**Введение**

Данный курсовой проект является проектом по разработке программного обеспечения, направленного на простое взаимодействие с треками на вашем устройстве. Зачастую для проигрывания треков не нужно подключение к интернету, но также есть и обратная сторона: синхронизация с базой данной для конкретного пользователя. Для обеспечения полного опыта в проигрывании музыки на различных персональных компьютерах было принято решение разрабатывать данное программное средство.

Цель курсового проекта – это создание программного обеспечения с графическим интерфейсом, направленного на простое, быстрое и удобное управление музыкальными композициями. Данное программное средство должно в малой степени нагружать систему, на котором используется приложение, и должно обладать удобным и понятным графическим интерфейсом для пользователей.

Перед началом выполнения проекта следует выделить наиболее важные задачи, которые будут заложены в основу данного программного обеспечения. Наиболее важной подзадачей является проектирование архитектуры программного средства, т. к. это является определяющим фактором дальнейшего развития и функционирования приложения. Следующий по степени влияния на дальнейшую разработку проекта это логика работы приложения, т.е. следующий этап – проектирование логики работы программного средства. Выбор языка программирования является ключевым аспектом данного проекта, т.к. нам потребуется высокоуровневый язык программирования, который сможет поддерживать интеграцию с графическим интерфейсом, а также позволит не сильно нагружать систему. Следующие этапы перечислю в виде списка, так как их аспекты будут раскрыты немного позже:

* Выбор технологии для создания графического интерфейса
* Проектирования эргономичного графического интерфейса
* Создание кода программного средства
* Тестирование программного средства

Минимальный функционал программы должен включать в себя возможность проигрывания треков с устройства и из базы данных, переключение между треками, систему аккаунтов, возможность добавлять или убирать треки, редактировать персональную информацию пользователя. Данные пункты должны быть использованы с помощью графического интерфейса.

Приняв во внимание все факты, изложенные выше, было принято решение использовать парадигму объектно-ориентированного программирования (ООП), что позволит сделать код более структурированным и обеспечить более легкую поддержку кода в будущем для сторонних разработчиков, если таковые появятся. Для разработки архитектуры приложения принято решение использовать архитектурный паттерн проектирования Model-View-Controller (MVC), который позволит разделить разрабатываемое приложение на слой представлений, которые будут отвечать за графическое отображение интерфейса, слой контроллера, который будет принимать на себя роль управления информацией от модели к представлению и наоборот, и модели, которая будет производить основную обработку информации приложения.

Выбор языка программирования оказывает существенное влияние на дальнейший процесс разработки, использованные технологии и скорость выполнения поставленных задач. Так как для нас будет важна скорость работы программы, широкая поддержка на различных устройствах, а также возможность использовать парадигму ООП, то стоит выбрать язык, созданный для работы с учетом всех этих условий. Основываясь на вышеперечисленных фактах был выбран язык программирования “C++”. Данный язык имеет поддержку ООП, является малотребовательным к возможностям персонального компьютера, например, если сравнивать с языком «Python», а также является гибким инструментом разработки за счет многолетнего существования на рынке и имеет многочисленное количество фреймворков.

В качестве среды разработки были выбраны следующие программы:

* CLion – для разработки основной части программы
* QT Creator – для разработки графического интерфейса, так как данная среда разработки предоставляет встроенный графический редактор.

В качестве основного фреймворка для разработки программы была выбрана технология QT, которая является кроссплатформенным инструментарием для разработки прикладного программного обеспечения, широко используемой для создания графических интерфейсов. Предоставляет быструю многоуровневую разработку, кроссплатформенность и имеет хорошую документацию.

**4.3.1 Ввод в эксплуатацию и обоснование минимальных технических требований к оборудованию.**

Далее будут описываться именно минимальные технические требования к оборудованию для работы с программным продуктом. Мы оцениваем нижнюю границу технических требования, чтобы достичь максимальной непрерывности и эффективности функционирования программного средства в худшем случае. Для того чтобы повысить удовольствие от используемого программного средство рекомендуем увеличить заданные характеристики.

Описывать абсолютно все технические характеристики не смысла, так как конечный пользователь будет в основном ориентироваться на основные, т.е. на параметры процессора (CPU), параметры графического вычислительного устройства (GPU), количество оперативной памяти (RAM) потребляемое приложением и количество занимаемой памяти на жестком диске.

Процессор не будет играть ключевую роль в работе программного средства, так что его параметры не обязаны быть значительными. Учитывая все необходимые компоненты для работы программного обеспечения, можно определить следующие характеристики.

* Количество ядер: 2
* Максимальная тактовая частота: 4.1 GHz
* Кэш-память: 6 MB
* Частота системной шины: 4 GT/s

Так как мы имеем дело с графическим интерфейсов, следовательно, параметры графического процессора, напротив, будут важны. Минимальные значения следующие.

* Видеопамять: 256 MB
* Разрядность шины видеопамяти: 128
* Частота видеопамяти: 600 MHz
* Частота графического процессора: 540 MHz

Как и характеристики графического процессора, количество оперативной памяти будет являться важным элементом технических характеристик. Этот параметр будет обеспечивать работу практически всей логики программного средства. Минимальное количество оперативной памяти таково.

* Оперативная память: 512 MB

Количество памяти на жестком диске устройства будет незначительным, ведь самому программному обеспечению требуется небольшое количество памяти (около 20-30 MB). Так как пользователь может добавлять собственные треки с устройства, нужно будет выделить на жестком диске и место под эти самые треки. Исходя из перечисленных фактов, минимальное требование для памяти на жестком диске:

* Объём памяти на жестком диске: 1 GB

Данное программное средство было протестировано на следующих операционных системах: WINDOWS OS (7/8/10/11) и Manjaro Linux 22.2 GNOME Edition. Для остальных операционных систем приложение может вести не так, как заявлено в требованиях, т.е. могут возникнуть с малой вероятностью случаи, когда приложение может не запуститься (если допустим операционная система слишком старая).

Подытоживая все вышеперечисленные требования, мы можем составить следующую таблицу.

|  |  |
| --- | --- |
| Графический процессор | Видеопамять: 216 MB  Разрядность шины видеопамяти: 128  Частота видеопамяти: 600 MHz  Частота графического процессора: 540 MHz |
| Процессор | Количество ядер: 2  Максимальная тактовая частота: 4.1 GHz  Кэш-память: 6 MB  Частота системной шины: 4 GT/s |
| Оперативная память | 512 MB |
| Дисковое пространство | 1 GB |
| Операционная система | Windows 7/Windows 8/Windows 8.1/Windows 10/Windows 11  Manjaro Linux GNOME Edition 20.2 Manjaro Linux GNOME Edition 22.2 |

**Заключение**

В ходе курсового проекта были выполнены все поставленные задачи. Была выполнена поставленная цель. Было разработано программное средство, удовлетворяющее поставленным задачам в начале курсового проекта. Были применены и освоены указанные технологии для разработки проекта. Приложение имеет архитектуру MVC, которая была заявлена, логику работы, соответствующая выбранной изначально. Были использованы некоторые принципы ООП.