## Пояснительная записка к проекту "M Музыка "

1. Введение

Данный проект представляет собой простой музыкальный плеер с графическим интерфейсом, разработанный с использованием Python, PyQt6 и Pygame. Плеер позволяет воспроизводить музыкальные файлы (mp3 и wav), управлять воспроизведением (воспроизведение/пауза, перемотка, смена треков), регулировать громкость, создавать плейлисты на основе базы данных и отображает метаданные треков (имя исполнителя и название композиции). Плеер включает модуль аутентификации пользователя.

2. Цели и задачи проекта

Цель: Разработка функционального и удобного музыкального плеера с возможностью управления плейлистом и аутентификацией пользователей.

Выполнение задачи:

• Разработка графического интерфейса пользователя (GUI) с использованием PyQt6.

• Реализация воспроизведения аудиофайлов с помощью Pygame.

• Создание системы управления воспроизведением (play/pause, next/previous).

• Реализация регулировки громкости.

• Хранение информации о песнях (метаданных) в базе данных SQLite.

• Реализация добавления песен в плейлист.

• Разработка модуля аутентификации пользователя (вход и регистрация).

• Ведение логов ошибок.

3. Архитектура проекта

Проект состоит из следующих основных частей:

• Графический интерфейс (GUI): Разработан с использованием PyQt6 и Qt Designer. Два основных окна: окно авторизации и основное окно плеера. XML-файлы ui-форм хранят информацию о структуре и стиле интерфейса.

• Модуль аутентификации: Использует библиотеку sqlite3 для хранение и проверки данных пользователя, secrets для создания уникальной соли и hashlib для хэширования паролей

• Модуль воспроизведения аудио: Использует библиотеку Pygame для загрузки и воспроизведения аудиофайлов.

• Обработка метаданных: Модуль metadata\_handler отвечает за получение метаданных из аудиофайлов и взаимодействие с базой данных SQLite. Функции add\_song\_metadata и get\_all\_songs обеспечивают сохранение и загрузку информации о песнях.

• Поток проверки авторизации (CheckThread): Обрабатывает запросы на вход и регистрацию, обеспечивая выполнение этих операций в отдельном потоке, чтобы не блокировать GUI.

• Логирование: Библиотека logging записывает ошибки в файл music\_player.log.

4. Реализация

• PyQt6: Используется для создания графического интерфейса, обработки событий и взаимодействия с пользователем.

• Pygame: Обеспечивает воспроизведение аудиофайлов.

• SQLite: Используется как база данных для хранения информации о песнях (метаданных).

• Многопоточность: Используется для обработки запросов авторизации в отдельном потоке, что предотвращает зависание интерфейса.

5. Тестирование

Проект был протестирован на работоспособность всех основных функций:

• Воспроизведение и пауза аудиофайлов.

• Регулировка громкости.

• Перемотка (не работает, в процессе разработки).

• Переключение между треками.

• Добавление песен в плейлист.

• Вход и регистрация пользователей.

• Загрузка и отображение метаданных(У фалов типа .wav не отображается длительность).

6. Будущие разработки

• Добавление поддержки большего числа аудиоформатов.

• Реализация функции поиска песен по названию или исполнителю.

• Возможность создания и сохранения плейлистов.

• Дополнительные функции управления воспроизведением (например, повторение, случайное воспроизведение).

• Улучшение дизайна графического интерфейса.

7. Заключение

Разработанный музыкальный плеер "М Музыка" представляет собой функциональное и относительно простое приложение, демонстрирующее основные навыки работы с Python, PyQt6 и Pygame. Проект может быть использован как основа для создания более сложных музыкальных плееров с расширенным функционалом.

8. Используемые библиотеки:

• pygame

• sys

• logging

• os

• io

• PyQt6 (uic, QtWidgets, QtCore, QtGui)

• sqlite3

• secrets

• hashlib