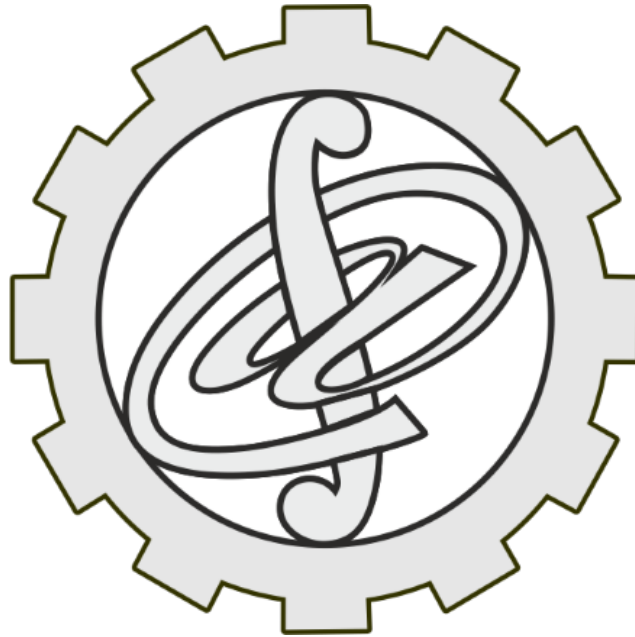


# Programowanie Obiektowe i Graficzne

Dokumentacja projektu 'Odtwarzacz multimedialny'

Patryk Gamrat, Radosław Olesiński, Radosław Szwed, Karol Zając, grupa 2/4

27 czerwca 2024



# Spis treści

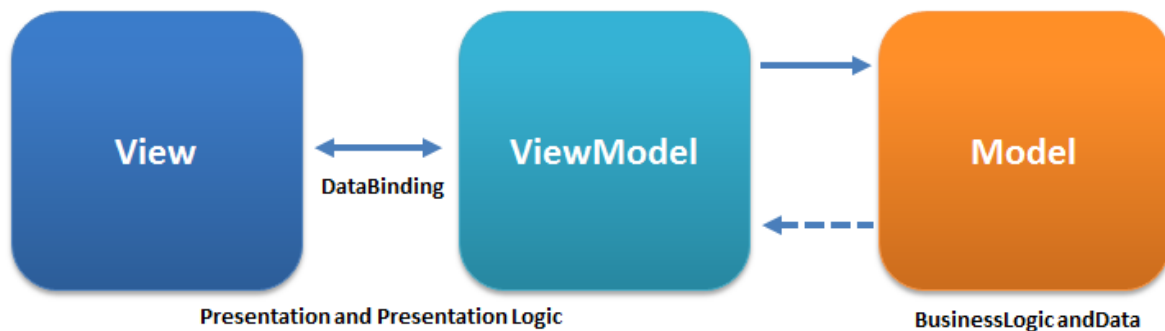
<b>1</b>	<b>Opis aplikacji</b>	<b>3</b>
1.1	Opis projektu . . . . .	3
1.2	Funkcje aplikacji . . . . .	4
1.3	Instrukcja obsługi . . . . .	5
1.4	Wykorzystane technologie . . . . .	12
1.5	Opis klas . . . . .	12
1.6	Bazy danych . . . . .	13
<b>2</b>	<b>Podsumowanie</b>	<b>13</b>
2.1	Wnioski . . . . .	13
2.2	Kierunki rozwoju . . . . .	13

# 1 Opis aplikacji

## 1.1 Opis projektu

Celem projektu było stworzenie odtwarzacza multimedialnego stworzonego w technologii **WPF** przy użyciu wzorca **MVVM**. Program pozwala na odtwarzanie filmów i muzyki oraz tworzenie, zapisywanie i odtwarzanie playlist dzięki bazie danych SQLite. Dodatkowo program posiada funkcję pobierania tekstów utworów.

Wzorzec MVVM (Model–view–viewmodel) zakłada istnienie w programie trzech warstw (model, viewmodel oraz view). Celem tego rozwiązania jest oddzielenie logiki biznesowej aplikacji od interfejsu użytkownika, co ułatwia wymianę widoku oraz testowanie aplikacji.



