

Aplikacja front-endowa StoreMusicApi:

Wykorzystano: **C#, WPF, .NET 6.0, Entity Framework, MS_SQL**, dokumentacje **Swagger**(główna strona), **Chmura Microsoft Azure, NLogger, Token JWT, Postman**, polityka **CORS, Middleware**.

Aplikacja desktopowa umożliwia przeglądanie (bez logowania) jako gość, wtedy można jedynie przeglądać wszystkich artystów, albumy i piosenki. Można się zalogować jako **USER** -> który może tylko przeglądać wszystko, bez dostępu do pola **ContactNumber** i **ContactEmail** danego Artysty. Następnie można zalogować się jako **PREMIUM_USER** -> możliwość tworzenia swoich Artystów o unikatowych nazwach, aby nie duplikowały się na liście już istniejącej u danego użytkownika. Tworzenie albumów jedynie dla swojego artysty i piosenek dla swojego albumu. Nazwy artystów, albumów, piosenek mogą się duplikować pomiędzy userami, czyli pierwszy user stworzy artystę o nazwie **Iron Maiden 1** o nr id -> **1**, a user drugi tworzy artystę o nazwie **Iron Maiden 1** z nr id -> **2**, taka sytuacja jest dopuszczalna. Można zalogować się jako **ADMIN** -> tworzenie, usuwanie, dodawanie, przeglądanie, tylko jeśli tworzy nazwę artysty, Albumu, Piosenki to musi być to nazwa unikatowa. W oknie logowania można się zarejestrować i po zalogowaniu każdy user ma wygenerowany **Token JWT**, poprzez którego sprawdza się autentykację, czy ma do danej czynności dostęp. Np. Użytkownik, który ma rolę **USER, PREMIUM_USER** gdy chce aktualizować niedodanego przez niego artystę, album, piosenkę, stworzyć nową piosenkę, album w niestworzonym przez niego artyście, wystąpi informacja o błędzie -> Status code: 403 -> Forbidden. Jeśli zalogowany jako gość będzie chciał mieć dostęp do opcji zabronionej dla niego to dostaje komunikat, że nie jest zalogowany i ma brak dostępu, bo nie posiada danego Tokena JWT.

W oknie **User Details** -> są informacje szczegółowe danego usera i utworzeni przez niego artyści, plus dane dodatkowe **ContactEmail** i **ContactNumber**.

W oknie **Show Artists** można dodawać, aktualizować, usuwać, zobaczyć szczegóły danego artysty. Można użyć paginacji: **Record per page** -> **5, 10, 15** ilość rekordów na danej stronie. **1 of 30** -> **1**-> strona aktualna, **30**-> ilość stron wszystkich, strona aktualna zmienia się względem wybrania danej strony dodatkowymi przyciskami '**>**' -> **nextPage**, '**<**' -> **PreviousPage**, '**>>**' -> **LastPage**, '**<<**' -> **FirstPage**. Możliwość sortowania za pomocą **Sort By** -> względem 3 kolumn: **Name, Description, KindOfMusic, SortDirection** -> **ASC**: rosnąco, **DESC**: malejąco sortuje po danej kolumnie wybranej. Przycisk **Search** -> szuka danego słowa po kolumnie **Name**. Wszystko działa po wciśnięciu przycisku **Search**.

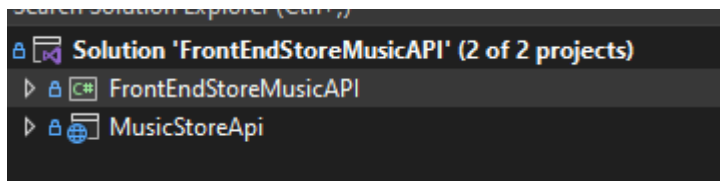
W oknie **Show Albums i Songs** działa podobnie, dodatkowo można użyć przycisku **Delete All** -> usuwając wszystkie rekordy. W oknie **Show All Albums** -> wyświetlają się automatycznie wszystkie albumy z bazy danych. W oknie **Show All Songs** -> wyświetlają się automatycznie wszystkie piosenki z bazy danych. W oknie **Show All Artists** -> wyświetlają się automatycznie wszyscy artyści z bazy danych. Gdy wraca się z okna **ShowAllSongsWindow** do okna **ShowAlbumsWindow**, to automatycznie wyświetlają się wszystkie albumy danego artysty. Użycie okna **UpdateCreateArtist, UpdateCreateAlbum, UpdateCreateSong** -> do dwóch działań, do tworzenia i aktualizowania. Użycie okna **ShowAllAlbumsWindow** do wyświetlania wszystkich albumów z bazy danych lub do wyświetlania wszystkich albumów danego artysty. Użycie okna **ShowAllSongsWindow** do wyświetlania wszystkich piosenek z bazy danych lub do wyświetlania wszystkich piosenek danego albumu.

