

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

09.03.01 Информатика и вычислительная техника
код и наименование направления подготовки

ОТЧЕТ

по преддипломной практике
на кафедре прикладной математики и кибернетики
института информатики и вычислительной техники

Выполнил:

студент гр. ИП-014

«26» мая 2024 г.

_____ / Бессонов А.О. /

_____ /
оценка

Руководитель практики от университета

к.т.н., доцент каф. ПМиК

«26» мая 2024 г.

_____ / Ситняковская Е.И. /

Новосибирск 2024

План-график проведения производственной практики

Вид практики

Тип практики: преддипломная практика

Способ проведения практики: стационарная

Форма проведения практики: дискретно по периодам проведения практик

Выдано обучающемуся Бессонову Александру Олеговичу

Фамилия Имя Отчество студента

Направление: 09.03.01 – Информатика и вычислительная техника

Код – Наименование направления

Профиль: Программное обеспечение средств вычислительной техники и автоматизированных систем

Курс 4, группа ИП-014

Тема ВКР: Разработка мобильного приложения для прослушивания музыки с рекомендациями на основе нейронных сетей

Содержание практики

Наименование видов деятельности	Дата (начало – окончание)
<i>Постановка задачи на практику, определение конкретной индивидуальной темы, формирование плана работ</i>	22.04.2024 – 25.04.2024
<i>Работа с библиотечными фондами, сбор и анализ материалов по теме практики</i>	26.04.2024 – 29.04.2024
<i>Выполнение работ в соответствии с составленным планом:</i>	30.04.2024 – 21.05.2024
<i>1. Исследование предметной области</i>	30.04.2024 – 05.05.2024
<i>2. Определение требований к приложению</i>	06.05.2024 – 10.05.2024
<i>3. Выбор средств разработки</i>	11.05.2024 – 15.05.2024
<i>4. Составление технического описания приложения</i>	16.05.2024 – 21.05.2024
<i>Анализ полученных результатов и произведенной работы, составление отчета по практике</i>	22.05.2024 – 26.05.2024

Согласовано:

Руководитель практики от СибГУТИ _____ / Ситняковская Е.И. /

Содержание

Задание на преддипломную практику.....	4
Введение.....	5
1 Постановка задачи.....	6
1.1 Исследование предметной области.....	6
1.2 Изучение аналогичных продуктов	7
1.3 Цели и задачи разработки	11
2. Выбор средств разработки	13
2.1 Выбор операционной системы	13
2.2 Выбор технологий разработки.....	13
2.3 Выбор среды разработки	18
2.4 Выбор базы данных.....	21
3 Техническое описание приложения	23
3.1 Описание структуры приложения	23
3.1.1 Мобильный клиент	23
3.1.2 Сервер.....	24
3.1.3 Микросервис с нейросетью.....	25
3.2 Описание структуры базы данных	26
Заключение	31
Список использованных источников	33

Задание на преддипломную практику

Целью преддипломной практики является исследования в области создания мобильного приложения для прослушивания музыки с рекомендациями на основе нейронных сетей.

В рамках преддипломной практики ставятся следующие задачи:

1. Исследование предметной области.
2. Изучение существующих аналогичных продуктов.
3. Определение требований к разрабатываемому приложению.
4. Выбор средств разработки.
5. Составление технического описания приложения.

Введение

В современном мире цифровых технологий сложно представить повседневную жизнь без интернета и мобильного телефона, что всегда под рукой. Смартфон с доступом в интернет стал универсальным инструментом, который облегчает нашу жизнь.

С развитием мобильных технологий появилось огромное количество приложений для различных целей: от социальных сетей и онлайн-игр до образовательных платформ и инструментов для управления финансами. Каждое из них стремится удовлетворить потребности пользователей, предложить уникальный функционал и обеспечить удобство использования.

В контексте разнообразия особое внимание привлекают мобильные приложения связанные с прослушиванием музыки. Ведь музыка – это не только искусство, но и способ самовыражения, релаксации и вдохновения. Именно поэтому она играет важную роль в жизни многих людей, а удобство мобильных платформ только способствуют её популярности.

Однако с каждым днем появляется всё больше и больше новых музыкальных композиций, в том числе и от новых артистов, в следствии чего для пользователей становится всё сложнее найти новые песни, что им действительно понравятся. Для решения этой проблемы создаются различные системы рекомендаций, которые способны помочь слушателям открыть для себя произведения, соответствующие их вкусам и предпочтениям.

Такие системы все чаще основываются на нейронных сетях и методах машинного обучения. За счет анализа больших объемов данных и выявления скрытых закономерностей, они позволяют создавать сложные алгоритмы рекомендаций, которые учитывают не только предпочтения пользователей, но и особенности самих музыкальных треков. Это открывает новые возможности для создания персонализированных рекомендаций и улучшения пользовательского опыта.

Целью данной работы является разработка мобильного приложения для прослушивания музыки с рекомендациями на основе нейронных сетей. В этом приложении будет реализован функционал анализа музыкальных треков и предпочтений пользователей для создания персональных рекомендаций. Таким образом пользователи смогут легко находить новую музыку, которая соответствует их вкусам, что сделает процесс прослушивания более приятным и увлекательным.

1 Постановка задачи

1.1 Исследование предметной области

Количество обладателей смартфонов постоянно растет из года в год, за последние года рост составлял минимум 5% в год. Под конец 2023 года в мире насчитывалось порядка 6.84 миллиардов смартфонов в мире [1]. Эта тенденция создает благоприятные условия для развития мобильных приложений.

Основными операционными системами мобильных устройств на момент первого квартала 2024 года являются Android и iOS, которые занимают 70,71% и 28,54% мирового рынка смартфонов соответственно [2].

iOS — это мобильная операционная система, разработанная компанией Apple для своих устройств, таких как iPhone и iPad [3]. Впервые представленная в 2007 году, она стала известна своим интуитивно понятным интерфейсом, высокой производительностью и строгой системой безопасности. Несомненным плюсом iOS является её интеграция с другими устройствами и сервисами от Apple, что обеспечивает пользователям удобство использования и совместимость между гаджетами. Однако это же и является недостатком данной операционной системы, она совместима только с устройствами от своей же компании.

Операционная система Android, разработанная компанией Google, является одной из самых популярных платформ для мобильных устройств. Запущенная в 2008 году, она быстро завоевала значительную долю рынка благодаря своей открытости и широким возможностям кастомизации [4]. В отличие от iOS, Android можно встретить на мобильных устройствах от самых различных компаний, обеспечивая пользователям широкий выбор смартфонов. Но это приводит к фрагментации устройств и версий операционной системы, в следствии чего появляется различия в производительности и качестве пользовательского опыта, а открытая архитектура делает Android более уязвимой в плане безопасности.