Rysunek 1: Diagram MVVM

Źródło: <https://en.wikipedia.org/wiki/Model-view-viewmodel/media/File:MVVMPattern.png>

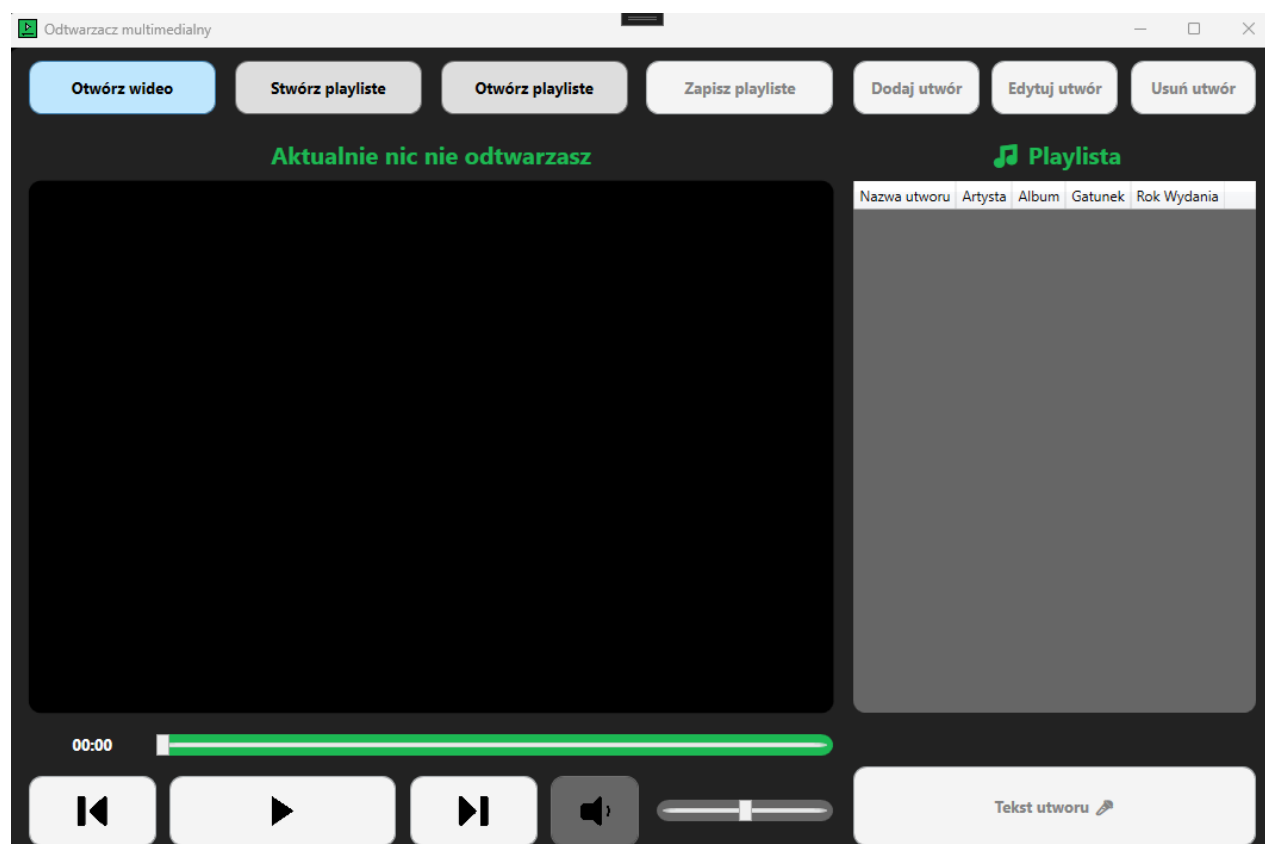
- **Model** - warstwa zawierająca dane i reguły biznesowe aplikacji
- **Viewmodel** - warstwa umożliwiająca komunikację między warstwą widoku i warstwą modelu.
- **View** - warstwa wizualnej reprezentacji programu, ma na celu interakcję z użytkownikiem.

