



**WYŻSZA SZKOŁA
INFORMATYKI i ZARZĄDZANIA**
z siedzibą w Rzeszowie

Wyższa Szkoła Informatyki i Zarządzania
Kolegium Informatyki Stosowanej, kierunek Programowanie

Mariia Volokhonska w67971

Projekt
Szkolenie techniczne 1

Rzeszow 2024

Contents

1	Link do repozytorium GitHub	3
2	Opis wybranego tematu, funkcjonalności systemu	3
2.1	Temat projektu	3
2.2	Funkcjonalność	3
3	Opisy klas i metod	3
3.1	Folder SongsRepo	3
3.2	RepositorySQL	3
3.3	Home	4
3.4	Authorization	4
3.5	Registration	5
3.6	SongViewModel	5
3.7	UserProfileViewModel	5
3.8	SongsList	5
3.9	SongsControl	6
3.10	FavouriteSongs	6
3.11	FavouritesControl	7
3.12	UserProfile	8
3.13	UserProfileInformation	8
3.14	Program	8
4	Diagram klasów	9
5	GUI	9
5.1	Home Page	9
5.2	Registration Page	10
5.3	Authorization Page	10
5.4	Songs Repository Page	11
5.5	Favourite Songs Page	12
5.6	UserProfile Page	12

1 Link do repozytorium GitHub

<https://github.com/MariiaVolokhonska/PlayerDesktop>

2 Opis wybranego tematu, funkcjonalności systemu

2.1 Temat projektu

Dla danej pracy projektowej został wybrany temat "Desktopowa aplikacja do wyboru plików muzycznych z bazy danych z możliwością odsłuchania", czyli player muzyczny.

Temat ten jest szczególnie istotny w dzisiejszych czasach, ponieważ większość ludzi codziennie korzysta z odtwarzaczy muzycznych. Trudno wyobrazić sobie podróż autobusem bez słuchawek, posiłek w restauracji bez muzyki w tle, czy trening na siłowni bez energetycznych rytmów, prawda? Ten projekt wyróżnia się tym, że zapobiega przypadkowemu nabywaniu pirackich utworów, ponieważ korzysta wyłącznie z legalnych, darmowych utworów dostępnych w internecie. Dodatkowo, projekt ten ma ogromny potencjał rozwojowy – może przekształcić się w aplikację z zaawansowanym systemem rekomendacji, możliwością tworzenia nowych playlist na różne nastroje, oddzielnymi kontami dla użytkowników i twórców utworów, i wiele więcej.

2.2 Funkcjonalność

Główne elementy funkcjonalności, które charakteryzują ten projekt:

1. Tworzenie konta nowego użytkownika.
2. Logowanie i wylogowanie z konta nowego użytkownika.
3. W zależności od zalogowanego użytkownika, wyświetlenie jego ulubionych piosenek.
4. Możliwość dodać utwór do ulubionych z tego Wspólnego Repozytorium Utworów.
5. Możliwość do odtwarzania utworów z listy ulubionych i z repozytorium.
6. Możliwość obejrzenia danych użytkownika na koncie użytkownika.

3 Opisy klas i metod

3.1 Folder SongsRepo

To jest folder, w którym są przechowywane utwory do odtwarzania.

3.2 RepositorySQL

Ten klas służy do połączenia z bazą danych oraz możliwości otwierania i zamykania połączenia. Zawiera:

1. Zawiera obiekt klasy SqlConnection, właśnie charakteryzuje to łączenie z bazą danych.

2. Metoda `OpenConnection()` - ta metoda otwiera połączenie z bazą danych. Najpierw sprawdza stan połączenia za pomocą `SqlConnection.State`. Jeśli stan połączenia jest `Closed` (zamknięty), to metoda `SqlConnection.Open()` otwiera połączenie.
3. Metoda `CloseConnection()` Zamyka połączenie z bazą danych. Najpierw sprawdza stan połączenia za pomocą `SqlConnection.State`. Jeśli stan połączenia jest `Open` (otwarty), to metoda `SqlConnection.Close()` zamyka połączenie.
4. Metoda `getConnection` - zwraca "teraźniejsze połączenie z bazą danych". Umożliwia dostęp do bazy danych w różnych częściach kodu.