В настоящее время существует большое множество мобильных приложений для прослушивания музыки, которые доступны на самых различных операционных системах, включая самые популярные из них - Android и iOS. Данные приложения предоставляют пользователям доступ к огромному каталогу музыкальных треков прямо со смартфонов. Основным функционалом этих приложений является воспроизведение музыки, что может быть достигнуто путем скачивания аудиофайлов на устройство или их потоковым вещанием с

сервера в реальном времени. Так же в музыкальных приложениях обычно предлагается широкий спектр возможностей, включая поиск музыкальных композиций и создание персонализированных плейлистов на основе предпочтений пользователя.

Рекомендации в мобильных приложениях для прослушивания музыки играют особую роль в улучшении пользовательского опыта, помогая людям открывать для себя новые треки и артистов, которые соответствуют их предпочтениям. Системы рекомендаций основаны на алгоритмах, которые анализируют музыкальные предпочтения пользователя, его поведение при прослушивании и другие различные факторы, чтобы предложить наиболее подходящие композиции [5].

Одним из самых распространённых подходов к составлению рекомендаций является анализ истории прослушивания. Приложения анализируют треки, которые пользователь слушал ранее, и на основе этой информации предлагает похожие музыкальные композиции по звучанию или по жанру. Этот метод позволяет учитывать индивидуальные пользовательские предпочтения в музыке.

Так же в мобильные приложения могут предложить пользователю давать свою оценку для музыкальных произведений, плейлистов, альбомов, артистов или жанров в целом. Что так же позволяет улучшить персонализированные рекомендации на основе личного мнения пользователя.

Нейронные сети и методы машинного обучения имеют ключевую роль в современных системах рекомендациях, так как позволяют анализировать больших объёмы информации и находить скрытые закономерности. Данные технологии могут быть использованы для анализа акустических характеристик музыкальных произведений. С помощью нейронных сетей можно обрабатывать звуковые данные и выделять особенности музыкальных произведений, такие как темп, настроение, инструментальная составляющая и вокал. На основе этих характеристик рекомендации пользователей становятся более точным, так как учитывают предпочтения пользователя в музыкальном звучании композиций.

1.2 Изучение аналогичных продуктов

«Яндекс Музыка» [6] – стриминговый сервис, созданный компанией «Яндекс», который позволяет слушать музыкальные композиции, их подборки,

альбомы и получать персональные рекомендации (рисунок 1.1). Он сейчас доступен через браузер, приложения для ПК на платформах Windows и MacOS, а так же мобильные приложения для iOS и Android.

«Яндекс» запустил отдельный сервис «Яндекс Музыка» 22 сентября 2010 года. А первое мобильное приложение вышло в мае 2012 года для операционной системы iOS, после чего спустя год в мае 2013 года вышло приложение для Android. По состоянию на 2023 год «Яндекс Музыка» насчитывает более 76 миллионов музыкальных треков.



Рисунок 1.1 – Логотип «Яндекс Музыка»

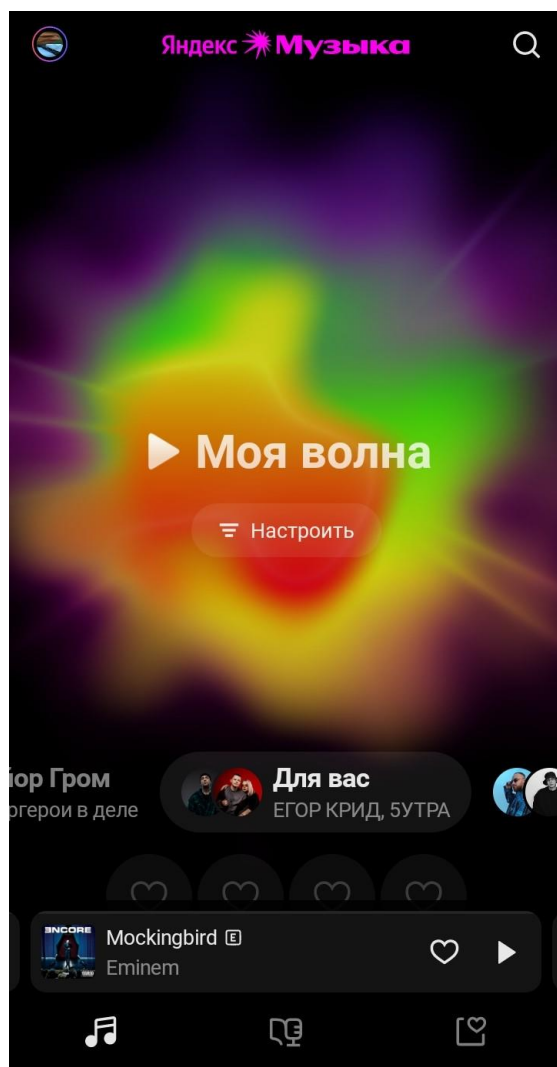
Для того чтобы пользоваться приложением «Яндекс Музыка» нужен аккаунт «Яндекс», а так же оформленная подписка «Яндекс Плюс». Однако у подписки есть бесплатный пробный период на 90 дней.

В мобильном приложении «Яндекс Музыка» есть три основных раздела в нижней части экрана: «Музыка», «Подкасты и книги» и «Коллекция» (рисунок 1.2).

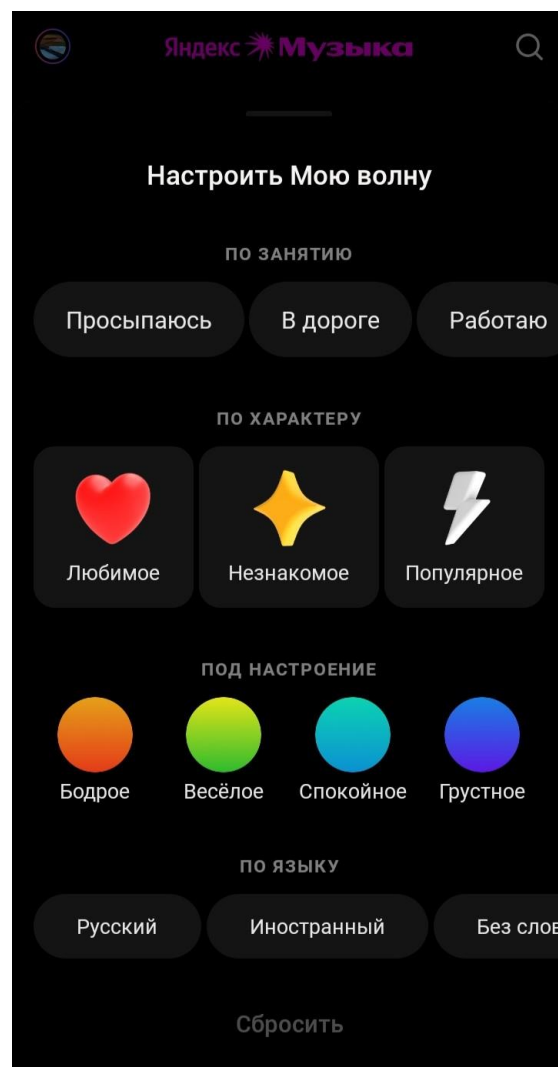
Музыкальные треки можно прослушивать в реальном времени, а так же скачивать их на мобильный телефон. В плеере приложения, есть возможность ставить на паузу и снимать с паузы, перематывать на нужное время, переключать на следующий или предыдущий трек в списке воспроизведения, а так же возможность ставить отметку «Нравится» или «Не нравится», при нажатии который песня либо добавится в коллекцию, либо будет убрана из рекомендаций навсегда соответственно. Так же есть возможность находить нужные треки, альбомы или артистов через кнопку поиска в приложении.

