходимым функционалом, а также иметь возможность подключать сторонние библиотеки, которые, в свою очередь, позволят более качественно реализовать конечный продукт. Обработка информации и взаимодействие с API ВКонтакте и Deezer происходит на сервере, поэтому было необходимо выбрать мощный и в то же время простой инструмент для создания серверной логики.

2.1.3 Node.js

Node.js – это серверная платформа для работы с JavaScript через движок V8. JavaScript выполняет действие на стороне клиента, а Node – на сервере. С помощью Node можно писать полноценные приложения. Node умеет работать с внешними библиотеками, вызывать команды из кода на JavaScript и выполнять роль веб-сервера [2].

С Node проще масштабироваться. При одновременном подключении к серверу тысяч пользователей Node работает асинхронно, то есть ставит приоритеты и распределяет ресурсы грамотнее. Java же, например, выделяет на каждое подключение отдельный поток.

Node появился в 2009 году благодаря Райану Далю. До этого в серверах царил подход «один поток на каждое соединение», а Дал придумал использовать систему, которая ориентирована на события, то есть реагирует на действие или бездействие и выделяет под это ресурс. Главная цель Node – построение масштабируемых сетевых серверов [3].

Благодаря использованию одного и того же языка программирования на клиенте и сервере, упрощается работа с данными, так как их формат одинаково стандартизирован в обеих частях приложения. Кроме этого, упрощается разработка, так как разработчику не приходится переключаться с одного языка на другой. Учитывая, что специфика языков программирования иногда ощутимо различается, разработчик не всегда может быстро и эффективно переключаться между языками, что влечет за собой ошибки и потерю времени.

2.1.4 PHP

PHP – скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов [4].

Язык и его интерпретатор (Zend Engine) разрабатываются группой энтузиастов в рамках проекта с открытым кодом. Проект распространяется под собственной лицензией, несовместимой с GNU GPL [5].

Из коробки имеет проблемы с пространствами имен, типизацией и безопасностью. Кроме того, факт не высокой сложности языка провоцирует неопытных разработчиков писать именно на это языке, что ведет к тому, что комьюнити языка находится на ужасающем уровне.

2.1.5 Ruby on rails

Ruby – динамический, рефлексивный, интерпретируемый высокоуровневый язык программирования. Язык обладает независимой от операционной системы реализацией многопоточности, строгой динамической типизацией, сборщиком мусора и многими другими возможностями. По особенностям синтаксиса он близок к языкам Perl и Eiffel, по объектно-ориентированному подходу – к Smalltalk. Также некоторые черты языка взяты из Python, Lisp, Dylan и Клу [6].

Кроссплатформенная реализация интерпретатора языка является полностью свободной.

Ruby on Rails (RoR) – фреймворк, написанный на языке программирования Ruby, реализует архитектурный шаблон Model-View-Controller для веб-приложений, а также обеспечивает их интеграцию с веб-сервером и сервером баз данных. Является открытым программным обеспечением и распространяется под лицензией MIT.

Создан Давидом Хейнемейером Ханссоном на основе его работы в компании 37signals над средством управления проектами Basecamp и выпущен в июле 2004 года. 23 декабря 2008 года команда проекта Merb объединилась с командой Rails с целью создания следующей версии Rails 3, которая объединит в себе лучшие черты обоих фреймворков.

Базируется на следующих принципах разработки приложений: максимальное использование механизмов повторного использования, позволяющих минимизировать дублирование кода в приложениях (принцип Don't repeat yourself); по умолчанию используются соглашения по конфигурации, типичные для большинства приложений (принцип Convention over configuration) – явная спецификация конфигурации требуется только в нестандартных случаях [7].

2.1.6 Python (Flask)

Python – высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций [8].

Python поддерживает несколько парадигм программирования, в том числе структурное, объектноориентированное, функциональное, императивное и аспектно-ориентированное. Основные архитектурные черты – динамическая типизация, автоматическое управление памятью, механизм обработки исключений, поддержка многопоточных вычислений и удобные высокоуровневые структуры данных. Код в Python организовывается в функции и классы, которые могут объединяться в модули (они в свою очередь могут быть объединены в пакеты).

Эталонной реализацией Python является интерпретатор CPython, поддерживающий большинство активно используемых платформ. Он распространяется под свободной лицензией Python Software Foundation License, позволяющей использовать его без ограничений в любых приложениях, включая проприетарные. Есть реализации интерпретаторов для JVM (с возможностью компиляции), MSIL (с возможностью компиляции), LLVM и других. Проект PyPy предлагает реализацию Python с использованием JIT-компиляции, которая значительно увеличивает скорость выполнения Python-программ.

Python – активно развивающийся язык программирования, новые версии (с добавлением/изменением языковых свойств) выходят примерно раз в два с половиной года. Вследствие этого и некоторых других причин на Python отсутствуют стандарт ANSI, ISO или другие официальные стандарты, их роль выполняет CPython [9]. Flask – это серверный микрофреймворк, реализованный на языке программирования Python. Приставка «микро» означает, что фреймворк является легковесным и имеет мало изначально встроенных функций. И в этом кроется его основное преимущество. Изначальная простота данного программного обеспечения открывает перед разработчиком большие возможности по настройке. Данный фреймворк используется в связке с серверными инструментами Gunicorn и Nginx, которые позволяют предоставлять доступ к веб-приложению по айпи адресу сервера [10].

Так же большим преимуществом в сторону выбора данного фреймворка является язык, который используется в работе с этим программным продуктом. Python сочетает в себе огромное количество разнообразных библиотек, простоту и лаконичность в написании кода, а так же развитое сообщество с большим количеством готовых решений для самых разнообразных задач.

2.1.7 Язык разметки HTML

HTML (от англ. HyperText Markup Language – «язык гипертекстовой разметки») – стандартизированный язык разметки документов во Всемирной паутине. Большинство веб-страниц содержат описание разметки на языке HTML (или XHTML). Язык HTML интерпретируется браузерами; полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства [11].