3.3 Home

To jest "forma", którą użytkownik widzi najpierwszą przy uruchamianiu aplikacji. Posiada ona 2 metody, które pełnią raczej funkcje nawigacyjne, oraz jeden konstruktor. Te metody:

1. konstruktor posiada metodę `InitializeComponent`, co pozwala na inicjalizację tej formy.
2. `button1_Click(object sender, EventArgs e)` - przekierowuje użytkownika na stronę rejestracji po naciśnięciu przycisku "Sign Up".
3. `button2_Click(object sender, EventArgs e)` - przekierowuje użytkownika na stronę logowania po naciśnięciu przycisku "Log In".

3.4 Authorization

To jest "forma", która odpowiada za logowanie użytkownika w systemie i pamięta identyfikator użytkownika w celu udostępnienia możliwości wyświetlania w profilu użytkownika jego danych z bazy danych.

1. Zawiera obiekt klasy `RepositorySQL`, żeby mieć możliwość robić zapytania do bazy danych.
2. Zawiera konstruktor, który odpowiada za inicjalizację formy `Windows Forms` oraz ustawia pozycję początkową na środek ekranu.
3. Metoda `SearchForLog()`, która przy pomocy `SqlDataAdapter` powiązuje dane z bazy danych oraz wirtualną tabelę, która lokalnie przechowuje teraz te dane. Dane te są otrzymane przy pomocy użycia zapytania do bazy danych (`select from where`), gdzie w "where" sprawdzano jest, czy jest użytkownik o wprowadzanych z pól tekstowych danych obecny w bazie danych. O tym można sądzić z " `if (table.Rows.Count == 1)`", dlatego że to oznacza, że w tą wirtualną tabelę został przekazany dokładnie jeden rekord z powyższymi wymaganiami do niego. Jeżeli użytkownik istnieje w bazie, korzystamy z "`SqlDataReader reader=command.ExecuteReader()`", żeby przeczytać z bazy w zmienną klasy statycznej (analog globalnej zmiennej, ponieważ takich w C nie istnieje) id użytkownika dla podalszej możliwości wyświetlania danych użytkownika w jego profilu. Również w tej metodzie znajduje się rząd poleceń, które przeniosą użytkownika po logowaniu na stronę ze wszystkimi utworami w tej aplikacji (repozytorium utworów).
4. `button1_Click(object sender, EventArgs e)` - tą metodą łączy zdarzenie naciśnięcia przycisku "Log In" z działaniem, które musi się wystąpić po tym zdarzeniu. W tym przypadku, będzie wywołana metoda `searchForLog()`, która sprawdzi, czy dane użytkownika są w bazie, wyświetli komunikat, mówiący o udanym czy nie udanym logowaniu oraz zapisze jego id.

3.5 Registration

To jest "forma", która odpowiada za rejestrację użytkownika w systemie.

1. Zawiera obiekt klasy RepositorySQL, żeby mieć możliwość robić zapytania do bazy danych.
2. Zawiera konstruktor, który odpowiada za inicjalizację formy windows forms.
3. Posiada metodę signUp(), która wyciąga z pól tekstowych wartości i przy pomocy if (command.ExecuteNonQuery() == 1), wykonując polecenie INSERT INTO dodaje do bazy danych nowego użytkownika. Jeżeli kod, zwrócony po dodaniu użytkownika =1, to użytkownika przekierowano do strony autorizacji użytkownika. Jeżeli kod się różni od 1, to wyświetlony jest komunikat, że konto nie zostało stworzone. Jeszcze ta funkcja posiada funkcjonalność sprawdzenia, czy wartość pola "Password" i "Repeat Password" jest jednakowa, jeżeli wartości się różnią, to wyświetlany jest komunikat o tym, że hasło nie prawidłowe.
4. Metoda signUpButton_Click(object sender, EventArgs e) łączy się zdarzenie kliknięcia na przycisk "Sign up" oraz działanie, które te kliknięcie powoduje. W tym przypadku, tym działaniem jest wywołanie metody "signUp()", która została opisana wyżej.

3.6 SongViewModel

To jest zwykła klasa, która przechowuje dane piosenek, które w przyszłości będą wyświetlane wizualnie na stronach. Oprócz pól z danymi, w których będą znajdować informacje z bazy danych dotyczące utworów, również znajduje się tam override ToString metoda, która zwraca string, zawierający postać tego, w jaki sposób chcemy reprezentować dane obiektu klasy SongViewModel (w tym przypadku, w takiej postaci: \$"Title Artist Album Genre Duration ").

