Оглавление

[1. Введение 2](#_Toc164174904)

[1.1 Цель документа 2](#_Toc164174905)

[1.2 Контекст 2](#_Toc164174906)

[2. Общее описание 3](#_Toc164174907)

[2.1 Продукт 3](#_Toc164174908)

[2.2 Основные функции 3](#_Toc164174909)

[3. Требования пользователя 5](#_Toc164174910)

[3.1 Функциональные требования 5](#_Toc164174911)

[3.1.1 Поиск музыки 5](#_Toc164174912)

[3.1.2 Воспроизведение музыки 5](#_Toc164174913)

[3.1.3 Плейлисты 5](#_Toc164174914)

[3.1.4 Персонализация 5](#_Toc164174915)

[3.5.1 Cтримы 5](#_Toc164174916)

[4. Системные требования 6](#_Toc164174917)

[4.1 Платформы 6](#_Toc164174918)

[4.2 Технические требования 6](#_Toc164174919)

[5. Требования к внешним интерфейсам 7](#_Toc164174920)

[5.1 Интерфейс 7](#_Toc164174921)

[5.2 Аналитика 7](#_Toc164174922)

[6. Контроль качества 8](#_Toc164174923)

[6.1 Тестирование 8](#_Toc164174924)

[6.2 Отзывы пользователей 8](#_Toc164174925)

[7. Этапы разработки 9](#_Toc164174926)

[Минусы метода: 9](#_Toc164174927)

[Пример use case для нашего проекта представлен в Приложении А. 9](#_Toc164174928)

[Приложение A 10](#_Toc164174929)

[Приложение Б 11](#_Toc164174930)

### 1. Введение

### 1.1 Цель документа

Цель данного документа состоит в определении требований к разработке и функциональности приложения для прослушивания музыки.

### 1.2 Контекст

MusicListen разрабатывается с целью предоставить пользователям удобное и многофункциональное приложение для доступа к музыкальному контенту. В современном мире музыка стала важной частью жизни людей, и MusicListen призвано удовлетворить их потребности в прослушивании музыки в любое время и в любом месте.

### 2. Общее описание

### 2.1 Продукт

MusicListen представляет собой мобильное приложение, которое доступно как на платформах: iOS, Android, Windows, Mac, а также через веб-интерфейс. Приложение предоставляет пользователям широкие возможности по поиску, организации и прослушиванию музыкального контента.

### 2.2 Основные функции

**Поиск музыки**

* Пользователи могут искать музыкальные треки, альбомы и исполнителей по ключевым словам, что позволяет им быстро найти интересующий контент.
* Результаты поиска отображаются в удобном формате, предоставляя подробную информацию о каждом найденном элементе.

**Воспроизведение музыки**

* MusicListen обеспечивает возможность воспроизведения выбранных музыкальных композиций в высоком качестве, обеспечивая приятный и невероятный звук.
* Управление воспроизведением (включение, пауза, перемотка, регулировка громкости) реализовано интуитивно понятным образом, чтобы пользователи могли легко контролировать процесс прослушивания.

**Плейлисты**

* Пользователи могут создавать персональные плейлисты и управлять ими, добавляя новые треки, удаляя старые и изменяя порядок воспроизведения.
* Также предусмотрена возможность автоматического создания плейлистов на основе предпочтений и истории прослушивания пользователя.

**Персонализация**

* MusicListen предоставляет персонализированные рекомендации по музыкальным предпочтениям каждого пользователя, что позволяет им открывать новых исполнителей и находить интересные композиции.
* Рекомендации формируются на основе истории прослушивания и предпочтений пользователя, обеспечивая индивидуальный подход к каждому.

**Оффлайн режим**

* MusicListen предоставляет возможность скачивать музыкальные треки для прослушивания в оффлайн режиме, что позволяет пользователям наслаждаться музыкой даже при отсутствии интернет-соединения.
* Управление оффлайн контентом реализовано простым и понятным образом, обеспечивая пользователям контроль над своей музыкальной библиотекой.
* **Просмотр стримов и донат стримерам**

Пользователь должен иметь возможность просматривать стримы в наилучшем качестве, а также иметь возможность вознаградить стримера “благодарностью”.

Основные возможности пользователя изображены в приложении Б.

### 3. Требования пользователя

### 3.1 Функциональные требования

### 3.1.1 Поиск музыки

* **Поиск по ключевым словам:** Пользователи должны иметь возможность вводить запросы для поиска музыкальных треков, альбомов и исполнителей.
* **Результаты поиска:** После выполнения запроса должны отображаться результаты поиска в форме списка, где каждый элемент содержит информацию о найденном музыкальном объекте, такую как название трека, альбома или исполнителя, а также обложку и другие сведения при наличии.

### 3.1.2 Воспроизведение музыки

* **Контроль воспроизведения** Пользователи должны иметь возможность управлять воспроизведением музыки, включая функции воспроизведения/паузы, перемотки трека вперед/назад и регулировки громкости.
* **Информация о треке:** Во время воспроизведения должна быть доступна информация о текущем треке, такая как название, исполнитель и обложка альбома.

### 3.1.3 Плейлисты

* **Создание плейлистов:** Пользователи должны иметь возможность создавать собственные плейлисты и добавлять в них музыкальные треки.
* **Управление плейлистами:** Должна быть реализована функция управления плейлистами, включая добавление, удаление и изменение порядка треков.

### 3.1.4 Персонализация

* **Персонализированные рекомендации:** Система должна предоставлять персонализированные рекомендации по музыкальным предпочтениям пользователя на основе его истории прослушивания и предпочтений.
* **Автоматическое создание плейлистов:** Пользователям должны предлагаться автоматически созданные плейлисты на основе их предпочтений и поведения в приложении.