Язык HTML был разработан британским учёным Тимом Бернерсом-Ли приблизительно в 1986-1991 годах в стенах ЦЕРНа в Женеве в Швейцарии. HTML создавался как язык для обмена научной и технической документацией, пригодный для использования людьми, не являющимися специалистами в области вёрстки.

HTML успешно справлялся с проблемой сложности SGML путём определения небольшого набора структурных и семантических элементов – дескрипторов. Дескрипторы также часто называют «тегами». С помощью HTML можно легко создать относительно простой, но красиво оформленный документ. Помимо упрощения структуры документа, в HTML внесена поддержка гипертекста. Мультимедийные возможности были добавлены позже.

Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (отображения). В идеале, текст с разметкой HTML должен был без стилистических и структурных искажений воспроизводиться на оборудовании с различной технической оснащённостью (цветной экран современного компьютера, монохромный экран органайзера, ограниченный по размерам экран мобильного телефона или устройства и программы голосового воспроизведения текстов). Однако современное применение HTML очень далеко от его изначальной задачи [12].

В целом в настоящее время по популярности и размеру сообщества у этого языка нет конкурентов.

2.1.8 Каскадные таблицы стилей CSS

Каскадные таблицы стилей (Cascading Style Sheets = CSS) – это язык, который отвечает за визуальное представление документов пользователю.

Под документом мы будем понимать набор информации о структуре страницы, описываемый языком разметки.

А представление документа пользователю, в свою очередь, означает его преобразование в удобную для восприятия форму. Браузеры, такие как Firefox, Chrome или Internet Explorer, были созданы для визуального отображения документов, например, на экране компьютера, проекторе или вывода через принтер [13].

В то время как язык разметки HTML создает структуру документа, язык стилей CSS описывает стилистические свойства компонентов страниц. Такие как позиционирование, внешний вид и другие.

Кроме того, CSS позволяет программировать базовые вещи в поведении объектов. Например, реакция элемента на наведения указателя мыши.

Нужно отметить что в данном проекте было принято решение использовать библиотеку bootstrap.css. Которая содержит в себе хорошо выглядящие css модули. Их удобно переиспользовать за счет чего многократно ускоряется процесс разработки приложений [14].

2.1.9 Язык программирования JavaScript

JavaScript изначально создавался для того, чтобы сделать web-страницы «живыми». Программы на этом языке называются скриптами. В браузере они подключаются напрямую к HTML и, как только загружается страница – тут же выполняются.

Во все основные браузеры встроен интерпретатор JavaScript, именно поэтому они могут выполнять скрипты на странице. Но, разумеется, JavaScript можно использовать не только в браузере. Это полноценный язык, программы на котором можно запускать и на сервере, и даже в стиральной машинке, если в ней установлен соответствующий интерпретатор.

Современный JavaScript – это «безопасный» язык программирования общего назначения. Он не предоставляет низкоуровневых средств работы с памятью, процессором, так как изначально был ориентирован на браузеры, в которых это не требуется.

Что же касается остальных возможностей – они зависят от окружения, в котором запущен JavaScript. В браузере JavaScript умеет делать всё, что относится к манипуляции со страницей, взаимодействию с посетителем и, в какой-то мере, с сервером:

- Создавать новые HTML-теги, удалять существующие, менять стили элементов, прятать, показывать элементы и т.п.;
- реагировать на действия посетителя, обрабатывать клики мыши, перемещения курсора, нажатия на клавиатуру и т.п.;
- посылать запросы на сервер и загружать данные без перезагрузки страницы (эта технология называется «AJAX»);
- получать и устанавливать cookie, запрашивать данные, выводить сообщения [15].

2.1.10 Система контроля версий Git

Система контроля версий Git – обязательная часть без которой не обходится ни один проект, разрабатываемый в команде. Система контроля версий или VCS может значительно облегчить работу разработчиков, пытающихся проанализировать изменения и вклады в общий код. Проще говоря, система контроля версий – это ключевой элемент в системе управления настройками программного обеспечения, которые отвечают потребностям проекта. VCS дают возможность назначать для определенных изменений/ревизий/обновлений буквенные или числовые значения. Также могут предоставить информацию о временных метках и идентификаторе человека внесшего изменения [16].

В 2005 году, Линус Торвальдс (человек, известный тем, что стоял за созданием ядра Linux OS) разработал GIT для управления версиями ядра Linux, однако, чуть позднее его поддержкой занялся другой человек, японский инженер-программист – Джунио Хамано. На сегодняшний день, GIT – это одна из самых известных систем контроля версий с открытым исходным кодом, на которую полагаются миллионы проектов по всем миру (включая как коммерческие, так и бесплатные проекты). GIT – это полностью бесплатное программное обеспечение, поддерживающее множество ОС, таких как Mac, Linux, Windows и Solaris. Вот несколько функций GIT достойных упоминания:

- распределенная система управления версиями, GIT следует принципу одноранговой сети – peer to peer (равный к равному) в отличие от других систем вроде Subversion (SVN), которая основана на модели client-server (клиент-сервер);

- GIT позволяет разработчикам иметь множество совершенно независимых веток кода. Создание, удаление и объединение этих веток происходит без каких-либо проблем и больших затрат времени;

- в GIT все операции атомарны; это означает, что любое действие может быть полностью удачным или провалиться (без каких-либо изменений). Это действительно важно, так как в некоторых системах контроля версий (вроде CVS), где действия не атомарны, некоторые повисшие операции по всему хранилищу, могут оставить его в нестабильном состоянии;

- в отличие от других VCS, таких как SVN или CVS где метаданные хранятся в скрытых папках (.cvs, .svn, и т.д.), в GIT все данные расположены в каталогах .git;

- он использует модель данных, которая помогает обеспечить криптографическую целостность всего, что присутствует в репозитории. Каждый раз, когда файлы добавляются или коммитятся, генерируются их контрольные суммы; аналогичный процесс происходит при их извлечении;

- ещё одна превосходная функция, присутствующая в GIT – это его индекс. В пределах индекса, разработчики могут форматировать коммиты и просматривать их до фактического применения [17].