## 1.2 Funkcje aplikacji

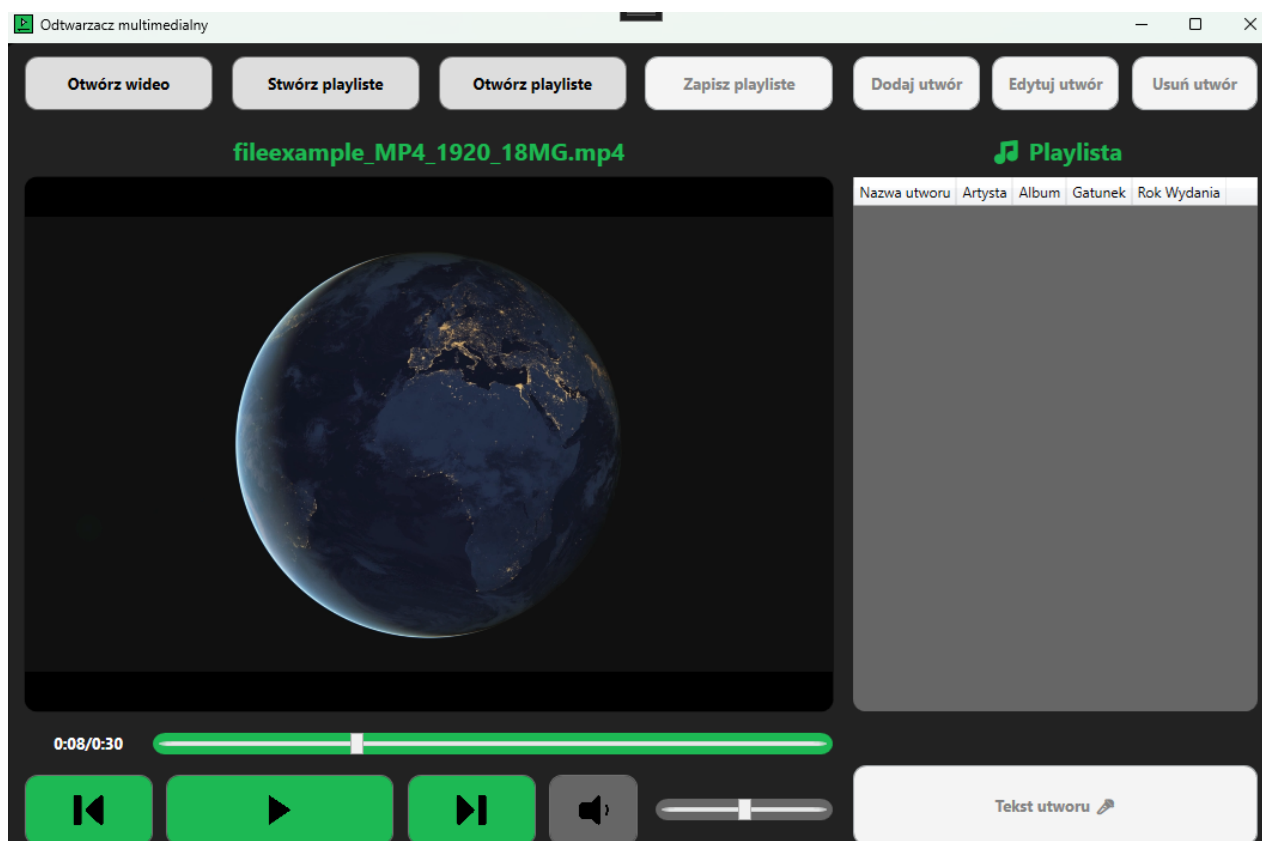
- Odtwarzanie plików wideo w różnych formatach (.avi, .mp4)
- Odtwarzanie plików audio w różnych formatach (.mp3, .wav, .wma, .aac)
- Możliwość zatrzymywania oraz przewijania odtwarzanych multimediiów
- Możliwość zmiany głośności oraz całkowite wyciszenie dźwięku
- Tworzenie własnych playlist
- Dodawanie utworów do playlisty z możliwością późniejszej edycji informacji o danym utworze (nazwa utworu, artysta, gatunek, rok wydania, ścieżka do pliku)
- Automatycznie pobieranie informacji o utworze oraz okładki z metadanych pliku
- Wyświetlanie okładki odtwarzanego utworu (jeżeli takowa jest dostępna)
- Pobieranie tekstu odtwarzanego utworu za pomocą zewnętrznego API