3.7 UserProfileViewModel

To jest zwykła klasa, która przechowuje dane użytkowników, które w przyszłości będą wyświetlane wizualnie na stronie profilu użytkownika.

3.8 SongsList

To jest forma, która wyświetla listę utworów, przechowywanych w bazie razem z dwoma przyciskami dla każdego rekordu, odpowiadającym działaniom "słuchać piosenkę" oraz "dodać na listę ulubionych". Składa ona się z:

1. Zawiera obiekt klasy RepositorySQL, żeby mieć możliwość robić zapytania do bazy danych.
2. Zawiera konstruktor, który odpowiada za inicjalizację formy windows forms.
3. Metoda LoadSongs() - definiuje zapytanie SQL, które łączy tabele Songs, Artists, Albums oraz Genres, aby uzyskać wszystkie potrzebne informacje o piosenkach; tworzy SqlCommand i wykonuje go przy pomocy SqlDataReader; pętla while odczytuje każdy wiersz wyników zapytania. Dla każdej piosenki pobiera odpowiednie kolumny (ID, tytuł, nazwisko artysty, nazwa albumu, gatunek, czas trwania, lokalizacja) i dodaje nową instancję SongViewModel do listy listOfSongs; dla każdej piosenki w liście listOfSongs, tworzy nową kontrolkę SongsControl i dodaje ją do flowLayoutPanel1.

4. Zdarzenie `showButton_Click()` - wywoływane po kliknięciu przycisku `showButton`, co powoduje wyświetlanie piosenek za pomocą metody `LoadSongs`.
5. Zdarzenie `logOutToolStripMenuItem_Click()` - wywoływane po kliknięciu opcji `logOut` w menu. Ukrywa bieżące okno i pokazuje okno `Home`.
6. Zdarzenie `favouritesToolStripMenuItem1_Click()` - wywoływane po kliknięciu opcji `favourites` w menu. Ukrywa bieżące okno i pokazuje okno `FavouriteSongs`.
7. Zdarzenie `favouritesToolStripMenuItem_Click` - wywoływane po kliknięciu opcji `profile` w menu. Ukrywa bieżące okno i pokazuje okno `UserProfile`.

3.9 SongsControl

Poniżej znajduje się opis klasy `SongsControl`, która jest kontrolką użytkownika w aplikacji, obsługującą odtwarzanie utworów muzycznych oraz dodawanie ich do ulubionych. Jest jedną z najważniejszych klas w tym projekcie. Składa się z:

1. Zawiera obiekt klasy `RepositorySQL`, żeby mieć możliwość robić zapytania do bazy danych.
2. Zawiera konstruktor, który: inicjalizuje komponenty; przypisuje wartości do pól `_songViewModel`, `_location` i `_songId`; Ustawia teksty etykiety (`label1`) oraz przycisków (`button1`, `button2`, `button3`); dodaje zdarzenia kliknięcia do przycisków `button1` i `button2`.
3. Metoda `PlayMp3FilePlay()` - sprawdza, czy instancja `waveOut` istnieje i zatrzymuje ją, jeśli tak; inicjalizuje `audioFileReader` i `waveOut`; odtwarza plik audio.
4. Metoda `StopMp3File()` - sprawdza, czy instancja `waveOut` istnieje i zatrzymuje ją; resetuje pozycję odczytu pliku audio.
5. Metoda `AddToFavourites()` - tworzy zapytanie SQL wstawiające piosenkę do tabeli `Favourites`; wykonuje zapytanie i wyświetla komunikat o sukcesie lub błędzie.
6. `button1_Click()`: Wywołuje `PlayMp3FilePlay`.
7. `button2_Click()`: Wywołuje `StopMp3File`.
8. `button3_Click`: Wywołuje `AddToFavourites` i otwiera nowe okno `FavouriteSongs`.

3.10 FavouriteSongs

Ta klasa umożliwia wyświetlanie na ekranie ulubionych utworów użytkownika oraz nawigację do innych stron, takich jak `Log Out`, `User Account` oraz `Songs Repository` (gdzie wyświetlane są wszystkie utwory w bazie).

1. Zawiera obiekt klasy `RepositorySQL`, żeby mieć możliwość robić zapytania do bazy danych oraz obiekt klasy `UserProfileInformation.USER_ID`, który umożliwi wyświetlanie ulubionych utworów tylko potocznego użytkownika.
2. Zawiera konstruktor, który inicjalizuje komponenty.