Ключевой особенностью «Яндекс Музыка» является система автоматизированных рекомендаций, которые работают в разделе «Моя волна» и плейлистах. Для более точных рекомендаций учитывается информация о пользователе: предыдущий слушательский опыт, лайки, дизлайки и жанровые предпочтения [7]. «Моя волна» — это потоковая функция данного сервиса, основанная на нейросети, которая постоянно подбирает для пользователя песню в соответствии с его предпочтениями на данный момент. Данную функцию можно настроить с помощью некоторых настроек, а именно указать для какого занятия нужна музыка, какой у нее должен быть характер, для какого настроения и на каком языке (рисунок 1.2).

Для первоначальной настройки рекомендаций новым пользователям предлагается указать их любимые жанры и несколько понравившихся артистов. Но данная настройка не является обязательной, поэтому если она не была пройдена, то пользователю рекомендуются самые популярную музыку на данный момент.



а)



б)

Рисунок 1.2 – Работа с приложением «Яндекс Музыка»: а) главная страница, б) меню настройки для «Моя волна»

«VK Музыка» [8] – стриминговый сервис, созданный компанией VK, который объединяет музыкальную платформу «ВКонтакте» и отдельное приложение для прослушивания музыки (рисунок 1.3). Ранее это были два отдельных сервиса BOOM и «Музыка ВКонтакте», но в ноября 2021 года они были объединены в один сервис. Сейчас «VK Музыка» доступна через веб-

интерфейс на сайте социальной сети «ВКонтакте» и мобильные приложения для iOS и Android.

Свое начало оригинальный сервис «Музыка ВКонтакте» берет ещё с создания социальной сети «ВКонтакте» в октябре 2006 года. За это время сервис сильно преобразился и расширил свой функционал, в том числе и развивалась система рекомендаций. В апреле 2022 года в «VK Музыка» появился новый алгоритм, который для каждого слушателя специальную подборку «Слушайте друг друга» с публичными плейлистами от людей и сообществ с похожим музыкальным вкусом.



Рисунок 1.3 – Логотип «VK Музыка»

Для того, чтобы пользоваться приложением «VK Музыка» нужно иметь аккаунт «VK». В отличие от «Яндекс Музыка» тут не нужно иметь платную подписку для использования сервиса. Однако при бесплатном использовании перед прослушиванием музыкальной композиции периодически будет появляться аудиореклама.

В мобильном приложении «VK Музыка» есть пять основных разделов в нижней части экрана: «Для вас», «Обзор», «Книги и шоу», «Поиск» и «Моя музыка» (рисунок 1.4).

Музыкальный плеер «VK Музыка» имеет схожий функционал с плеером в «Яндекс Музыка», однако есть отличие, которое заключается в кнопке «Добавить в мою музыку» вместо кнопки «Нравится».

В разделе «Для вас» расположены две персонализированные подборки для пользователей: «Рекомендации» и «VK Mix». Именно они являются основной системой рекомендаций в приложении.

«VK Mix» — это бесконечный плейлист, который подстраивается под музыкальные вкусы пользователя и обновляется каждый день. Рекомендации подстраиваются под жанровые предпочтения, историю прослушивания и возможные музыкальные симпатии [9]. Однако в отличие от «Моя волна» в «Яндекс Музыка» пользователь никак не может настроить данный плейлист, но в мобильном приложении существуют плейлисты по настроению, что отчасти решают эту проблему.

«Рекомендации» являются частью нового алгоритма рекомендаций с плейлистами от людей с похожими вкусами [10]. Однако стоит заметить, что в разделе «Рекомендации» так же присутствуют официальные подборки от VK, альбомы, миксы по трекам и артистам. В данном разделе пользователь может прослушивать снippets из предложенных подборок, то есть ему будут предлагаться самые яркие фрагменты трека для того, чтобы он смог принять решение слушать полностью подборку или нет.