Он довольно прост в использовании. Для начала, вы можете либо создать новое хранилище, либо добавить существующее. После установки, команда `git-init` поможет вам с настройкой нового хранилища, или команда `git clone` поможет настроить пользователя для работающей копии локального хранилища [18].

2.2 Обоснование выбора средств разработки

В качестве языка программирования для серверной части было принято решение использовать язык программирования Python по нескольким причинам:

- написанная на Python библиотека vk_api;
- достаточное количество пакетов и модулей;
- большое сообщество;
- знания программиста.

Языки HTML/CSS для верстки страниц являются стандартом в разработке веб-приложений, поэтому их использование очевидно. Точно так же, как и JavaScript, для создания интерактивности на странице.

Для облегчения разработки дружелюбного интерфейса приложения используется фреймворк Bootstrap, который, помимо большого количества предопределённых стилей, имеет встроенную адаптивность, что позволит разработать сайт, который будет выглядеть одинаково приятно на любых экранах.

Написание запросов от клиента к серверу на чистом JavaScript имеет свои особенности, поэтому для реализации общения с сервером принято решение использовать фреймворк JQuery, который имеет все необходимые функции для отправки и получения данных.

В качестве системы контроля версий была выбрана git, которая используется повсеместно для отслеживания изменений в проекте.

Система управления баз данных Sqlite выбрана за свою доступность, а также из-за отсутствия большого количества хранимой информации, которая удаляется сразу после завершения сеанса использования веб-сервиса.

Для написания серверной части было принято решение использовать фреймворк Flask, реализованный на языке программирования Python. Использование данного фреймворка объясняется его легковесностью и простотой с возможностью гибкой настройки под нужды разработчика. Так как

этот фреймворк реализован на Python, то это даёт программисту обширные возможности работы со строковыми и json данными.

В качестве среды разработки был выбран редактор Visual Studio Code. Он имеет встроенную подсветку кода, поддержку обширного набора языков программирования, инструменты отладки, встроенные механизмы взаимодействия с командной строкой, большое количество расширений для различных нужд. Использование расширений для выбранного редактора позволяет вести разработку через ssh-соединение на удалённом сервере и подключать репозитории.

2.3 Выводы

В данном разделе была спроектирована архитектура приложения, рассмотрены самые популярные инструменты для разработки веб-сервисов выбранной архитектуры. Выбранные технологии обеспечили быструю, надёжную и прогнозируемую разработку программного обеспечения.

3 Разработка веб-сервиса

Разработку веб-сервиса было решено начать с разработки клиентской части. Получение авторизационных данных необходимо для разработки и отладки серверной части. Поэтому были реализованы механизмы авторизации, а затем реализована серверная часть.

3.1 Разработка клиентской части

Алгоритм работы сервиса заключается в получении авторизационных данных от пользователя с последующим взаимодействием с социальной сетью «ВКонтакте» и музыкальным сервисом «Deezer» с сервера.

На этапе анализа существующих решений был сделан вывод относительно обязательного использования официальных средств авторизации. Ограничения, наложенные разработчиками социальной сети «ВКонтакте», заставляют использовать обычную учётную запись «ВКонтакте», с помощью которой будут извлекаться данные. В связи с этим принято решение использовать авторизацию «ВКонтакте» лишь для получения Id пользователя. В свою очередь, использование авторизации на музыкальной площадке «Deezer» подразумевает получение полного доступа к учётной записи пользователя. Так как в социальной сети и на музыкальном сервисе используется один и тот же протокол авторизации OAuth 2.0, способ получения авторизационных данных от этих веб-сервисов почти идентичен, за исключением некоторых моментов. В результате нам необходимы 2 кнопки на странице приложения (рисунки 10 и 11), по нажатию которых будут вызываться функции, написанные на языке JavaScript, именуемые `openVK` и `openDeezer` соответственно (рисунок 12).

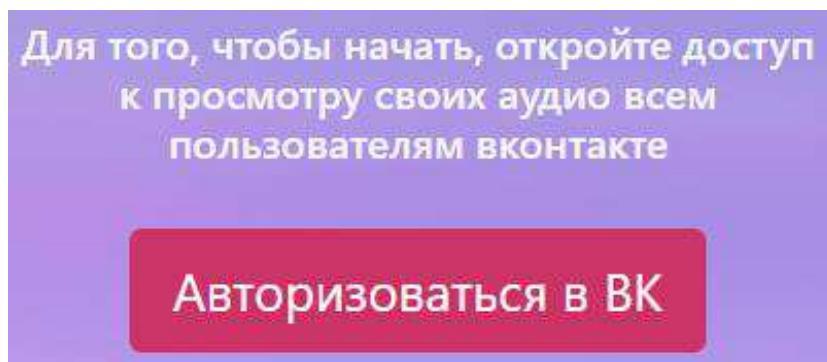


Рисунок 10 – Кнопка «Авторизоваться в ВК» и аннотация к ней

Так как при разработке используется стороннее API «ВКонтакте», приложение использует учётную запись, которая никак не связана с конечным пользователем. Следствием этого является необходимость доступа к списку треков конечного пользователя. Для уточнения требуемых от пользователя действий, над кнопками приведены краткие объяснения. Это действие выполняется в настройках социальной сети «ВКонтакте» самим пользователем и не имеет никаких сложностей.