3. Metoda LoadSongs() - tworzy nową listę listOfSongs dla przechowywania ulubionych piosenek i otwiera połączenie z bazą danych; tworzy zapytanie SQL, które łączy tabele Favourites, Songs, Artists, Albums i Genres, aby uzyskać informacje o ulubionych piosenkach użytkownika. Następnie tworzy SqlCommand i wykonuje go przy pomocy SqlDataReader; pętla while odczytuje każdy wiersz wyników zapytania, pobierając odpowiednie kolumny (ID, tytuł, nazwisko artysty, nazwa albumu, gatunek, czas trwania, lokalizacja) i dodaje nową instancję SongViewModel do listy listOfSongs; dla każdej piosenki w liście listOfSongs tworzy nową kontrolkę FavouritesControl i dodaje ją do flowLayoutPanel2.
4. button1_Click(): Wywoływane po kliknięciu przycisku button1, co powoduje załadowanie ulubionych piosenek za pomocą metody LoadSongs.
5. songRepositoryToolStripMenuItem_Click(): wywoływane po kliknięciu opcji songRepository w menu. Ukrywa bieżące okno i pokazuje okno SongsList.
6. userAccountToolStripMenuItem_Click(): wywoływane po kliknięciu opcji userAccount w menu. Ukrywa bieżące okno i pokazuje okno UserProfile.
7. wywoływane po kliknięciu opcji logOut w menu. Ukrywa bieżące okno, resetuje ID użytkownika i pokazuje okno Home.

3.11 FavouritesControl

Poniżej znajduje się opis klasy FavouritesControl, która jest kontrolką użytkownika w aplikacji, obsługującą odtwarzanie utworów muzycznych. Składa się z:

1. Zawiera obiekt klasy RepositorySQL, żeby mieć możliwość robić zapytania do bazy danych.
2. Zawiera konstruktor z parametrem songViewModel, który: inicjalizuje komponenty GUI; przypisuje wartości do pól _songViewModel, _location i _songId; ustawia tekst etykiety (label1) na reprezentację tekstową obiektu songViewModel; ustawia teksty przycisków button1 i button2; dodaje zdarzenia kliknięcia do przycisków button1 i button2.
3. Zawiera ikonstruktor domyślny, który inicjalizuje komponenty GUI dla klasy FavouritesControl.
4. Metoda PlayMp3FilePlay() - sprawdza, czy instancja waveOut istnieje i zatrzymuje ją, jeśli tak; inicjalizuje audioFileReader i waveOut; odtwarza plik audio.
5. Metoda StopMp3File() - sprawdza, czy instancja waveOut istnieje i zatrzymuje ją; resetuje pozycję odczytu pliku audio.
6. Metoda AddToFavourites() - tworzy zapytanie SQL wstawiające piosenkę do tabeli Favourites; wykonuje zapytanie i wyświetla komunikat o sukcesie lub błędzie.
7. button1_Click(): wywołuje metodę PlayMp3FilePlay, aby odtworzyć plik audio z lokalizacji _location.
8. button2_Click(): wywołuje metodę StopMp3File, aby zatrzymać odtwarzanie pliku audio.

3.12 UserProfile

To jest forma, która wyświetla dane zalogowanego w systemie użytkownika. Posiada ta klasa 3 metody:

1. Zawiera obiekt klasy RepositorySQL, żeby mieć możliwość robić zapytania do bazy danych.
2. Zawiera konstruktor, który odpowiada za inicjalizację formy windows forms oraz wywołuje metodę ShowUserData(), w wyniku czego zostanie wyświetlona informacja o potoczym użytkowniku bez potrzeby w żadnych zdarzeniach.
3. Metoda ShowUserData() polega na tym, że tworzymy zapytanie, używając SqlCommand oraz stringu z zapytaniem do bazy danych (select from), po tym tworzymy obiekt klasy SqlDataReader, który będzie przechowywał rezultat wyszukiwań zapytania command. Następnym krokiem jest reader.Read(), co oznacza, że przeczytany został 1 wiersz z przechowywanych w reader informacji, a to jest informacja o naszym zalogowanym użytkowniku. Tworzymy zmienne typu string, do których przypisujemy znaczenia, otrzymane z konkretnych pól readera, które jeszcze potrzebują konwertacji na string przy pomocy polecenia ToString(). Po tych działaniach przypisujemy tekstom labelów, które znajdują się na wizualnej reprezentacji tej strony, wartości otrzymanych zmiennych tyout string.