### 1.3 Instrukcja obsługi

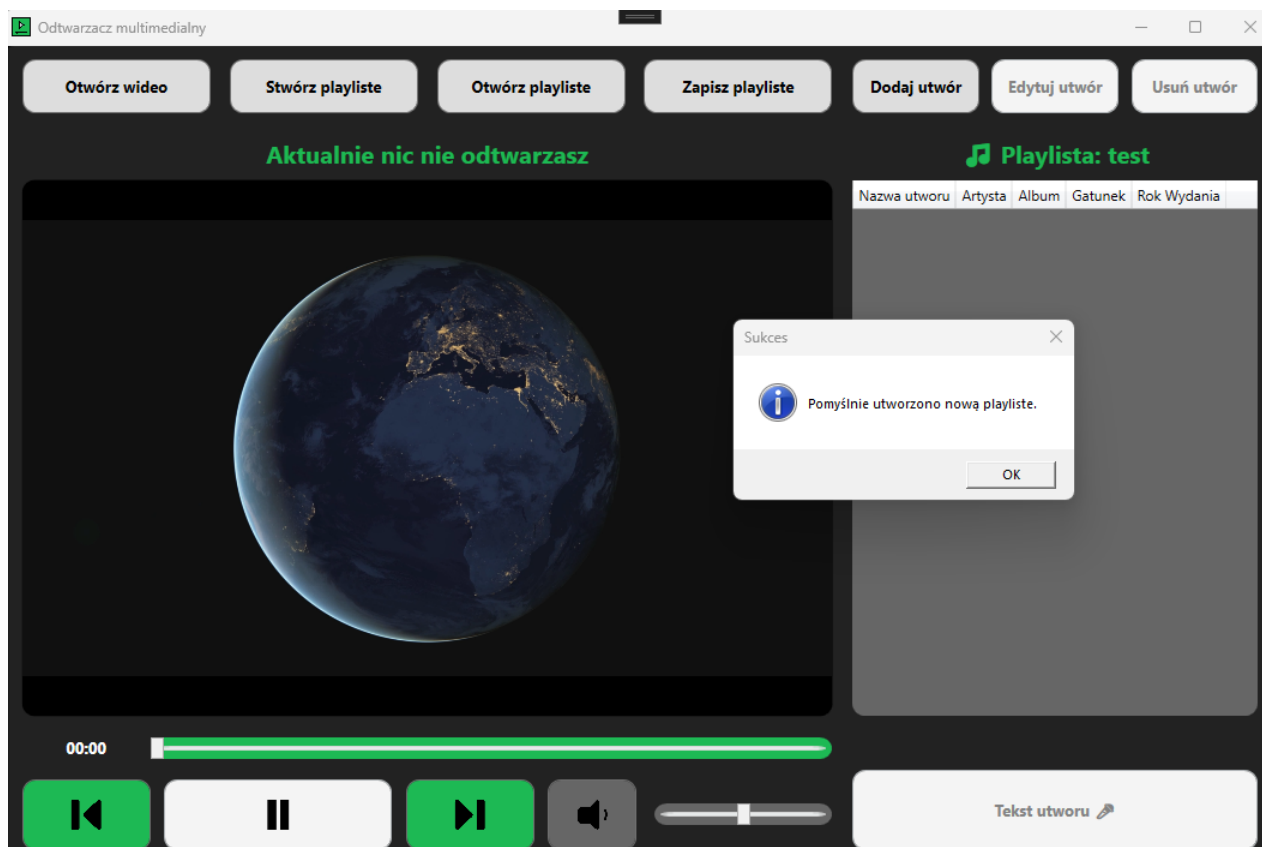
Interfejs aplikacji jest responsywny i można go skalować do większego okna. Poniżej przedstawione są zrzuty ekranu prezentujące działanie aplikacji:





Rysunek 2: Główny widok programu po uruchomieniu



Rysunek 3: Odtwarzanie wideo



Rysunek 4: Tworzenie nowej playlisty

 Dodaj nowy utwór 

## DODAJ NOWY UTWÓR

**Plik**

Wybierz

**Nazwa utworu**

**Artysta**

**Album**

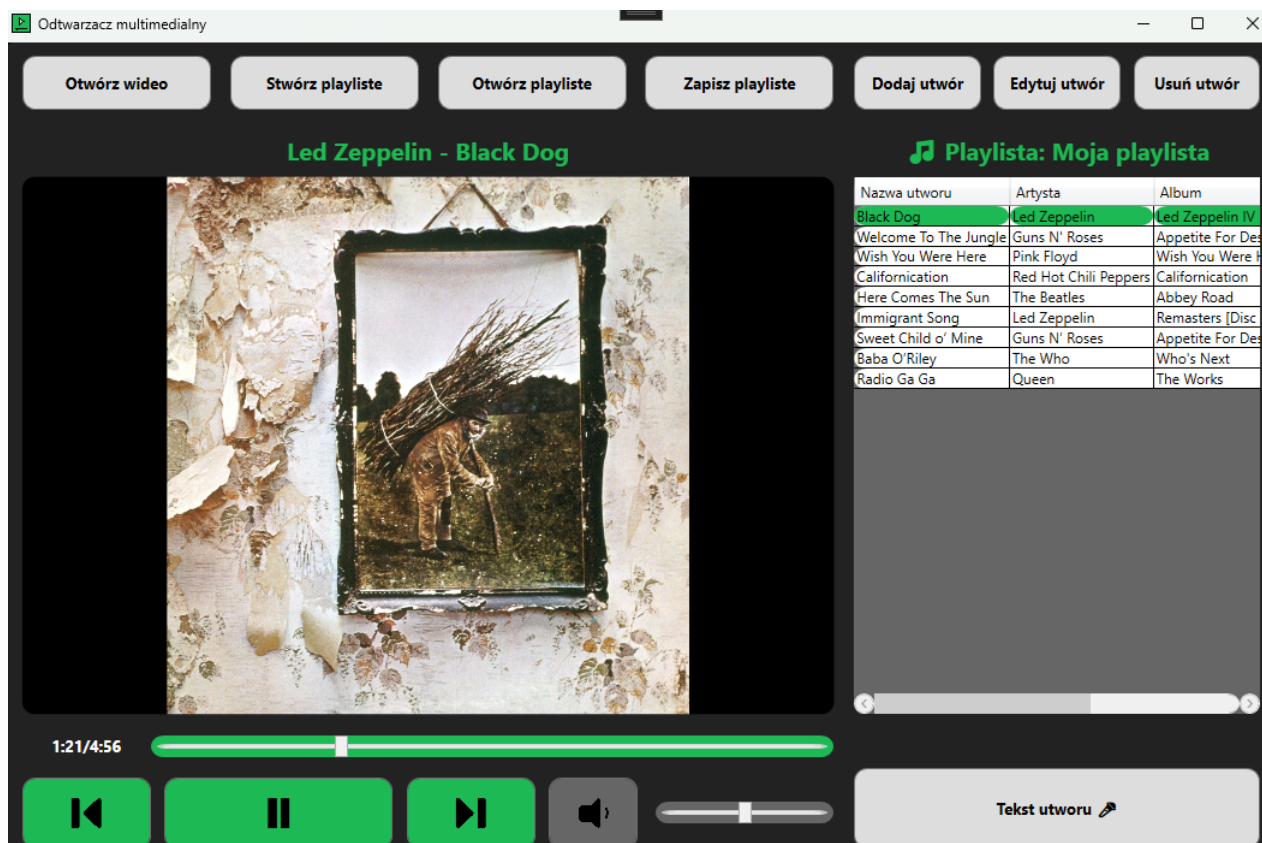
**Gatunek**

**Rok wydania**

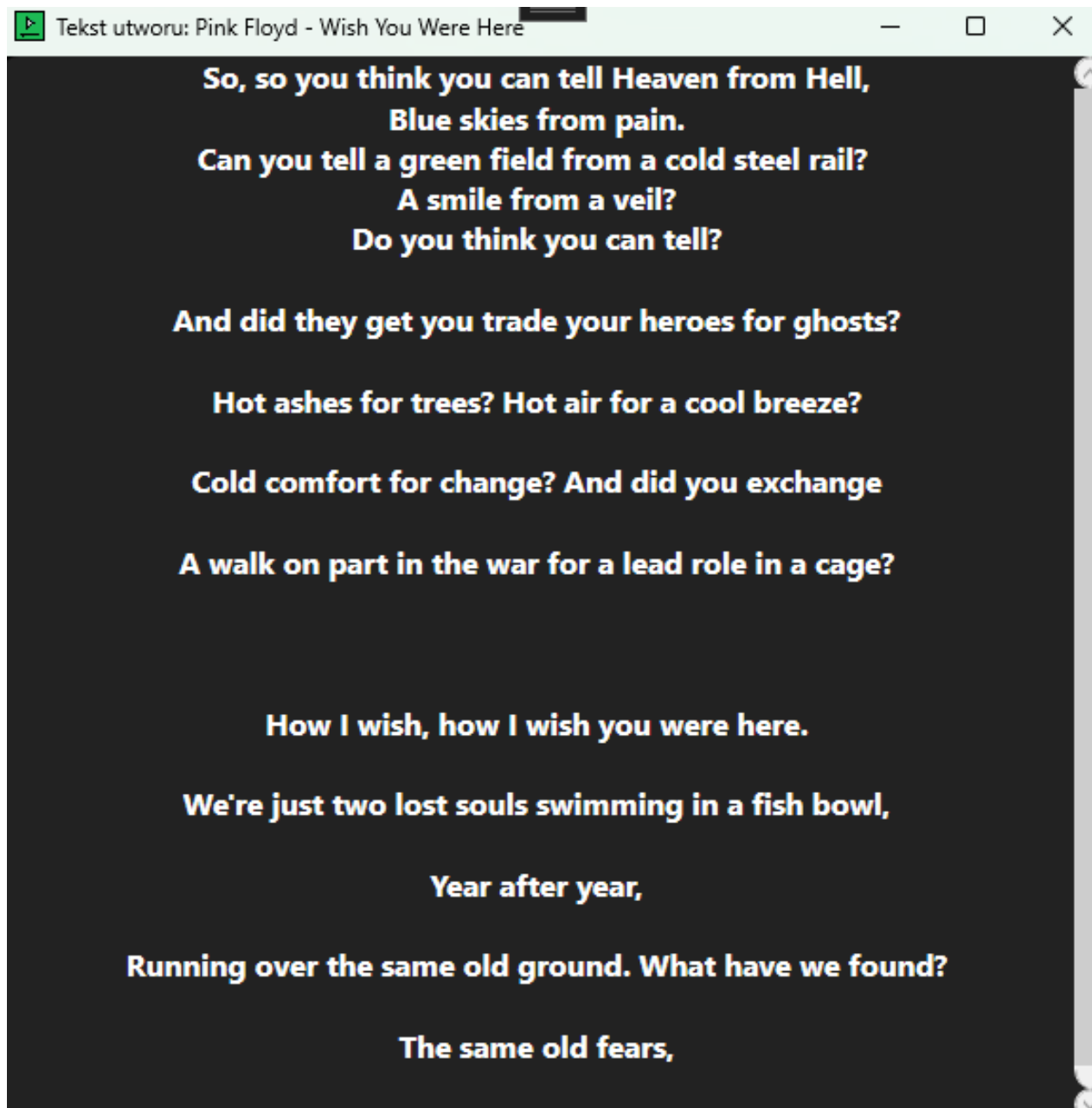