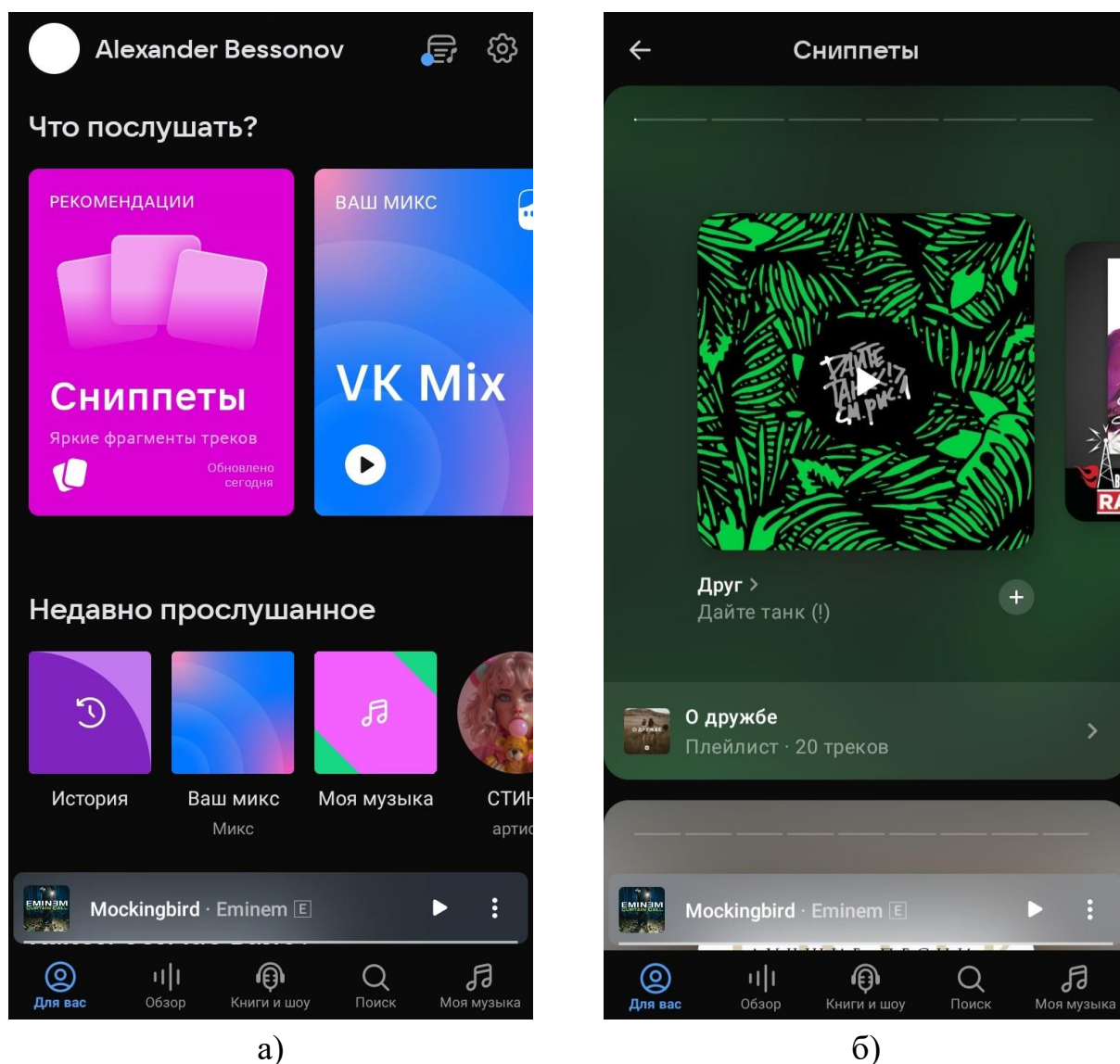


Рисунок 1.4 – Работа с приложением «VK Музыка»: а) главная страница, б) просмотр раздела «Сниппеты»

1.3 Цели и задачи разработки

Целью разработки является создание мобильного приложения для прослушивания музыки с рекомендациями на основе нейронных сетей.

Особенностью этого мобильного приложения будет возможность создавать сборник треков на основе рекомендаций от нейросети с учетом музыкальных предпочтений. Пользователь так же сможет настраивать генерацию данных плейлистов по своему желанию и добавлять к себе понравившиеся.

Перед разработкой приложения необходимо определиться с требованиями к приложению, которые будут задачами разработки. На основе проведенного исследования предметной области и анализе существующих музыкальных приложений, которые предоставляют своим пользователям персональные рекомендации, можно определить следующие требования.

Основные функциональные требования к приложению:

- Воспроизведение музыки;
- Скачивание музыки;
- Поиск музыки по названию;
- Создание и управления плейлистами;
- Добавление в избранное музыкальных треков, альбомов и артистов;
- Получение персонализированных рекомендаций с учетом пожеланий пользователя;
- Возможность оценивать музыкальные треки;
- Создание учетной записи пользователя;
- Авторизация в учетную запись;
- Возможность изменения данных учетной записи.

Интерфейс приложения должен быть интуитивно понятен пользователю, обеспечивая простоту использования.

На основе данных требований лучшим вариантом для реализации будет клиент-серверная структура. Для возможности в будущем легко расширять функционал приложения серверная часть будет состоять из главного сервера и микросервисов. В данном случае будет один микросервис, отвечающий за составления рекомендации для пользователей с помощью нейронных сетей.

Для взаимодействий между компонентами структуры приложения будет использован RESTful API. В базах данных же будет храниться необходимая для работы приложения информация.

2. Выбор средств разработки

2.1 Выбор операционной системы

Для разработки мобильного приложения необходимо определиться с целевой операционной системой мобильных устройств. Согласно статистике с сайта Statista в первом квартале 2024 года абсолютное большинство на всемирном рынке смартфонов занимает Android [2]. Однако существуют страны, в которых iOS занимает лидирующую позицию по пользователям. Но это не относится к России, у нас сохранена мировая тенденция на доминирование операционной системы Android на рынке мобильных телефонов.

В связи с широкой распространённостью и доступностью устройств на базе Android, именно эта операционная система была выбрана в качестве целевой, и разработка приложения будет именно для неё.

Android [4] – это операционная система, разработанная компанией Google, которая широко используется на различных устройствах по всему миру (рисунок 2.1). Она основана на ядре Linux и собственной реализации виртуальной машины Java компании Google. Android имеет открытый исходный код, что позволяет разработчикам адаптировать систему под свои нужды и вносить необходимые изменения. Эта гибкость одно из ключевых преимуществ, так как она позволяет создавать приложения с разнообразными функциями и настройками.



Рисунок 2.1 – Логотип Android

2.2 Выбор технологий разработки

Мобильный клиент на Android

При разработке мобильных приложений под операционную систему Android обычно выбирают один из двух языков: Java и Kotlin. В качестве основного языка программирования для мобильного клиента приложения был выбран Kotlin.

Kotlin [11] – это статически типизированный, объектно-ориентированный язык программирования, работающий поверх Java Virtual Machine и разрабатываемый компанией JetBrains (рисунок 2.2). Его создание началось в

2010 году, а первый релиз состоялся в феврале 2016 года. Название языка происходит от острова Котлин в Финском заливе, который находится недалеко от Санкт-Петербурга, где расположен один из офисов JetBrains.

Основной целью разработки Kotlin было создание языка, который был бы более выразительным, лаконичным и безопасным, чем Java, и более простой, чем Scala. При этом язык полностью совместимым с существующей экосистемой JVM (Java Virtual Machine).

На конференции для разработчиков I/O 2019 Google объявила, что язык программирования Kotlin теперь является предпочтительным языком для разработчиков приложений под Android.



Рисунок 2.2 – Логотип Kotlin

Преимущества использования Kotlin:

- Удобство использования: Более лаконичный и выразительный синтаксис делает код на Kotlin легче для понимания и поддержки. Позволяет писать меньше кода по сравнению с Java.
- Безопасность: Улучшенная система типов и встроенные механизмы для обработки null-значений значительно снижают вероятность ошибок.
- Совместимость с Java: Kotlin полностью совместим с Java, что позволяет использовать существующий Java код, вместе с библиотеками и фреймворками на языке Java.
- Поддержка современных возможностей: Такие функции, как корутины и DSL, позволяют разрабатывать более эффективные и масштабируемые приложения.

Серверная часть приложения

В качестве основного языка программирования для серверной части приложения был выбран Java с фреймворком Spring Boot.

Java [12] – строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией Sun Microsystems, которую в последующем приобрела Oracle (рисунок 2.3). Был

разработан Джеймсом Гослингом с его командой в 1995 году как язык для программирования бытовых приборов, но позже был переориентирован на создание веб-приложений и корпоративных систем.



Рисунок 2.3 – Логотип Java

Преимущества использования Java:

- **Широкое использование:** Java активно используется в разработке корпоративных приложений, Android-приложений и веб-сервисов
- **Масштабируемость:** Java подходит для создания крупных распределенных систем и высоконагруженных серверных приложений.
- **Производительность:** Несмотря на то, что Java – интерпретируемый язык, современные JVM и JIT-компиляция обеспечивают высокую производительность.
- **Платформенная независимость:** Принцип «Write Once, Run Anywhere» позволяет запускать Java-приложения на любых устройствах с JVM.
- **Обширная экосистема:** Java обладает богатой экосистемой библиотек, фреймворков и инструментов, что ускоряет разработку и улучшает качество приложений.
- **Безопасность:** Встроенные функции безопасности и регулярные обновления делают Java надежным выбором для разработки критически важных приложений.