Podczas najechania kursorem myszki na jakikolwiek przycisk, pole tekstowe, pojawia się dodatkowa informacja za co odpowiada dany przycisk. Możliwość połączenia się aplikacji Frontendowej bezpośrednio z aplikacją webową, która jest na **chmurze Azure**, pod linkiem : <https://musicstore->

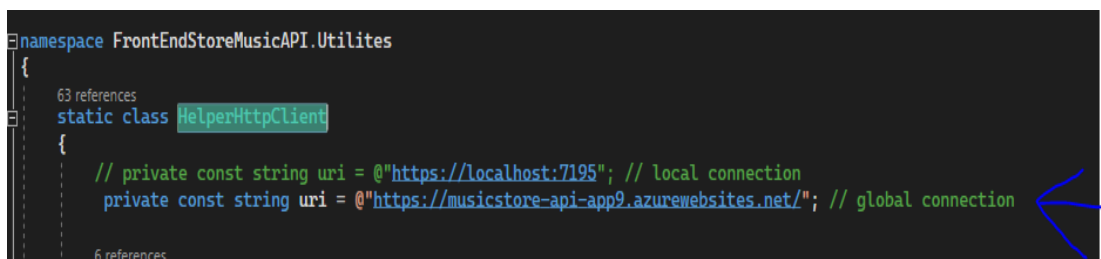
api-app9.azurewebsites.net , trzeba jedynie zmienić pole **uri** w katalogu

Utilites/HelperHttpClient.cs. Podczas pierwszego uruchomienia lokalnie utworzą się rekordy do testowania i 3 userów też do testowania.

Aplikacja desktopowa została stworzona, aby łączyła się z aplikacją backendową **MusicStoreApi** na 3 sposoby:



1. Łączy się bezpośrednio z **web API** , które jest razem z aplikacją Frontendową w katalogu projektu, tylko trzeba ustawić wartość zmiennej **uri** na prawidłowe połączenie lokalne: <https://localhost:numerPortu> -> numer portu, na którym jest uruchomiona aplikacja web API, to można sprawić w katalogu **MusicStoreApi/Properties/launchSettings.json**, najlepiej uruchomić za pomocą **HTTPS**, wtedy działa aplikacja bez problemu.
2. Łączy się z tym samym **web API** , które jest uruchomione jako inny projekt na komputerze lokalnym oraz zmieniając zmienną uri jak opisane wyżej i uruchamia za pomocą **HTTPS**.

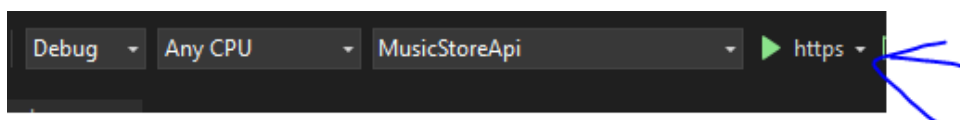


3. Łączy się z tym samym **web API**, które jest uruchomione w chmurze: tylko trzeba ustawić prawidłową wartość zmiennej uri na <https://musicstore-api-app9.azurewebsites.net/swagger/index.html> , która znajduje się w katalogu **Utilites/HelperHttpClient.cs** .