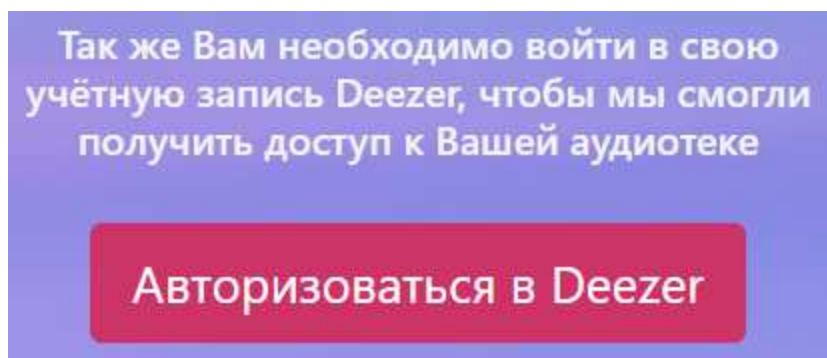


Рисунок 11 – Кнопка «Авторизоваться в Deezer» и аннотация к ней

```
function openDeezer(){
    window.open('https://connect.deezer.com/oauth/auth.php?'
    + 'app_id=352824&redirect_uri=http://localhost:5000/getDeezerAccess'
    + '&perms=basic_access,manage_library,offline_access',
    'RegWindow',
    'width=600,height=600');
};

function openVK(){
    window.open('https://oauth.vk.com/authorize?client_id=7013007'
    + '&display=page&redirect_uri=http://localhost:5000/getVKAccess'
    + '&scope=audio,offline&response_type=code&v=5.95',
    'RegWindow',
    'width=600,height=600');
};
```

Рисунок 12 – Код функций openVK и openDeezer

При нажатии на каждую из кнопок вызывается роупр-окно, в котором пользователь вводит свои данные (рисунки 13 и 14).

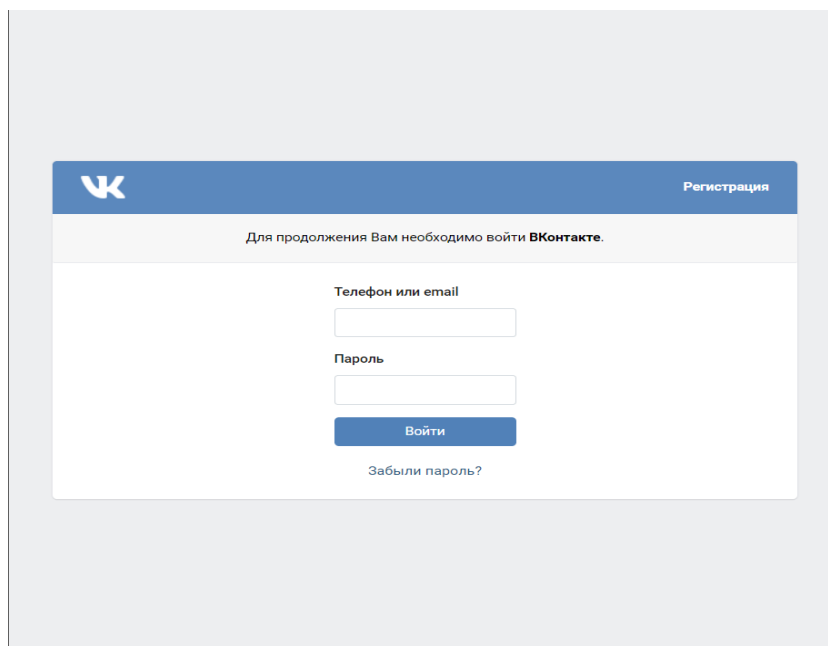


Рисунок 13 – Окно авторизации «ВКонтакте»

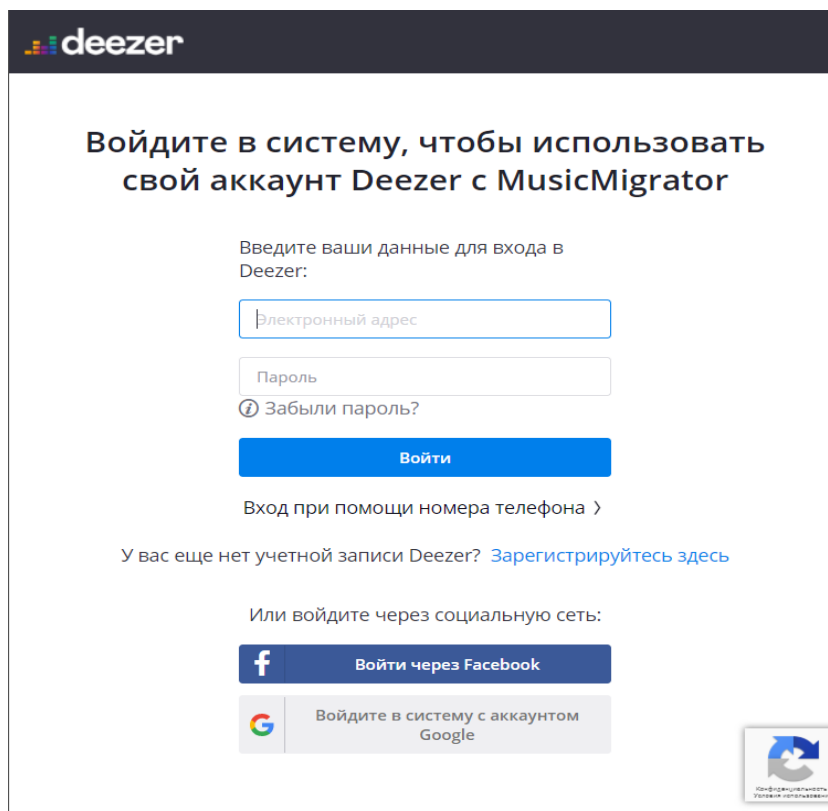


Рисунок 14 – Окно авторизации «Deezer»

После ввода корректных авторизационных данных пользователем вызываются серверные методы `getDeezerAccess` и `getVKAccess` соответственно для каждой кнопки, которые на данном этапе разработки являются заглушками без функционала. URL-адреса этих методов регистрируются на сервисах в кабинете разработчика при создании приложения.

После успешных авторизаций на веб-сервисах пользователю необходимо нажать кнопку для запуска процесса переноса (рисунок 15). Кнопка вызывает функцию `transfer_tracks`, которая также реализована на языке JavaScript (рисунок 16). Данная функция вызывает анимированный элемент, который даёт пользователю понимание о том, что происходит процесс переноса, вместе с этим отправляя post-запрос на сервер по адресу `/transfer`.