Dodaj

Rysunek 5: Dodawanie nowego utworu

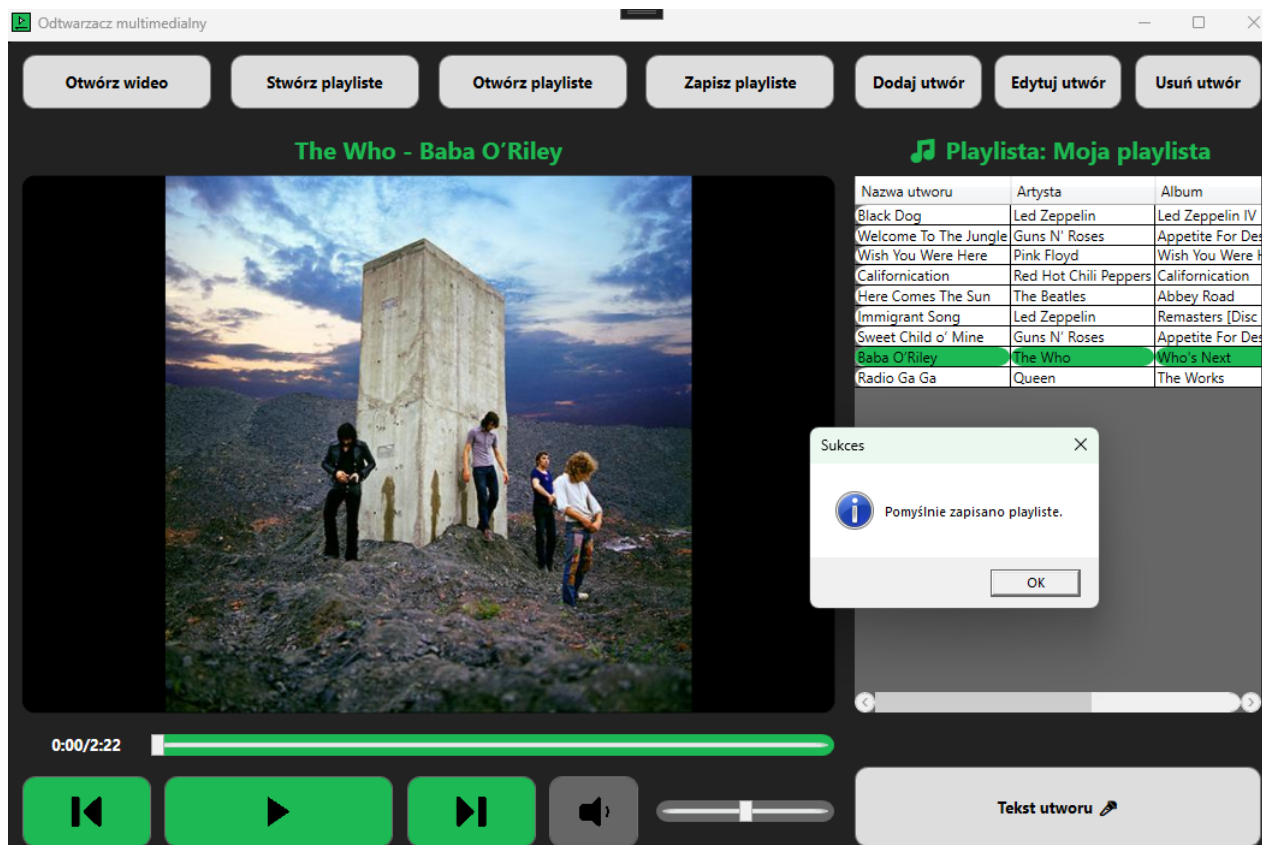




Rysunek 6: Odtwarzanie pliku dźwiękowego z playlisty



Rysunek 7: Wyświetlanie tekstu piosenki



Rysunek 8: Zapisywanie playlisty

## 1.4 Wykorzystane technologie

Aplikacja została stworzona w języku C# przy użyciu technologii WPF. W celu tworzenia playlist wykorzystano bazę danych **SQLite**. W celu pobierania tekstów piosenek wykorzystano API **Lyrics.ovh**, a do odczytywania metadanych z plików dźwiękowych użyto biblioteki **TagLibSharp**. W celu ułatwienia pracy zespołowej wykorzystano serwis GitHub.