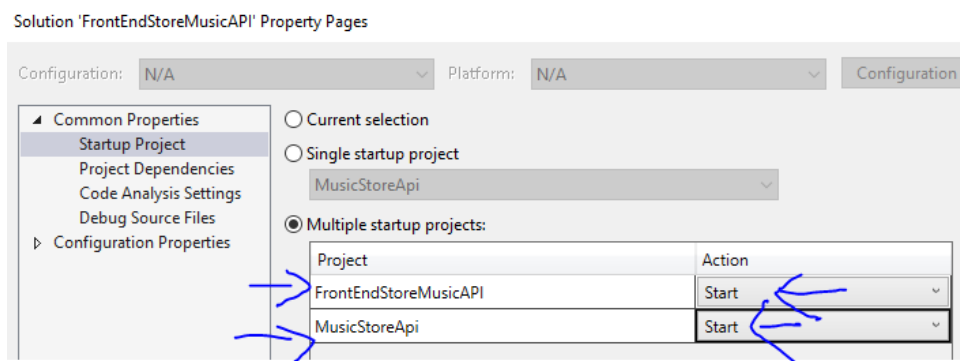
Opis działania aplikacji:

1. Start aplikacji:
 - Pobierz z githuba -> **Code** -> **Download Zip**.
 - Otwórz aplikację w **Visual Studio Community 2022**

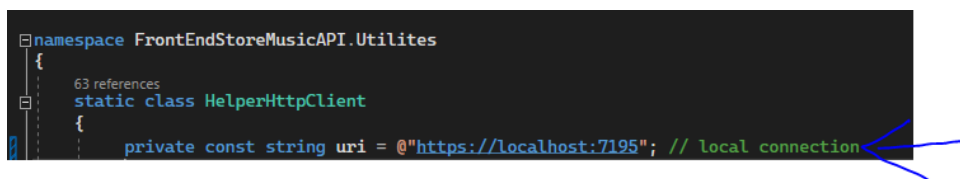
Uruchomienie z **Web Api** , które znajduje się w projekcie, to:



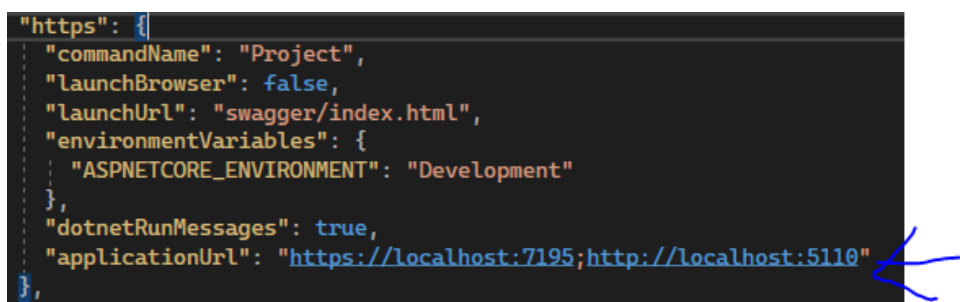
- Upewnij się że uruchomienie **Web Api** masz na **HTTPS**.
- Potem wejdź w **Https-> Configure Startup Projects**



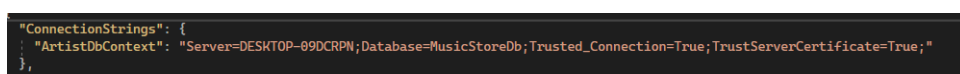
- Ustaw oba projekty na **Start**.



- Wejdź w katalog **FrontEndStoreMusicAPI/Utilites/HelperHttpClient.cs**.



- Upewnij się że masz prawidłowy adres zmiennej **uri**, który możesz sprawdzić w katalogu **MusicStoreApi/Properties/LaunchSettings.json** w **HTTPS**.



- Wprowadź prawidłowe połączenie z bazą danych w katalogu **MusicStoreApi/appsettings.Development.json**

- Potem wejdź w konsolę **Package Manager Console** i wprowadź komendę **update-database**, jeśli brak migracji, to **add-migration Init**, potem **update-database**.
- Jeśli wszystko działa poprawnie, to uruchom aplikację. Przy pierwszym uruchomieniu, utworzą się przykładowe dane w bazie danych, aby móc testować i korzystać z aplikacji.