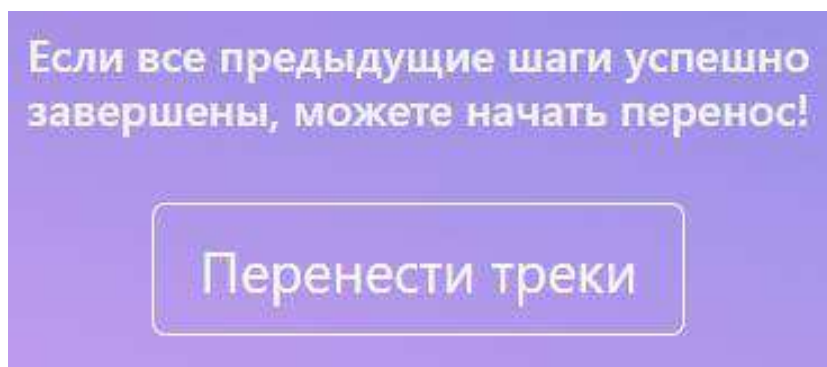


Рисунок 15 – Кнопка «Перенести треки» и аннотация к ней

```
function transfer_tracks(){
    $('#missing_tracks').html('');
    $('#hidden_row').show();
    $.post('/transfer', {
    }).done(function(response){
        var str = ""
        $.each(response['text'], function(index, value){
            str = str + value['artist'] + ' - ' + value['title'] + '<br \>'
        });
        $('#missing_tracks').html(str);
    })
};
```

Рисунок 16 – Код функции `transfer_tracks`

Для предоставления наглядной информации создан блок фиксированной высоты с вертикальным скролл-баром, в котором, после завершения переноса, анимированный элемент заменяется списком песен, которые не удалось найти в аудиотеке Deezer (рисунки 17 и 18).

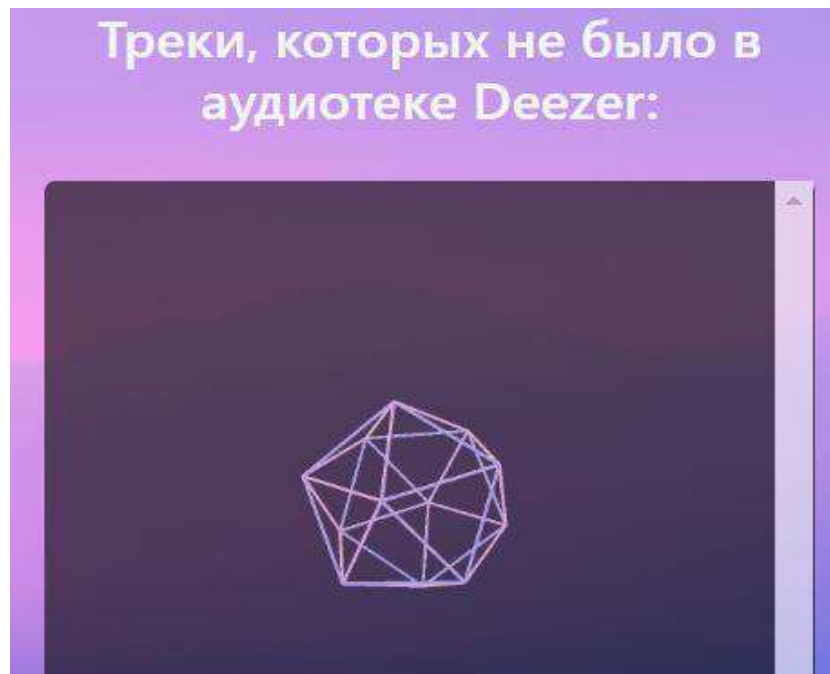


Рисунок 17 – Блок для отображения результатов во время переноса



Рисунок 18 – Блок для отображения результатов после завершения переноса

Общий внешний вид приложения представлен ниже на рисунках 19, 20 и 21.