### Wykorzystane biblioteki:

- **Newtonsoft.Json** - do odczytywania tekstu piosenki z API
- **System.Data.SQLite** - do zapisu playlist
- **TagLibSharp** - do odczytu metadanych z plików audio

## 1.5 Opis klas

- **Model**
  - **DbConnection** - klasa typu singleton zapewniająca połączenie z bazą danych.
  - **LyricsService** - klasa pozwalająca na pobranie tekstów piosenek przy użyciu zewnętrznego API
  - **Playlist** - klasa umożliwiająca tworzenie, zapisywanie, wczytywanie i modyfikowanie playlist.
  - **PlaylistFileDialog** - klasa obsługująca okno wyboru pliku.
  - **Track** - zawiera informacje dotyczące utworu
- **ViewModel**
  - **AddEditTrack** - obsługuje okno służące do dodawania lub edytowania szczegółów utworu takich jak nazwa utworu i wykonawca, gatunek, ścieżka do pliku itp.
  - **MediaElementViewModel** - obsługuje element wyświetlający pliki multimedialne oraz przyciski do kontroli, czyli odtwarzanie, pauza i głośność. Pokazuje nazwę odtwarzanego pliku i pozwala użytkownikom wyciszyć dźwięk.
  - **MediaPlayer** - obsługuje główny interfejs aplikacji, pomaga odtwarzać muzykę i filmy w aplikacji, zarządza listami odtwarzania
  - **ViewModelShare** - służą do przekazywania danych pomiędzy różnymi częściami aplikacji
- **View**
  - **MainWindow** - główne okno, widok programu zawierający odtwarzacz multimedialny oraz wszystkie kontrolki
  - **AddEditTrackWindow** - okno służące do dodania lub edycji utworu w aktualnie otworzonej playliście
  - **LyricsWindow** - okno które wyświetla tekst obecnie słuchanej piosenki

## 1.6 Bazy danych

W wykorzystanej bazie danych SQLite znajdują się informacje dotyczące utworów należących do playlist: tytuł, wykonawca, nazwa albumu, gatunek muzyczny, rok wydania oraz ścieżka do pliku audio.

## 2 Podsumowanie

### 2.1 Wnioski

- Wykorzystanie wzorca Model-view-viewmodel (MVVM) pozwala na lepszą separację warstwy graficznej i warstwy biznesowej.
- Dzięki technologii WPF można w prosty sposób utworzyć graficzny interfejs aplikacji.
- Wykorzystanie platformy GitHub do zarządzania projektem znacząco przyczyniło się do poprawy organizacji, współpracy i kontroli wersji w procesie rozwoju aplikacji.
- Wykorzystanie API z lyrics.ovh w naszym projekcie umożliwiło dodanie funkcji wyświetlania tekstów piosenek podczas odtwarzania muzyki.
- Wykorzystanie bazy danych SQLite do tworzenia i zarządzania playlistami w naszym odtwarzaczu multimedialnym znacznie ułatwiło organizację i dostęp do ulubionych utworów użytkowników.

### 2.2 Kierunki rozwoju

- Zaimplementowanie funkcji equalizera dźwięku
- Dodanie możliwości zmiany prędkości odtwarzania audio i wideo
- Wprowadzenie dodatkowych sposobów odtwarzania playlisty, np odtwarzanie w kolejności losowej
- Dodanie możliwości wyświetlania wideo na pełnym ekranie
- Utworzenie funkcji pozwalającej przeglądać utwory według artystów, albumów lub gatunków
- Poprawa i dodanie funkcjonalności związanych z odtwarzaniem playlist, takich jak ładowanie ich do aplikacji razem z jej startem
- Dodanie możliwość wyświetlania tekstu utworu w różnych językach
- Wprowadzenie opcji śledzenia tekstu utworu w czasie rzeczywistym
- Zaimplementowanie systemu rekomendacji nowych utworów na podstawie naszej playlisty