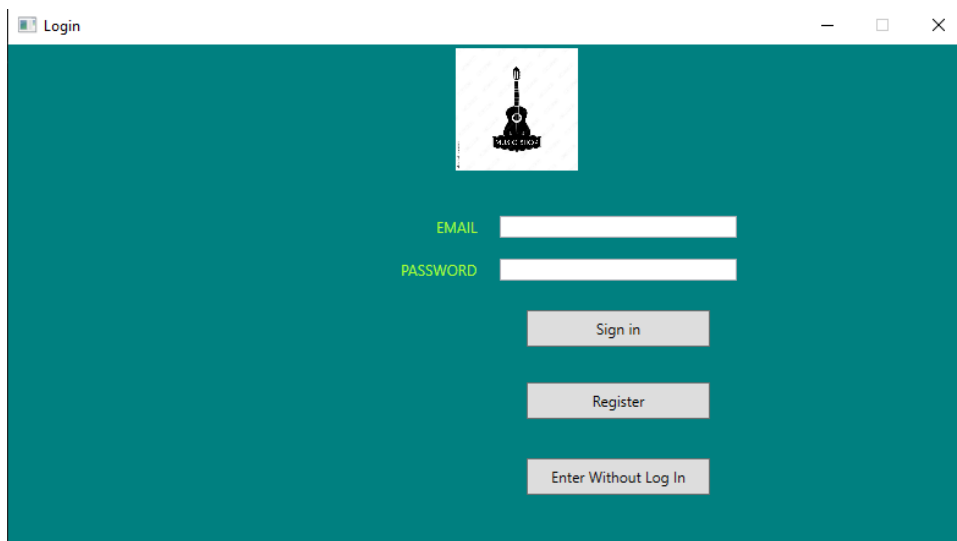
Uruchomienie z **Web Api**, które znajduje się w **chmurze Azure**, to:

```
private const string uri = @"https://musicstore-api-app9.azurewebsites.net/"; // global connection
```

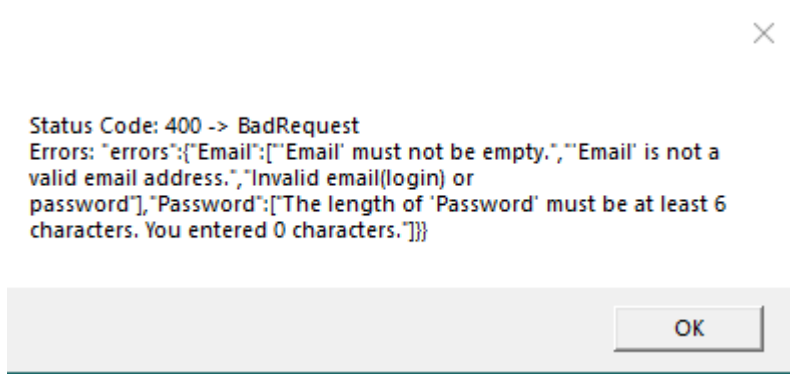
6 references

- Upewnij się że masz prawidłowy adres zmiennej uri na <https://musicstore-api-app9.azurewebsites.net>
- Uruchom tylko aplikację Frontendową i testuj aplikację.
- Jeśli wystąpi błąd **"Microsoft.Data.SqlClient.SqlException: 'There is already an object named 'Addresses' in the database.'"** -> to w **Package Manager Console** wprowadź komendy: **drop-database**, potem **update-database**, jak wszystko przeszło ok, to uruchom aplikację tylko frontendową.

Uruchomiona aplikacja:



1. **Okno logowania:**
 - Aby móc się zalogować trzeba podać **email** i **password**

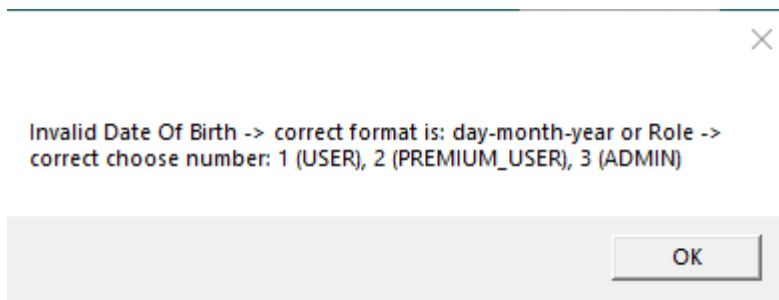


- Jeśli nic się nie poda to wyskoczy okno informacyjne o danych błędach:

Email -> nie może być pusty, sprawdzane jest czy jest poprawny adres, Invalid email or password: informuje o nieprawidłowym emailu.

Password -> musi mieć przynajmniej 6 znaków, informuje jeśli password jest nieprawidłowe.