3.13 UserProfileInformation

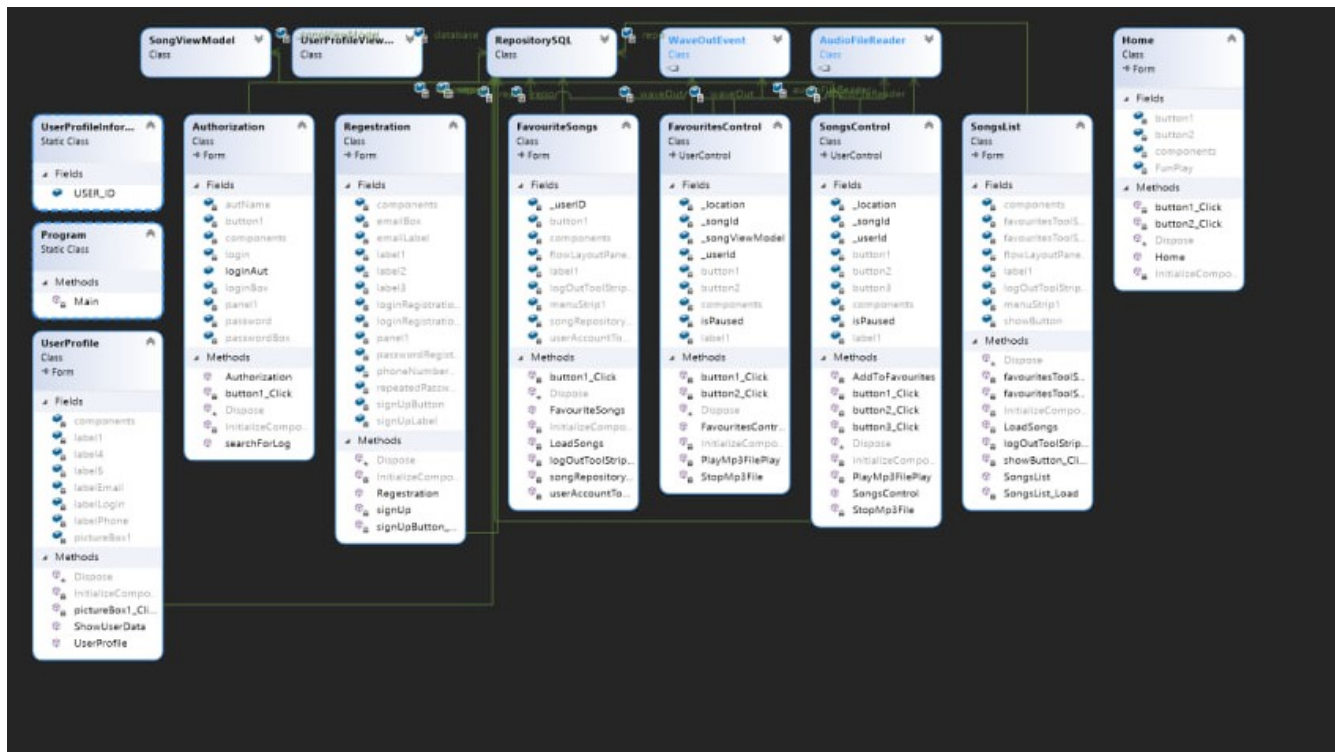
To jest statyczna klasa, jaka zawiera tylko i wyłącznie zmienną statyczną USER_ID, w której jest przechowywany id użytkownika, który teraz znajduje się w systemie. Jeżeli żaden użytkownik nie jest zalogowany, to USER_ID=-1.

3.14 Program

To jest klasa statyczna, która definiuje główny punkt wejściowy dla aplikacji Windows Forms. Dzięki temu aplikacja może rozpocząć działanie i wyświetlić interfejs użytkownika.

Metoda Main jest głównym punktem wejściowym dla aplikacji. Oznacza to, że jest to pierwsza metoda, która zostanie wywołana, gdy aplikacja zostanie uruchomiona.