Spring Boot [13] — это фреймворк для разработки на языке Java, созданный компанией Pivotal (теперь VMware) и впервые выпущенный в апреле 2014 года (рисунок 2.4). Spring Boot является частью Spring Framework и направленный на упрощение работы с данным фреймворком.



Рисунок 2.4 – Логотип «Spring Framework»

Преимущества использования Spring Boot:

- Быстрый старт: Благодаря автоконфигурации и Spring Initializr, разработчики могут быстро начать новый проект и сконцентрироваться на бизнес-логике.
- Снижение количества конфигурационного кода: Автоконфигурация и умные стандартные настройки уменьшают необходимость написания большого объема конфигурационного кода.
- Готовые к использованию приложения: Встроенные серверы и автономность позволяют легко создавать и развертывать приложения.
- Обширная экосистема: Полная интеграция с экосистемой Spring обеспечивает доступ ко множеству модулей и библиотек, таких как Spring Data, Spring Security, Spring Cloud и другие.
- Микросервисная архитектура: Отлично подходит для создания микросервисов благодаря своей легкости и модульности.
- Упрощенное тестирование и мониторинг: Инструменты для тестирования и мониторинга облегчают обеспечение качества и поддержание приложения в рабочем состоянии.

Java вместе с фреймворком Spring Boot отлично подходят для разработки масштабируемых приложений, благодаря автоматической настройке Spring Boot ускоряется темп разработки. Так же за счёт полной интеграции с экосистемой Spring можно использовать Spring Security для безопасности сервера.

Микросервис с нейросетью

В качестве основного языка программирования для микросервиса с нейронными сетями был выбран Python с фреймворком FastAPI.

Python [14] – это высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, созданный Гвидо ван Россум и впервые выпущенным в феврале 1991 года (рисунок 2.5). Язык был разработан с ориентиром на простоту и читаемость кода, что повышает производительность разработчиков.



Рисунок 2.5 – Логотип Python

Преимущества использования Python:

- **Удобство и скорость разработки:** Простота синтаксиса и доступность множества библиотек позволяют быстро разрабатывать и прототипировать приложения.
- **Широкое применение:** Python используется в самых различных областях, включая анализ данных, искусственный интеллект и веб-разработку.
- **Кроссплатформенность:** Python работает на всех основных операционных системах, включая Windows, macOS и Linux.
- **Большое сообщество и экосистема:** Python имеет большое сообщество разработчиков, что приводит к широкому выбору библиотек и фреймворков для самых разных задач, таких как Django, Flask, Pandas, NumPy, TensorFlow и многие другие.

FastAPI [15] — это веб-фреймворк для создания API на Python, разработанный Себастьяном Рамиресом и впервые выпущенный в 2018 году (рисунок 2.6). В основе FastAPI лежат две библиотеки — Starlette (ASGI-фреймворк) и Pydantic (для описания схем данных). Целью создания данного фреймворка была в предоставлении простого в использовании инструмента для создания быстрого API.



Рисунок 2.6 – Логотип «FastAPI»

Преимущества использования FastAPI:

- **Высокая производительность:** Использование асинхронных возможностей Python делает FastAPI одним из самых производительных веб-фреймворков.

- **Скорость разработки:** Благодаря автоматической валидации данных и генерации документации, разработчики могут быстрее создавать и поддерживать API.

- **Современные стандарты:** Поддержка современных стандартов API, таких как OpenAPI и JSON Schema, облегчает интеграцию с другими сервисами и инструментами.

- **Безопасность:** Встроенная поддержка аутентификации и авторизации, включая OAuth2 и JWT.

Python с использованием фреймворка FastAPI позволяют создать высокопроизводительное приложение с использованием API. Обширный выбор библиотек для Python облегчает анализ аудиофайлов и обучение нейронных сетей.

2.3 Выбор среды разработки

Мобильный клиент на Android

В качестве среды разработки для мобильного клиента на Android была выбрана Android Studio.

Android Studio [16] — это официальная интегрированная среда разработки (IDE) для создания приложений под операционную систему Android, разработанная Google в 2013 году (рисунок 2.7). Она основана на платформе IntelliJ IDEA от JetBrains и заменила Eclipse Android Development Tools (ADT) в качестве основного инструмента для разработки приложений под Android. Официально поддерживает следующие языки программирования: Kotlin, Java и C++.



Рисунок 2.7 – Логотип «Android Studio»

Преимущества использования Android Studio

- **Официальная поддержка от Google:** Как официальная IDE для разработки под Android, Android Studio получает регулярные обновления и поддержку от Google, что обеспечивает актуальность и совместимость с последними версиями Android.
- **Поддержка Kotlin:** Полная поддержка языка программирования Kotlin, который является официальным языком для разработки Android-приложений наряду с Java.
- **Современные инструменты разработки:** Включает современные инструменты для визуального редактирования интерфейсов, отладки и тестирования, что упрощает процесс разработки.
- **Мощная система сборки:** Gradle обеспечивает гибкость и мощь в управлении зависимостями и процессами сборки, что важно для крупных проектов.
- **Многофункциональный эмулятор:** Эмулятор Android позволяет разработчикам тестировать свои приложения на различных устройствах и версиях Android без необходимости иметь физические устройства.

Серверная часть приложения

В качестве среды разработки для серверной части приложения была выбрана IntelliJ IDEA.

IntelliJ IDEA [17] — это интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, разработанная компанией JetBrains в январе 2001 года (рисунок 2.8). Основной целью создания IDE было предоставление мощного и интеллектуального инструмента, который бы повысил продуктивность разработчиков благодаря широкому набору функций и продвинутой поддержке кода.



Рисунок 2.8 – Логотип «IntelliJ IDEA»

Преимущества IntelliJ IDEA

- Интеграция с Spring Boot: IntelliJ IDEA предоставляет специальные инструменты и поддержку для разработки приложений с использованием Spring Boot. Это включает автоконфигурацию, поддержку аннотаций Spring, автоматическую генерацию конфигурационных файлов и многое другое.
- Умный редактор кода: IntelliJ IDEA предоставляет мощные функции автодополнения, анализа кода и рефакторинга, которые помогают разработчикам писать чистый и эффективный код.
- Поддержка тестирования: Интеграция с фреймворками для тестирования, такими как JUnit и TestNG, позволяет легко писать и запускать тесты.
- Управление зависимостями: Легкость в управлении зависимостями и конфигурациями сборки с использованием Maven и Gradle, что является стандартом для проектов Spring Boot.