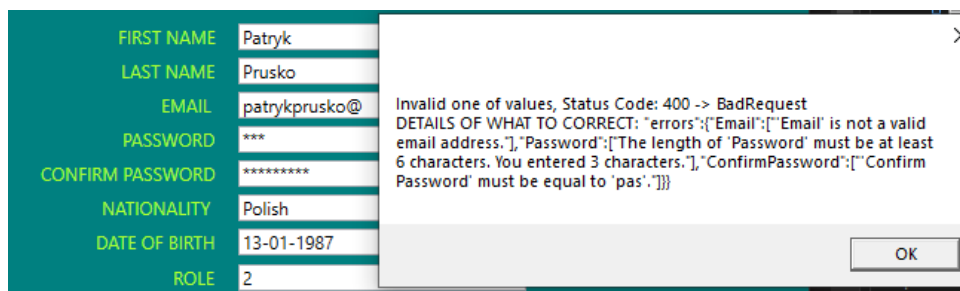
- Przycisk **Register** -> otwiera nowe okno rejestracji, gdzie można stworzyć swoje nowe konto i zostać dodanym jako nowy użytkownik do bazy danych.



- Okno informacyjne o błędach:

DateOfBirth -> musi mieć format **day-month-year** np. 13-01-1987.

Role -> **1, 2, 3 (USER, PREMIUM_USER, ADMIN)**.

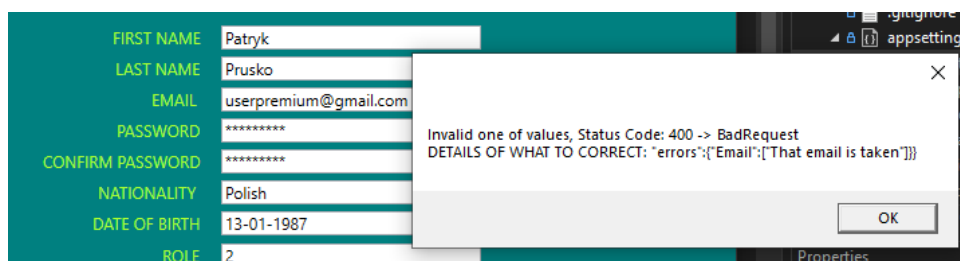


The screenshot shows a registration form with the following fields: FIRST NAME (Patryk), LAST NAME (Prusko), EMAIL (patrykprusko@), PASSWORD (***), CONFIRM PASSWORD (*****), NATIONALITY (Polish), DATE OF BIRTH (13-01-1987), and ROLE (2). A modal dialog box is displayed over the form, showing an error message: "Invalid one of values, Status Code: 400 -> BadRequest. DETAILS OF WHAT TO CORRECT: 'errors':{'Email':['Email is not a valid email address.'], 'Password':['The length of 'Password' must be at least 6 characters. You entered 3 characters.'], 'ConfirmPassword':['Confirm Password' must be equal to 'pas'.']}}". An "OK" button is at the bottom right of the dialog.

Następne okno informacyjne o błędach:

Email -> sprawdza poprawność,

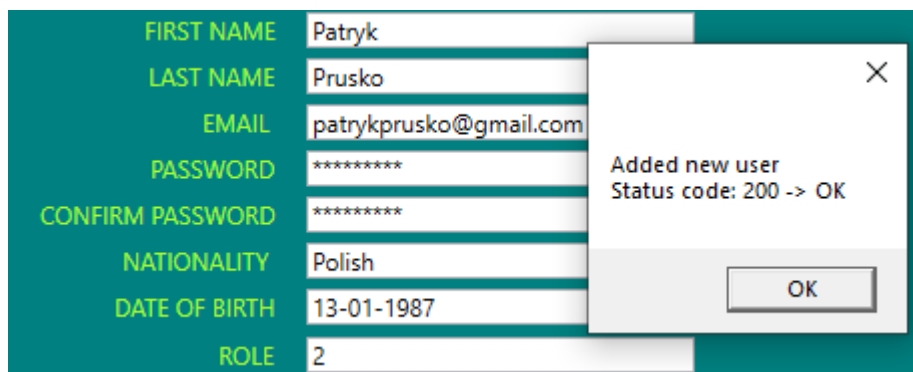
Password -> musi mieć przynajmniej 6 znaków, musi być taki sam jak **ConfirmPassword**.