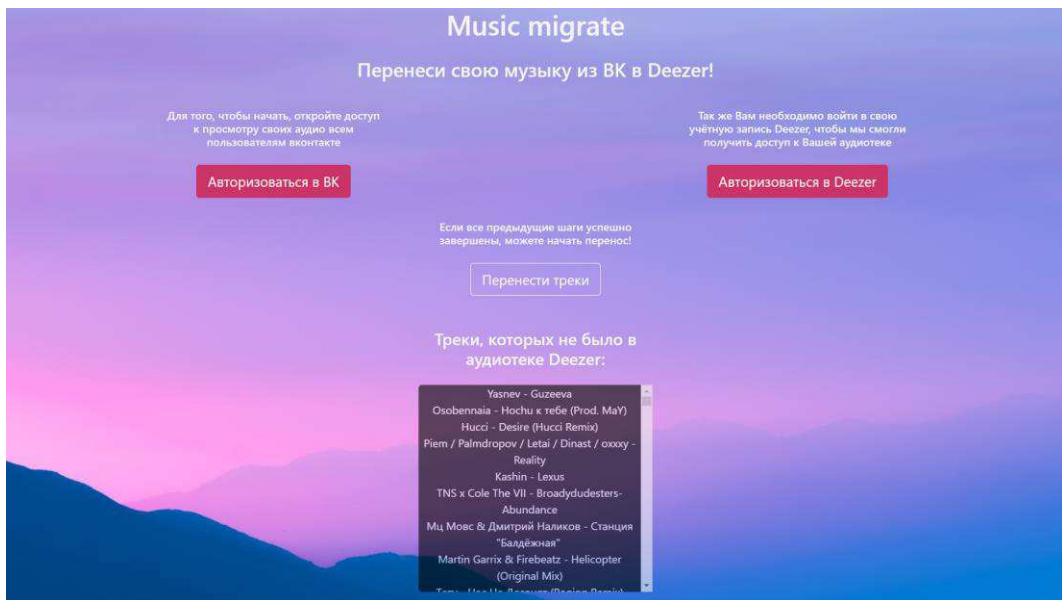


Рисунок 19 – Внешний вид веб-сервиса после завершения работы

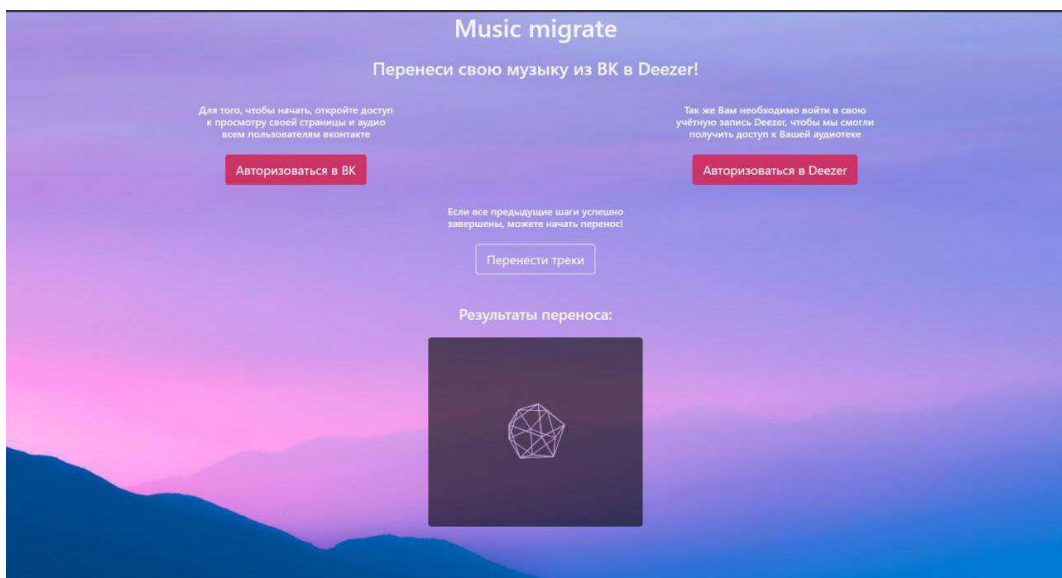


Рисунок 20 – Внешний вид веб-сервиса во время выполнения переноса

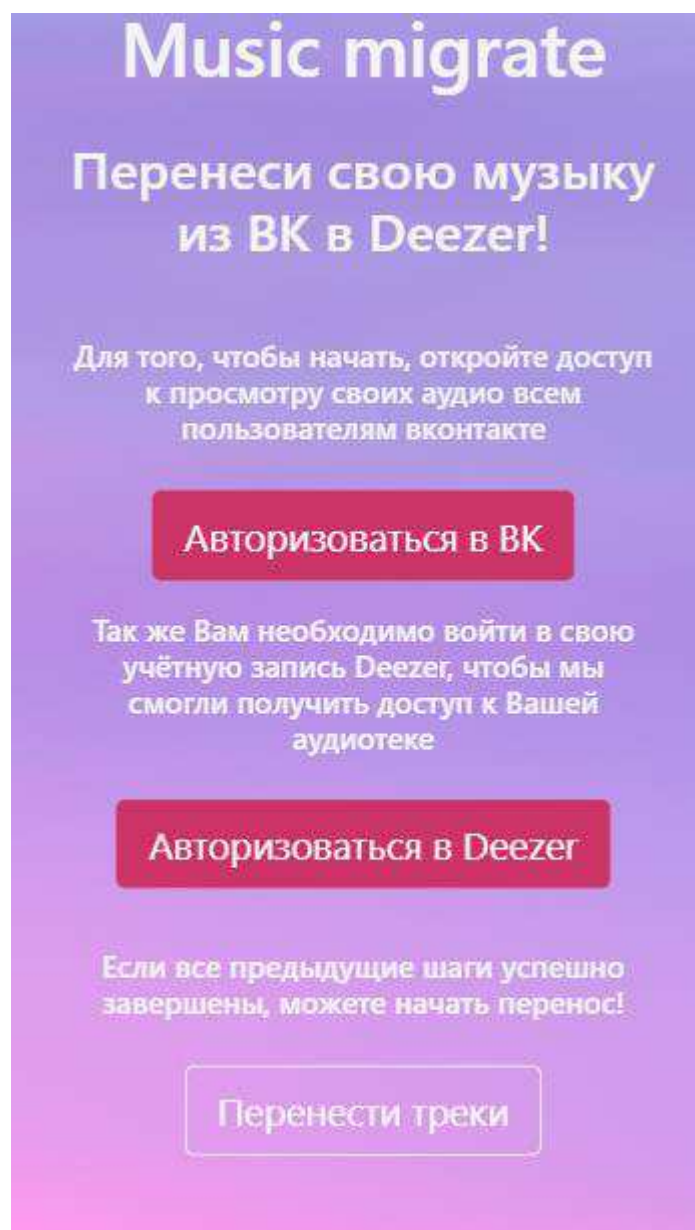


Рисунок 21 – Внешний вид веб-сервиса на мобильном устройстве

3.2 Разработка серверной части

Для создания серверной части было решено продумать разработать стратегию последовательного получения и обработки данных от социальной сети «ВКонтакте». Получение данных от социальной сети занимает некоторое время, и обусловлено это ограничениями на количество http-запросов в секунду.

После разработки клиентской части были выявлены необходимые URL-адреса для запросов, Список данных URL-адресов следующий:

- GET: /, /index для получения страницы приложения;
- GET, POST: /getVKAccess для получения id пользователя «ВКонтакте»;
- GET, POST: /getDeezerAccess для получения access token от учётной записи «Deezer»;
- GET, POST: /transfer для запуска процесса переноса и последующего ответа клиенту со списком отсутствующих треков.