Микросервис с нейросетью

В качестве среды разработки для микросервиса с нейронными сетями был выбрана PyCharm.

PyCharm [18] — это кроссплатформенная интегрированная среда разработки для языка программирования Python, разработанная компанией JetBrains на основе IntelliJ IDEA в феврале 2010 года (рисунок 2.9). Она была разработана с целью предоставить мощный и удобный инструмент для разработчиков, предлагая широкий набор функций для повышения продуктивности и качества кода.



Рисунок 2.9 – Логотип PyCharm

Преимущества PyCharm:

- Мощные инструменты для разработки: Умный редактор кода, мощные средства рефакторинга и анализа кода значительно ускоряют процесс разработки.
- Широкая поддержка экосистемы Python: Поддержка различных фреймворков, библиотек и инструментов делает PyCharm универсальным решением для разработки на Python в самых различных областях.
- Управление зависимостями: Поддержка виртуальных окружений позволяет легко управлять зависимостями проекта, что особенно полезно при работе с большими проектами или при разработке нескольких проектов одновременно.

2.4 Выбор базы данных

В качестве базы данных для приложения была выбрана база данных PostgreSQL.

PostgreSQL [19] — это свободная объектно-реляционная система управления базами данных с открытым исходным кодом (рисунок 2.10). Создана в 1986 году профессором Майклом Стоунбрейкером как преемник системы управления базами данных INGRES и имел целью реализовать новые идеи в области баз данных. PostgreSQL базируется на языке SQL и поддерживает многие из возможностей стандарта SQL:2011 и ряд возможностей SQL:2016 в части работы с данными в формате JSON. PostgreSQL является одной из наиболее популярных и надежных систем управления базами данных в мире.



Рисунок 2.10 – Логотип PostgreSQL

Преимущества использования PostgreSQL:

- **Надежность и стабильность:** PostgreSQL широко известен своей надежностью, стабильностью и возможностью работы в промышленных условиях.
- **Гибкость и расширяемость:** Возможность расширения функциональности базы данных с помощью пользовательских типов данных, функций и расширений делает PostgreSQL чрезвычайно гибким.
- **Безопасность:** PostgreSQL предоставляет развитые механизмы управления доступом и безопасности, включая контроль доступа на уровне строк (RLS) и шифрование данных.
- **Высокая производительность:** Эффективная обработка сложных запросов и параллельная работа с данными благодаря использованию MVCC и различных типов индексов.
- **Совместимость с современными стандартами:** Поддержка актуальных стандартов SQL и возможность работы с современными форматами данных, такими как JSON и XML.

3 Техническое описание приложения

3.1 Описание структуры приложения

В результате исследования предметной области была поставлена цель разработать мобильное приложение, которое будет состоять из трех основных компонентов: мобильного клиента, сервера и микросервиса с нейросетью. Каждый из этих компонентов играет важную роль в обеспечении функциональности и взаимодействия системы.

3.1.1 Мобильный клиент

Клиентское приложение на Android представляет собой пользовательский интерфейс, через который пользователи взаимодействуют с музыкальным контентом и получают персонализированные рекомендации. Определим, что из себя будет представлять мобильный клиент.

При запуске приложения появляется экран с авторизацией в учетную запись пользователя. На данном экране имеется кнопка регистрации учетной записи, при нажатии которой приложении переходит на экран регистрации.

На экране регистрации новому пользователю надо будет заполнить регистрационную форму: имя пользователя, адрес электронной почты и пароль. После заполнения пользователь нажимает кнопку регистрации, тем самым отправляя данные на сервер.

В случае успешной авторизации после входа или регистрации пользователь попадает на главный экран приложения. Внизу экрана есть имеется меню с тремя вкладками основных разделов приложения: «Главная», «Поиск» и «Моя музыка». Вверху имя пользователя, при нажатии на которое открывает экран с профилем пользователя.

В разделе «Главная» имеются секции под жанры музыки, артистов, альбомы, а так же секция с персональными рекомендациями, где пользователь так же может открыть форму с созданием запроса для получения персональной рекомендации с указанием количества треков, возможных жанров, а так же с процентом уже знакомых треков.

В разделе «Поиск» имеется поля поиска с возможностью ввода туда текста пользователем, после чего внизу отображаются результаты поиска по названию среди артистов, альбомов и треков. Если текстовое поле пустое внизу под этим

полем имеется секция «История прослушивания» со списком прослушанных треков.

В разделе «Моя музыка» есть секции для добавленных в личную коллекцию артистов, альбомов, плейлистов и треков. В секции плейлистов имеется возможность создавать новый плейлист или редактировать существующие, менять названия плейлистам или их содержимое.

В профиле пользователя указан его логин, имя, адрес электронной почты, дата регистрации и последнее изменение данных аккаунта. Так же имеется возможность поменять имя и пароль на новые после заполнения соответствующих форм.

При нажатии на трек он должен начать воспроизводиться в плеере приложения, в котором можно поставить на паузу и снять с нее, перемотать время в треке, переключиться на следующий или предыдущий, поставить свою оценку, добавить в свою коллекцию.

3.1.2 Сервер

Серверная часть приложения отвечает за обработку запросов от клиентского приложения, управления данными и взаимодействие с микросервисом.

Основные модули сервера:

1. Модуль авторизации и регистрации пользователей.
2. Модуль управления музыкальным каталогом.
3. Модуль обработки пользовательских запросов.
4. Модуль взаимодействия с микросервисом.

Сервер предоставляет RESTful API для взаимодействия с клиентским приложением. Запросы и ответы передаются в формате JSON.

RESTful API (Representational State Transfer Application Programming Interface) — это архитектурный стиль для разработки веб-сервисов, который использует стандартные HTTP методы (GET, POST, PUT, DELETE) для взаимодействия между клиентом и сервером.

Основные эндпоинты сервера:

- «user/login» - вход в учетную запись.
- «user/register» - регистрация учетной записи.
- «user/edit» - изменение данных учетной записи.
- «user/profile» - получения профиля учетной записи.
- «track/**» - взаимодействие с треками.
- «album/**» - взаимодействие с альбомами.
- «artist/**» - взаимодействие с артистами.
- «playlist/**» - взаимодействие с плейлистами.
- «recommendation/**» - взаимодействие с рекомендациями.
- «search/**» - поиск результатов по названию.

Стоит заметить, что все эндпоинты, кроме «user/login» и «user/register», являются не публичными, для доступа к ним нужен JWT токен, получаемый после авторизации в учетную запись.

JWT (JSON Web Token) — это компактный, URL-безопасный токен, который используется для передачи информации между сторонами в виде JSON-объекта. Он чаще всего используется для аутентификации и авторизации. JWT состоит из трех частей, разделенных точками: заголовка, полезной нагрузки и подписи.

Благодаря JWT токену сервер может понимать, кто является отправителем запроса, что позволяет повысить безопасность приложения. Преимуществом данного подхода также является то, что сам токен после создания на сервере и передаче клиент после авторизации не хранится на самом сервере.