4 Diagram klasów

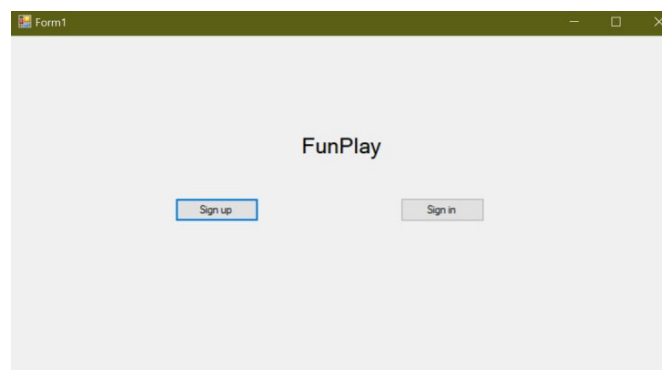


Rys. 1: Class Diagram

5 GUI

5.1 Home Page

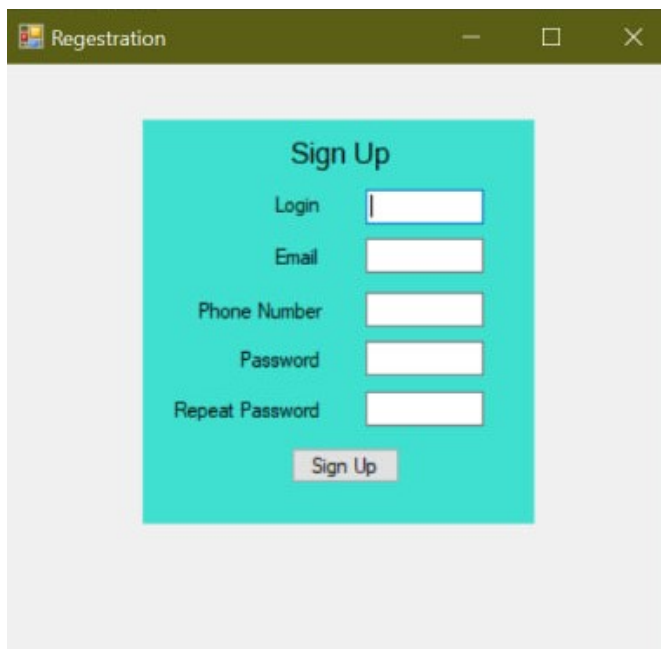
Home Page jest pierwszą stroną, którą widzi użytkownik po uruchomieniu programu. Ona posiada możliwość przekierowania użytkownika na stronę rejestracji czy logowania w zależności od przycisku, na jaki klika użytkownik ("Sign Up" i "Log In" odpowiednio).



Rys. 2: Home Page

5.2 Registration Page

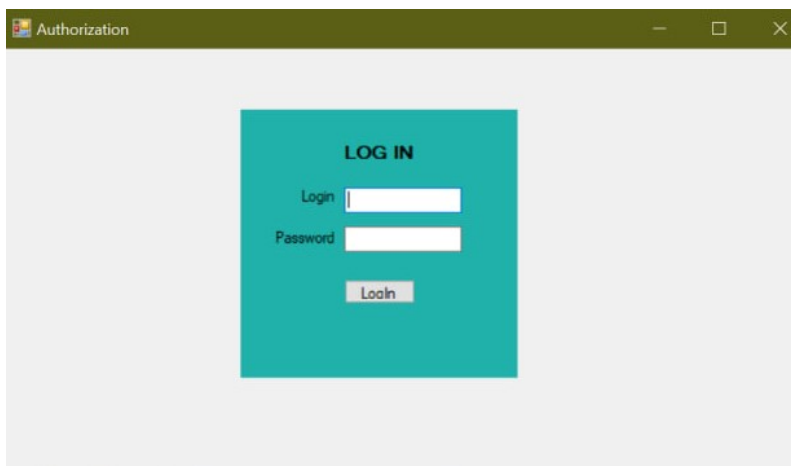
To jest strona rejestracji, niezbędne do uzupełnienia pola, żeby dodać użytkownika do systemu aplikacji. Po kliknięciu "Sign Up" użytkownik zostaje przekierowany do strony logowania.

The image shows a web application window titled "Registration". Inside the window is a light blue rectangular form titled "Sign Up". The form contains five input fields, each with a label to its left: "Login", "Email", "Phone Number", "Password", and "Repeat Password". Below these fields is a button labeled "Sign Up". The window has a standard Windows-style title bar with minimize, maximize, and close buttons.