Для реализации URL-адресов Flask предоставляет разработчику декораторы для обёртки обычной python-функции. Результатом является модуль routes.py, в котором описаны все возможные маршруты, доступные для клиента (рисунок 22).

```

from flask import render_template, jsonify, request, session
import requests
from app import app
import VKtoDeezer

@app.route('/')
@app.route('/index')
def index():
    ... return render_template('index.html', title='MusicMigrate -

@app.route('/transfer', methods=['GET', 'POST'])
def transfer_tracks():
    ... return jsonify({'text': VKtoDeezer.transfer_tracks(session

@app.route('/getDeezerAccess', methods=['GET', 'POST'])
def get_deezer_access():
    ... requestUri = 'https://connect.deezer.com/oauth/access_token
    ... r = requests.get(requestUri)
    ... session['deezerAccessToken'] = r.json()['access_token']
    ... print(session.get('deezerAccessToken', None))
    ... return render_template('close.html')

@app.route('/getVKAccess', methods=['GET', 'POST'])
def get_vk_code():
    ... requestUri = 'https://oauth.vk.com/access_token?client_id=
    ... r = requests.get(requestUri)
    ... session['vkUserId'] = r.json()['user_id']
    ... print(session.get('vkUserId', None))
    ... return render_template('close.html')

```

Рисунок 22 – Фрагмент кода модуля routes.py

При вызове URL-адреса /transfer вызывается функция transfer_tracks, которая вызывает метод transfer_tracks из модуля VKtoDeezer.py. Данная функция запускает процесс переноса треков (рисунок 23).

```
def transfer_tracks(vk_user_id, deezer_access_token):
    get_tracks(vk_user_id, deezer_access_token)
    ids = get_found_tracks()
    playlist_id = str(create_playlist(deezer_access_token, 'fromVK'))
    add_tracks_to_playlist(deezer_access_token, playlist_id, ids)
    missing_tracks = get_missing_tracks()
    Track.query.delete()
    Playlist.query.delete()
    return missing_tracks
```

Рисунок 23 – Код функции transfer_tracks

Весь сценарий переноса сосредоточен в этой функции. В модуле VKtoDeezer.py реализованы функции взаимодействия с API «ВКонтакте» и API «Deezer». Они вызываются последовательно.

Первым идёт метод get_tracks, который занимается получением данных с последующим добавлением их в БД (рисунок 24).

```
def get_tracks(owner_Id, deezer_access_token):
    login = 'login'
    password = 'password'
    vk_session = vk_api.VkApi(login=login, password=password)
    try:
        vk_session.auth()
    except vk_api.AuthError as error_msg:
        print(error_msg)
    return

    vkaudio = VkAudio(vk_session)
    print(owner_Id)
    tracksInfo = vkaudio.get(owner_id=owner_Id)

    client = deezer.Client(app_id='352824',
                           app_secret='secret',
                           access_token=deezer_access_token)

    playlist = Playlist(name='main', count=len(tracksInfo), ownerId=owner_
```

Рисунок 24 – Фрагмент кода функции get_tracks

В этой функции происходит авторизация в социальной сети «ВКонтакте» с последующим запросом списка треков пользователя с owner_id. Затем каждый трек ищется в БД «Deezer», и при нахождении полю deezerId объекта Track

присваивается значение, по которому можно найти трек в БД «Deezer». Если не найдено ни одного совпадения, значение остаётся равным -1.

Для улучшения результатов поиска было принято решение очищать названия треков и имена исполнителей от «мусорных» символов. В модуле nameSplitter.py реализована функция clean_str, которая, находя вхождение определённой подстроки из словаря, удаляет это вхождение из строки (рисунок 25).

```
splitters = [
    'X',
    '[',
    '.FEAT',
    'FEAT',
    '#NR',
    '(OST',
    'FEAT.',
    'FT',
    '(FT',
    'PROD',
    '(PROD',
    '+',
    '(FEAT.',
    '/'
]

def clean_str(s):
    s = s.upper()
    for sp in splitters:
        if s.find(sp) != -1:
            s = s[0:s.find(sp):1]
    return s
```

Рисунок 25 – Код модуля nameSplitter.py

После получения данных вызывается функция get_found_tracks, которая составляет строковый список id найденных в БД «Deezer» треков без повторений (рисунок 26). Обязательное отсутствие повторений связано с тем, что плейлист в музыкальном сервисе «Deezer» должен содержать только уникальные треки, а аудиотека «ВКонтакте», в свою очередь, имеет достаточное количество повторений. Таким образом мы избавляемся от лишней информации и очищаем будущий плейлист.

```

def get_found_tracks():
    tracks = Track.query
    tracks = tracks.filter(Track.deezerId != -1)
    ids = ''
    for tr in tracks:
        if(ids.find(str(tr.deezerId))==-1):
            ids+=str(tr.deezerId)+','
    ids = ids[:-1]
    return ids

```

Рисунок 26 – Код функции get_found_tracks

Перед добавлением треков в плейлист, его сначала нужно создать. Для этого вызывается функция create_playlist, которая использует access token, полученный при авторизации (рисунок 27). Она создаёт плейлист и возвращает его id в БД «Deezer».

```

def create_playlist(access_token: str, playlist_name: str):
    return requests.post('http://api.deezer.com/user/me/playlists?access_token=' + access_token, {
        'title': playlist_name}).json()['id']

```

Рисунок 27 – Код функции create_playlist

После создания плейлиста вызывается функция add_tracks_to_playlist, которая принимает на входе access token от «Deezer», строковый список треков, составленный ранее, и id плейлиста в БД музыкального сервиса (рисунок 28).

```

def add_tracks_to_playlist(access_token: str, playlist_id: str, tracks: str):
    return requests.post('http://api.deezer.com/playlist/' + playlist_id + '/tracks?access_token=' + access_token, {
        'songs': tracks})

```

Рисунок 28 – Код функции add_tracks_to_playlist

При помощи единственного запроса к API «Deezer» все треки добавляются в плейлист. Последним этапом является вывод информации об отсутствующих в аудиотеке «Deezer» треков. Используя функцию get_missing_tracks, запрашиваем у БД список всех треков, и те, которые имеют поле deezerId равным -1, добавляем в словарь отсутствующих треков (рисунок 29).

Данный словарь отправляется на клиент, где дополнительно обрабатывается для большей наглядности.

```
def get_missing_tracks():
    tracks = Track.query
    tracks = tracks.filter(Track.deezerId == -1)
    missing_tracks = []
    for tr in tracks:
        missing_tracks.append({'artist': tr.artist, 'title': tr.title})
    return missing_tracks
```

Рисунок 29 – Код функции get_missing_tracks

3.3 Тестирование

Тестирование сервиса показало, что результаты переноса, представленные в таблице, составляют 75-80% от изначального количества треков.