### 3.5.1 Cтримы

**Просмотр стримов и донат стримерам**: Пользователь должен иметь возможность просматривать стримы в наилучшем качестве, а также иметь возможность вознаградить стримера “благодарностью”.

**Запуск трансляций:** каждый пользователь должен имеет возможность для запуска трансляций за которыми могут наблюдать пользователи.

### 4. Системные требования

### 4.1 Платформы

* **Мобильные платформы:** Приложение должно быть доступно как на платформе iOS (для устройств Apple) и Android (для устройств на базе операционной системы Android).
* **Десктопные платформы:** Приложение должно быть доступно как на Windows OS, так и на Mac OS.
* **Веб-платформа:** Предусмотрена веб-версия приложения для доступа через веб-браузер на компьютерах и других устройствах.

### 4.2 Технические требования

* **Качество звука:** Приложение должно обеспечивать высокое качество воспроизведения звука, предпочтительно с использованием современных аудио-кодеков.
* **Скорость и производительность:** Приложение должно обеспечивать быструю загрузку и воспроизведение музыкальных треков, а также отзывчивость интерфейса на всех поддерживаемых платформах.

### 5. Требования к внешним интерфейсам

### 5.1 Интерфейс

* **Интуитивно-понятный дизайн:** Пользовательский интерфейс должен быть легким в освоении и приятным для использования. Это включает в себя понятную навигацию, простые элементы управления и понятное отображение информации.
* **Адаптивность:** Дизайн приложения должен быть адаптивным для различных размеров экранов и устройств, включая мобильные телефоны, планшеты и компьютеры.

### 5.2 Аналитика

* **Сбор и анализ данных:** Приложение должно включать встроенную систему аналитики для сбора данных о поведении пользователей. Это позволит проводить анализ пользовательского опыта, выявлять тренды и принимать обоснованные решения по улучшению приложения.
* **Конфиденциальность и безопасность:** Собираемые данные должны обрабатываться в соответствии с политикой конфиденциальности и безопасности, чтобы обеспечить защиту личной информации пользователей.

### 6. Контроль качества

### 6.1 Тестирование

* **Функциональное тестирование:** Провести тестирование всех функций приложения для проверки их работоспособности и соответствия требованиям.
* **Тестирование совместимости:** Убедиться, что приложение корректно работает на различных устройствах и в разных окружениях.
* **Тестирование производительности:** Проверить скорость работы приложения и его производительность, особенно в условиях высокой загрузки и при воспроизведении музыки в оффлайн режиме.

### 7. Этапы разработки

Для определения этапов разработки мы будем использовать user story mapping для наилучшего понимания какие из функций наиболее важны для разработки программного обеспечения.

Плюсы данного метода:

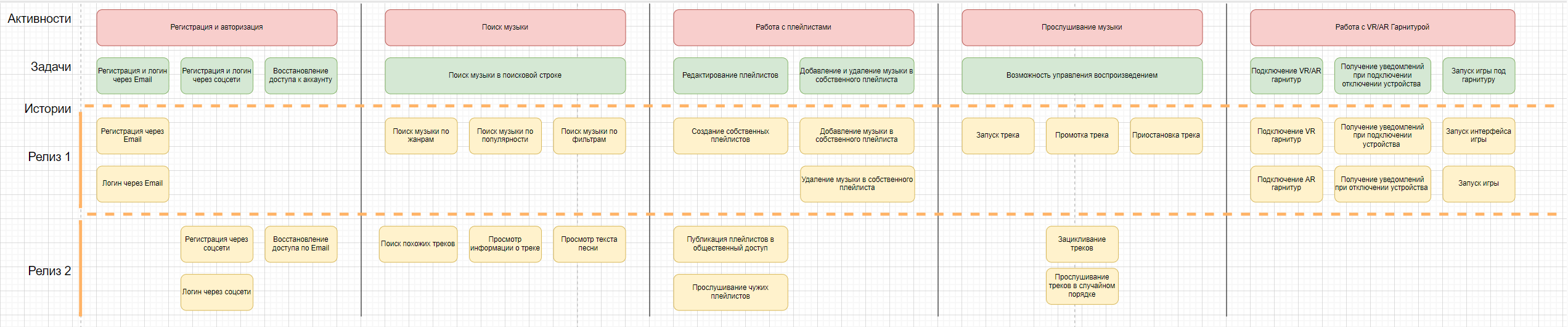
* **Улучшение коммуникаций внутри команды**. У каждого из нас уникальное видение чего-либо. В обычной жизни это хорошо, но для людей, работающих над одним продуктом, картина должна быть общей. UserStoryMapping—простой способ показать команде, над чем она работает и к каким результатам должна прийти совместными усилиями.
* **Фокус на пользователях**. В глазах потребителей ценный продукт = закрытые потребности. Если это игнорировать и работать только над функционалом, то ценность продукта будет сведена к нулю. Техника UserStoryMappingфокусирует внимание команды на потребностях конечных пользователей, а не только на функционале.
* **Максимальная визуализация.** Карты пользовательских историй раскрывают UserFlow—визуальное представление того, какие действия потребитель совершает, чтобы достигнуть цели. Это позволяет разработчикам не упустить важных шагов, а пользователям —получить продукт, который закроет их потребности.

Минусы метода:

* У UserStoryMappingтоже есть. как минимум —сложность организации. Если продукт объемный, составить бэклогза стандартные 2–3 часа невозможно. Проект придется разбить на несколько отдельных задач и выстраивать USM по каждой в отдельности.

### Пример use case для нашего проекта представлен в Приложении А.

### Приложение A

****

### Приложение Б