3.1.3 Микросервис с нейросетью

Микросервисная архитектура позволяет более легко масштабировать приложения и улучшать их производительность с устойчивостью. Поэтому было

целесообразно разделить серверную часть приложения, ответственную за рекомендации на основе нейронных сетей, в отдельный микросервис.

Микросервис предоставляет RESTful API, для того чтобы сервер мог отправлять запросы, связанные с рекомендациями пользователей. Доступ к API ограничен для всех, кроме сервера, для избежания возможных проблем.

Данный микросервис имеет возможность получать аудио признаки музыкальных файлов разными методами с помощью соответствующих библиотек. На основе пользовательского мнения у каждого трека формируется свой рейтинг, в который учитывается оценка трека, оценка артиста этого трека, оценка альбома с этим треком, оценка плейлиста с этим треком, оценка жанра этого трека, добавлен ли трек в «Моя музыка», был ли трек недавно прослушан.

На основе полученной информации обучается модель нейронной сети, после чего она используется для составления рекомендаций для этого пользователя.

3.2 Описание структуры базы данных

Для хранения всей необходимой информации для функционирования приложения была разработана структура базы данных. Всего в базе данных насчитывается 29 таблиц:

- album – информация о альбоме.
 - id – integer (главный ключ)
 - name – text
 - release_date – date
 - image_filename – text
- album_artist – информация о взаимосвязи альбомов и артистов.
 - album_id – integer (главный ключ) (внешний ключ к album)
 - artist_id – integer (главный ключ) (внешний ключ к artist)
 - artist_status_id – integer (внешний ключ к artist_status)
- album_track – информация о взаимосвязи альбомов и треков.
 - album_id – integer (главный ключ) (внешний ключ к album)
 - track_id – integer (главный ключ) (внешний ключ к track)
 - track_number_in_album – integer

- **artist** – информация о артистах.
 id – integer (главный ключ)
 name – text
 image_filename – text
- **artist_status** – информация о статусе артиста в альбоме/треке
 id – integer (главный ключ)
 name – text
- **artist_track** – информация о взаимосвязи артистов и треков.
 album_id – integer (главный ключ) (внешний ключ к album)
 track_id – integer (главный ключ) (внешний ключ к track)
 artist_status_id – integer (внешний ключ к artist_status)
- **genre** – информация о жанрах музыки.
 id – integer (главный ключ)
 name – text
- **license** – информация о лицензиях на музыку.
 id – integer (главный ключ)
 name – text
- **playlist** – информация о плейлистах.
 id – integer (главный ключ)
 name – text
 creation_date – date
 last_update_date – date
- **playlist_track** – информация о взаимосвязи плейлистов и треков.
 playlist_id – integer (главный ключ) (внешний ключ к playlist)
 track_id – integer (главный ключ) (внешний ключ к track)
 track_number_in_playlist – integer
- **track** – информация о треке.
 id – integer (главный ключ)
 primary_genre_id – integer (внешний ключ к genre)
 license_id – integer (внешний ключ к license)
 name – text
 duration_in_seconds – integer
 release_date – date
 audio_filename – text

- **track_audio_feature** – информация о аудио признаках треков.
 id – integer (главный ключ)
 track_id – integer (внешний ключ к track)
 track_audio_feature_extraction_type_id – integer (внешний ключ к track_audio_feature_extraction_type)
- **track_audio_feature_extraction_type** – информация о способах получения аудио признаков треков.
 id – integer (главный ключ)
 start_delta – integer
 segment_duration – integer
- **track_segment_audio_feature** – информация о аудио признаках сегментов треков.
 id – integer (главный ключ)
 track_audio_feature_id – integer (внешний ключ к track_audio_feature)
 segment_number – integer
 data – float[]
- **user_album** – информация о добавленных альбомах пользователями в свою личную коллекцию.
 user_id – integer (главный ключ) (внешний ключ к user_data)
 album_id – integer (главный ключ) (внешний ключ к album)
 album_number_in_user_list – integer
 added_date – date
- **user_album_rating** – информация о оценке альбомов пользователями.
 user_id – integer (главный ключ) (внешний ключ к user_data)
 album_id – integer (главный ключ) (внешний ключ к album)
 user_rating_id – integer (внешний ключ к user_rating)
- **user_artist** – информация о добавленных артистов пользователями в свою личную коллекцию.
 user_id – integer (главный ключ) (внешний ключ к user_data)
 artist_id – integer (главный ключ) (внешний ключ к artist)
 artist_number_in_user_list – integer
 added_date – date
- **user_artist_rating** – информация о оценке артистов пользователями.
 user_id – integer (главный ключ) (внешний ключ к user_data)
 artist_id – integer (главный ключ) (внешний ключ к artist)

user_rating_id – integer (внешний ключ к user_rating)

- user_data – информация о пользователях.

id – integer (главный ключ)

username – text

email – text

password – text

nickname – text

registration_date – date

last_update_date – date

- user_neural_network_configuration – информация о конфигурациях нейронных сетей для составления рекомендаций пользователей.

id – integer (главный ключ)

user_id – integer (внешний ключ к user_data)

track_audio_feature_extraction_type_id – integer (внешний ключ к track_audio_feature_extraction_type)

training_date – date

model_config – jsonb

model_weights – bytea

- user_playlist – информация о добавленных плейлистах пользователями в свою личную коллекцию.

user_id – integer (главный ключ) (внешний ключ к user_data)

playlist_id – integer (главный ключ) (внешний ключ к playlist)

access_level_id – integer (внешний ключ к access_level)

playlist_number_in_user_list – integer

added_date – date

- user_playlist_access_level – информация о уровнях доступа пользователей к плейлисту.

id – integer (главный ключ)

name – text

- user_playlist_rating – информация о оценке плейлистов пользователями.

user_id – integer (главный ключ) (внешний ключ к user_data)

playlist_id – integer (главный ключ) (внешний ключ к playlist)

user_rating_id – integer (внешний ключ к user_rating)

- user_rating – информация о возможных оценках пользователей.

id – integer (главный ключ)

name – text

- user_recommendation – информация о персональных рекомендациях пользователей.

- id – integer (главный ключ)

- user_id – integer (внешний ключ к user_data)

- user_neural_network_configuration_id – integer (внешний ключ к user_neural_network_configuration)

- user_rating_id – integer (внешний ключ к user_rating)

- creation_date – date

- user_recommendation_track – информация о взаимосвязи рекомендаций и треков.

- user_recommendation_id – integer (главный ключ) (внешний ключ к user_recommendation)

- track_id – integer (главный ключ) (внешний ключ к playlist)

- track_number_in_list – integer

- user_track – информация о добавленных треках пользователями в свою личную коллекцию

- user_id – integer (главный ключ) (внешний ключ к user_data)

- track_id – integer (главный ключ) (внешний ключ к track)

- track_number_in_user_list – integer

- added_date – date

- user_track_history – информация о прослушанных треках пользователями.