Таблица – Результаты тестирования веб-сервиса

Начальное количество треков	Итоговое количество треков	Процент переноса
1043	813	77,9%
1466	1105	75,4%
804	650	80,8%
2512	1984	78,9%

Потери треков неизбежны, потому как библиотеки музыкальных композиций используемых сервисов отличаются. Также на потери треков влияют повторения треков в аудиотеке «ВКонтакте» и некорректные названия отдельных дорожек.

3.4 Выводы

В данной главе был освещён процесс разработки веб-сервиса. Итогом стало одностраничное веб-приложение, которое даёт пользователю возможность авторизоваться в социальной сети «ВКонтакте» и музыкальном сервисе «Deezer». Также была разработана серверная часть веб-сервиса с логикой приложения и необходимыми методами и проведено тестирование.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы был проведён анализ существующих решений для достижения поставленной цели. Среди имеющихся решений были выявлены следующие недостатки:

- отсутствие полноценного бесплатного решения;
- использование нестандартных методов авторизации пользователей;
- требовательные к действиям пользователя возможные способы решения.

Чтобы устранить перечисленные недостатки, было принято решение разработать веб-сервис, требующий от пользователя минимальное количество времени и личных данных.

В процессе разработки были решены все задачи, а именно:

- изучены доступные решения и их недостатки;
- найдены способы получения данных от социальной сети «ВКонтакте» посредством использования неофициального API vk_api;
- разработан веб-сайт для получения авторизационных данных от пользователя;
- разработана серверная часть для получения данных от раздела «музыка» социальной сети «ВКонтакте» с последующей обработкой и добавлением найденных треков в базе данных «Deezer».

Цель разработать веб-сервис для миграции музыкальных плейлистов из социальной сети «ВКонтакте» в потоковый музыкальный сервис «Deezer» достигнута. Использование данного программного продукта позволило снизить трудовые и временные затраты на перенос своей аудиотеки пользователей из социальной сети «ВКонтакте» в музыкальный потоковый сервис «Deezer».

Данная разработка в будущем может быть расширена добавлением взаимодействия с другими музыкальными сервисами. Во время выполнения выпускной квалификационной работы была решена самая важная проблема в данной области – это получение данных о музыкальных дорожках «ВКонтакте».

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1.Официальный сайт Trello [Электронный ресурс]: About | What is Trello?
– Режим доступа: <https://trello.com/about>
- 2.Официальный сайт Nodejs [Электронный ресурс]: About | Node.js –
Режим доступа: <https://nodejs.org/en/about/>
- 3.Нетология [Электронный ресурс]: Что такое Node.js и где он пригодится
– Режим доступа: <https://netology.ru/blog/node>
- 4.Википедия свободная энциклопедия [Электронный ресурс]: PHP –
Википедия – Режим доступа: <https://ru.wikipedia.org/wiki/PHP>
- 5.Официальный сайт PHP [Электронный ресурс PHP: Hypertext
Preprocessor – Режим доступа: <http://php.net/>
- 6.Web creator [Электронный ресурс]: Язык программирования Ruby –
«Веб Креатор» – Режим доступа: <https://web-creator.ru/articles/Ruby>
- 7.Официальный сайт Ruby on Rails [Электронный ресурс]: Ruby on Rails |
A web-application framework that includes everything needed to create database-
backed web applications according to the Model-View-Controller (MVC) pattern. –
Режим доступа: <https://rubyonrails.org/>
- 8.Официальный сайт Python [Электронный ресурс]: About Python |
Python.org – Режим доступа: <https://www.python.org/about/>
- 9.Python.su [Электронный ресурс]: Python – Wiki Портала-Python
программистов – Режим доступа: <http://wiki.python.su/Python/>
- 10.Официальный сайт Flask [Электронный ресурс]: Welcome to Flask –
Режим доступа: <http://flask.pocoo.org/docs/1.0/>
- 11.W3schools.com [Электронный ресурс]: HTML Tutorial – Режим
доступа: <https://www.w3schools.com/Html/>
- 12.Википедия свободная энциклопедия [Электронный ресурс]: HTML –
Википедия – Режим доступа: <https://ru.wikipedia.org/wiki/HTML>
- 13.MDN web docs [Электронный ресурс]: CSS: Cascading Style Sheets |
MDN – Режим доступа: <https://developer.mozilla.org/en-US/docs/Web/CSS>

14.Официальный сайт bootstrap [Электронный ресурс]: Bootstrap: About – Режим доступа: <https://getbootstrap.com/docs/4.1/about/overview/>

15.Современный учебник JavaScript [Электронный ресурс]: Введение в JavaScript – Режим доступа: <https://learn.javascript.ru/intro/>

16.Википедия свободная энциклопедия [Электронный ресурс]: Система управления версиями – Википедия – Режим доступа: https://ru.wikipedia.org/wiki/Система_управления_версиями

17.Хабр [Электронный ресурс]: Опрос по системам контроля версий / Хабр – Режим доступа: <https://habr.com/post/233935/>

18.Официальный сайт Git [Электронный ресурс]: Git – Режим доступа: <https://git-scm.com>

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
«Информатика»
кафедра

УТВЕРЖДАЮ

Заведующий кафедрой

А. С. Кузнецов

подпись инициалы фамилия

« 05 » 07 2019 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.04 «Программная инженерия»

код — наименование направления

Разработка веб-сервиса для миграции музыкальных плейлистов из социальной сети
«ВКонтакте» в потоковый музыкальный сервис «Deezer»

тема

Руководитель

А. Кузнецов 05/07/2019
подпись, дата

доцент, канд. техн. наук
должность, ученая степень

А. В. Хныкин
инициалы, фамилия

Выпускник

Мангиров 05/07/2019
подпись, дата

А. К. Мангиров
инициалы, фамилия

Нормоконтролер

О. А. Антамошкин 05.07.19
подпись, дата

доцент, канд. техн. наук
должность, ученая степень

О. А. Антамошкин
инициалы, фамилия

Красноярск 2019