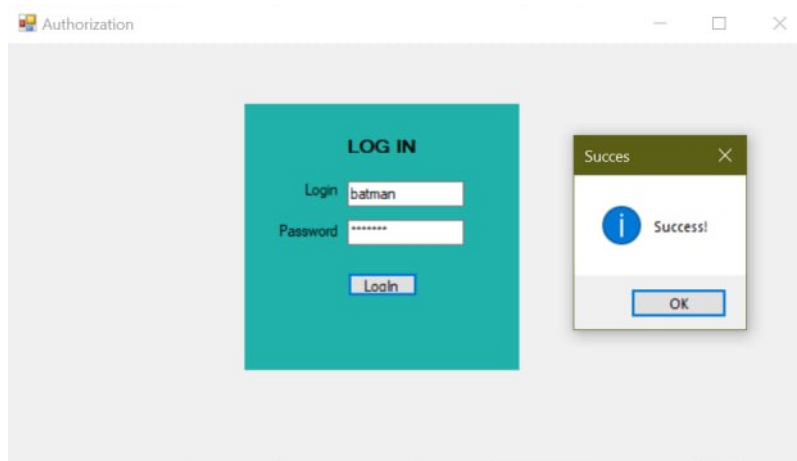
Rys. 3: Registration Page

5.3 Authorization Page

To jest strona, głównym celem której jest sprawdzenie, czy istnieje użytkownik w systemie, porównując wprowadzone w pola przez użytkownika loginem i hasłem z danymi z bazy danych. Jeżeli logowanie się udało, to użytkownik zostanie przekierowany do strony, na której są zlokalizowane wszystkie wspólne utwory systemu. Jeżeli nie udało się zalogować, to wyświetli się komunikat o błędnym logowaniu.

The image shows a web application window titled "Authorization". Inside the window is a light blue rectangular form titled "LOG IN". The form contains two input fields, each with a label to its left: "Login" and "Password". Below these fields is a button labeled "Login". The window has a standard Windows-style title bar with minimize, maximize, and close buttons.