- user_id – integer (главный ключ) (внешний ключ к user_data)

- track_id – integer (главный ключ) (внешний ключ к track)

- listen_date – date

- user_track_rating – информация о оценке треков пользователями.

- user_id – integer (главный ключ) (внешний ключ к user_data)

- track_id – integer (главный ключ) (внешний ключ к track)

- user_rating_id – integer (внешний ключ к user_rating)

Заключение

В ходе преддипломной практики по разработке мобильного приложения для прослушивания музыки с рекомендациями на основе нейронных сетей были проведены соответствующие плану работы: исследование предметной области, анализ существующих аналогов, определения требований к приложению, выбор средств разработки и составления технического описания приложения.

Во время исследования предметной области была замечена всемирная тенденция на популяризацию смартфонов, что благоприятно влияет на развития мобильных приложений для них. Так же были рассмотрены основные операционные системы для мобильных устройств, а именно Android и iOS. Было проведено изучение мобильных приложений для прослушивания музыки, в ходе которого внимание уделялась функциональности приложения и различным методам построения рекомендаций, в том числе и основанных на нейронных сетях и методах машинного обучения.

Для анализа существующих аналогов были взяты «Яндекс Музыка» и «VK Музыка». Во время разбора данных музыкальных сервисов делался акцент на функциональность и интерфейс их мобильных приложений, а так же на предлагаемую пользователю систему рекомендаций.

По итогам исследования предметной области и анализа существующих решений были составлены требования к мобильному приложению для прослушивания музыки с рекомендациями на основе нейронных сетей, а так же была сформулирована цель разработки данного приложения.

Для реализации мобильного приложения были выбраны средства разработки и рассмотрены преимущества данных решений. В качестве операционной системы для мобильного приложения был выбран Android, а основными технологиями разработки являются: Kotlin для мобильного клиента, Java с Spring Boot для серверной части приложения, Python с FastAPI для микросервиса с нейросетью. Базой данных в этом проекте будет PostgreSQL.

Было составлено техническое описание приложения, в котором были описаны компоненты приложения и их функциональность, а так же реализована структура базы данных приложения.

В результате проведенной работы в рамках преддипломной практики были сформированы требования к приложению, полученные благодаря исследованию предметной области и анализу существующих решений. На основе цели и задачи

проекта были выбраны подходящие средства разработки для дальнейшей реализации приложения. После чего было составлено техническое описание приложения, в соответствии с которым проект будет разрабатываться в будущем.

Список использованных источников

1. Howarth J. How Many People Own Smartphones? (2024-2029) [Электронный ресурс] // Exploding Topics [сайт]. [2023]. URL: <https://explodingtopics.com/blog/smartphone-stats/> (дата обращения: 31.04.2024).
2. Sherif A. Market share of mobile operating systems worldwide from 2009 to 2024, by quarter [Электронный ресурс] // Statista [сайт]. [2024]. URL: <https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/> (дата обращения: 02.05.2024).
3. iOS [Электронный ресурс] // Википедия : Свободная энциклопедия [сайт]. URL: <https://ru.wikipedia.org/wiki/IOS> (дата обращения: 02.05.2024).
4. Android [Электронный ресурс] // Википедия : Свободная энциклопедия [сайт]. URL: <https://ru.wikipedia.org/wiki/Android> (дата обращения: 02.05.2024).
5. Как работают рекомендательные системы музыкальных сервисов [Электронный ресурс] // РБК Тренды [сайт]. URL: <https://trends.rbc.ru/trends/industry/5f2c271c9a7947ead6c655f3> (дата обращения: 03.05.2024).
6. Официальный сайт «Яндекс Музыка» [Электронный ресурс]. URL: <https://music.yandex.ru/> (дата обращения: 06.05.2024).
7. Гарькуша И. Как устроены рекомендации «Яндекс.Музыки» [Электронный ресурс]. // Институт музыкальных инициатив [сайт]. [2021]. URL: <https://i-m-i.ru/post/how-does-yandex-music-work> (дата обращения: 06.05.2024).
8. Официальный сайт «VK Музыка» [Электронный ресурс]. URL: <https://music.vk.com/> (дата обращения: 08.05.2024).
9. ВКонтакте обновила музыкальные рекомендации [Электронный ресурс] // ВКонтакте [сайт]. [2020]. URL: <https://vk.com/press/music-recommendations> (дата обращения: 09.05.2024).
10. Слушайте друг друга: VK Музыка запускает новый алгоритм рекомендаций с плейлистами от людей с похожими вкусами [Электронный ресурс] // ВКонтакте [сайт]. [2022]. URL: <https://vk.com/press/music-playlists> (дата обращения: 09.05.2024).
11. Kotlin [Электронный ресурс] // Википедия : Свободная энциклопедия [сайт]. URL: <https://ru.wikipedia.org/wiki/Kotlin> (дата обращения: 16.05.2024).
12. Java [Электронный ресурс] // Википедия : Свободная энциклопедия [сайт]. URL: <https://ru.wikipedia.org/wiki/Java> (дата обращения: 16.05.2024).

13. Spring Boot [Электронный ресурс] // Wikipedia : The free encyclopedia [сайт]. URL: https://en.wikipedia.org/wiki/Spring_Boot (дата обращения: 16.05.2024).
14. Python [Электронный ресурс] // Википедия : Свободная энциклопедия [сайт]. URL: <https://ru.wikipedia.org/wiki/Python> (дата обращения: 17.05.2024).
15. FastAPI [Электронный ресурс] // Википедия : Свободная энциклопедия [сайт]. URL: <https://ru.wikipedia.org/wiki/FastAPI> (дата обращения: 17.05.2024).
16. Android Studio [Электронный ресурс] // Википедия : Свободная энциклопедия [сайт]. URL: https://ru.wikipedia.org/wiki/Android_Studio (дата обращения: 18.05.2024).
17. IntelliJ IDEA [Электронный ресурс] // Википедия : Свободная энциклопедия [сайт]. URL: https://ru.wikipedia.org/wiki/IntelliJ_IDEA (дата обращения: 18.05.2024).
18. PyCharm [Электронный ресурс] // Википедия : Свободная энциклопедия [сайт]. URL: <https://ru.wikipedia.org/wiki/PyCharm> (дата обращения: 18.05.2024).
19. PostgreSQL [Электронный ресурс] // Википедия : Свободная энциклопедия [сайт]. URL: <https://ru.wikipedia.org/wiki/PostgreSQL> (дата обращения: 18.05.2024).
20. Android. Программирование для профессионалов = Android programming. The big nerd ranch guide : [учебник] / Б. Филлипс, К. Стюарт, К. Марсикано ; пер. с англ. Е. Матвеева. - 3-е изд. - Санкт-Петербург : ПИТЕР, 2017. - 687 с.