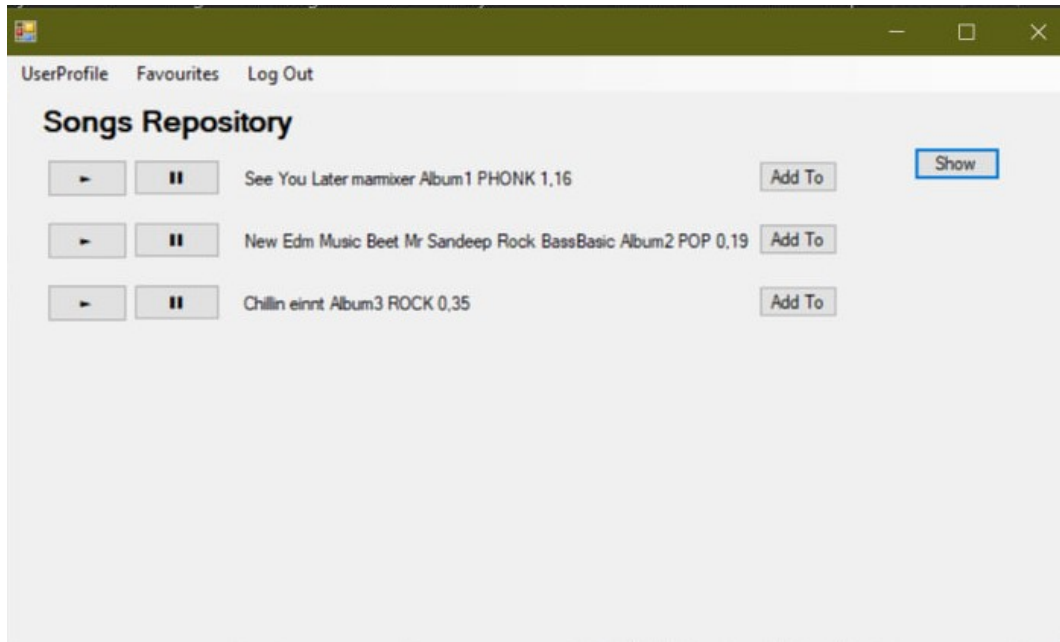
Rys. 4: Authorization Page



Rys. 5: Skuteczne logowanie

5.4 Songs Repository Page

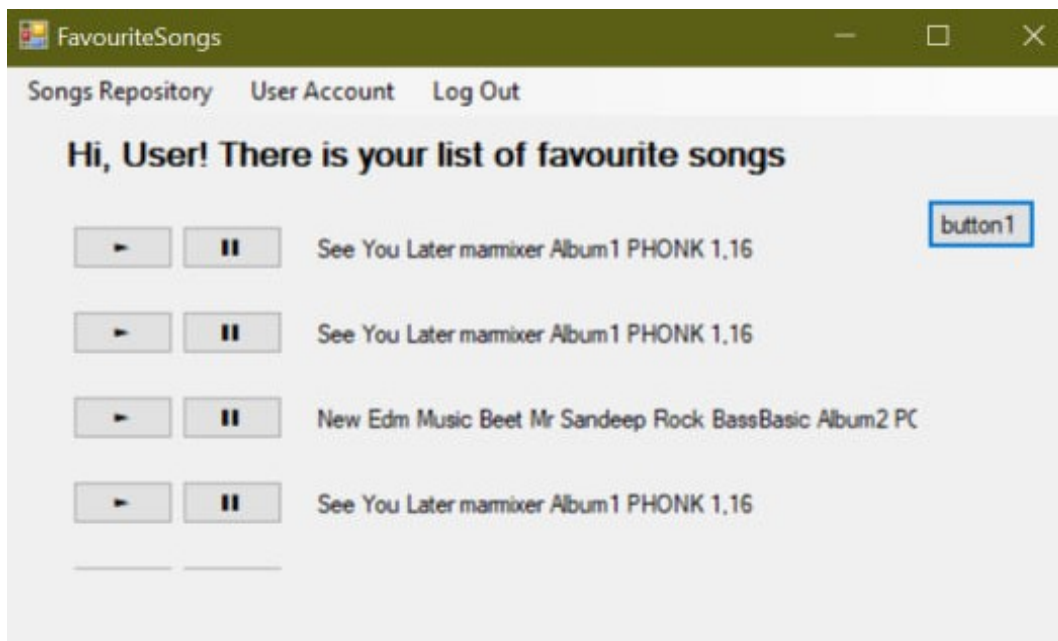
Ta strona po kliknięciu na przycisk "Show" wyświetla wszystkie wspólne utwory w bazie danych. Również na tej stronie znajdują przyciski, które pozwalają na odtwarzanie i zatrzymywanie każdego utworu, oraz przycisk dla dodawania piosenki do listy ulubionych. Menu zawiera przyciski, które udostępniają możliwość do nawigacji w tej aplikacji ("UserProfile" przekieruje do strony z danymi poptodznego użytkownika, "Favourites" do listy ulubionych utworów, "Log Out" wyloguje użytkownika z systemu i przekieruje na stronę domową).



Rys. 6: Songs Repository Page

5.5 Favourite Songs Page

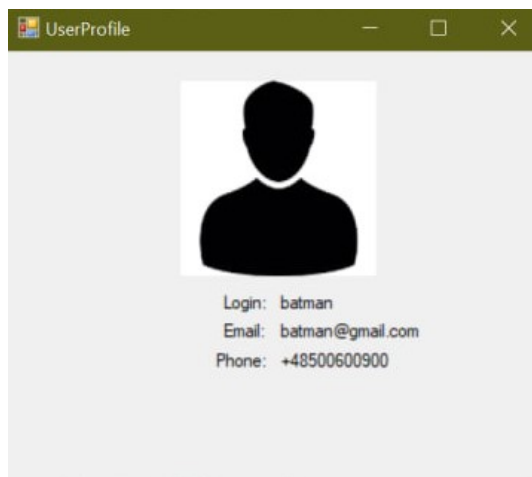
Ta strona wyświetla po kliknięciu na przycisk "Show" wszystkie ulubione utwory potocznego użytkownika. Również około każdej piosenki znajdują się przyciski do odtwarzania i zatrzymania utworów. Menu zawiera przyciski, które udostępniają możliwość do nawigacji w tej aplikacji ("User Account" przekieruje do strony z danymi potocznego użytkownika, "Songs Repository" do strony z repozytorium wszystkich utworów wspólnych, "Log Out" wyloguje użytkownika z systemu i przekieruje na stronę domową).



Rys. 7: Favourite Songs Page

5.6 UserProfile Page

Strona UserProfile wyświetla dane potocznego użytkownika oraz obrazek.



Rys. 8: UserProfile Page