The screenshot shows the same registration form, but the EMAIL field now contains "userpremium@gmail.com". The modal dialog box displays a different error message: "Invalid one of values, Status Code: 400 -> BadRequest. DETAILS OF WHAT TO CORRECT: 'errors':{'Email':['That email is taken']}}". An "OK" button is at the bottom right of the dialog.

Następne okno informacyjne o błędach:

Jeśli wprowadzi się emaila, który już występuje w bazie danych wyświetla się informacja, że już jest zajęty.



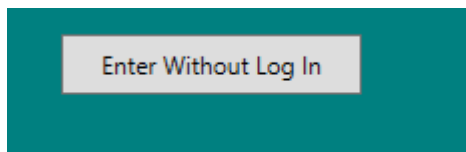
The screenshot shows the registration form with the EMAIL field containing "patrykprusko@gmail.com". A modal dialog box is displayed with the message: "Added new user. Status code: 200 -> OK". An "OK" button is at the bottom right of the dialog.

Dodanie nowego użytkownika:

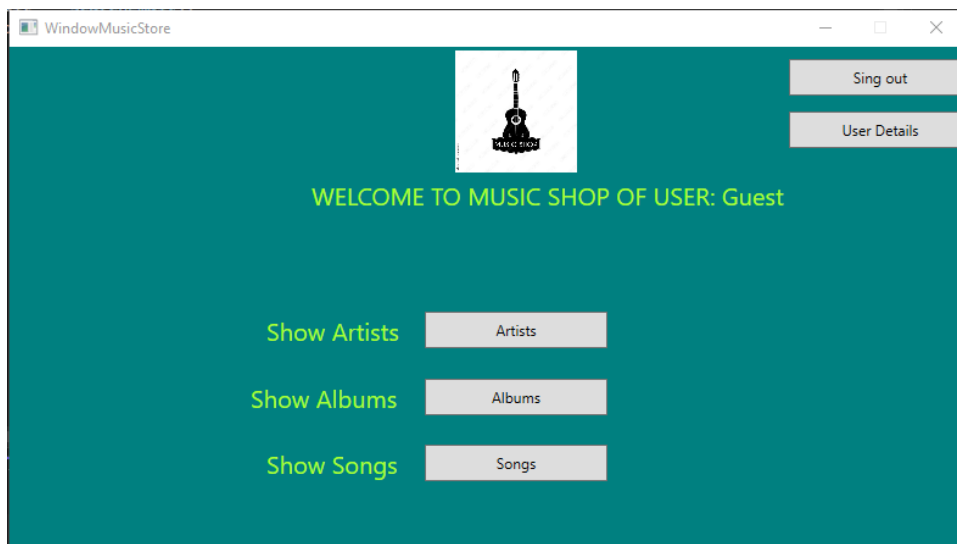
Jeśli wszystko jest poprawne to pojawia się informacja, że dodano nowego usera i **status code 200**.

Potem przechodzi do okna logowania. Przycisk **Return** -> powrót to okna logowania, przycisk **Clear**-> czyści wszystkie pola, przycisk **Save** -> tworzy i zapisuje nowego usera.

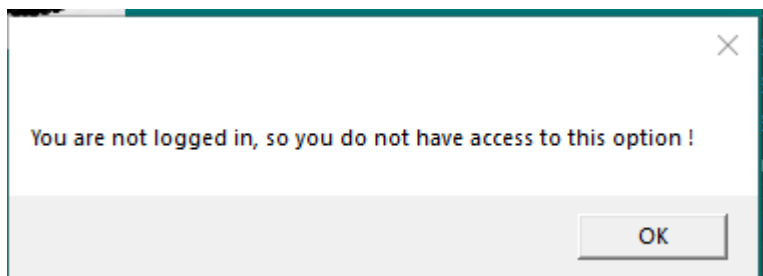
Działanie logowania jako **GOŚĆ**:



- Przycisk **Enter Without Log In** -> loguje jako gość, użytkownika niezalogowanego.



Otwiera się następne główne okno Sklepu Muzycznego. Nazwa usera Jako: **Guest**.



Jeśli próba korzystania z przycisku **User Details** to pojawi się komunikat -> **You are not logged in, so you do not have access to this option**. Możliwość jedynie przeglądania **Artystów, Albumów, Piosenek**.

Działanie jako użytkownik zalogowany:

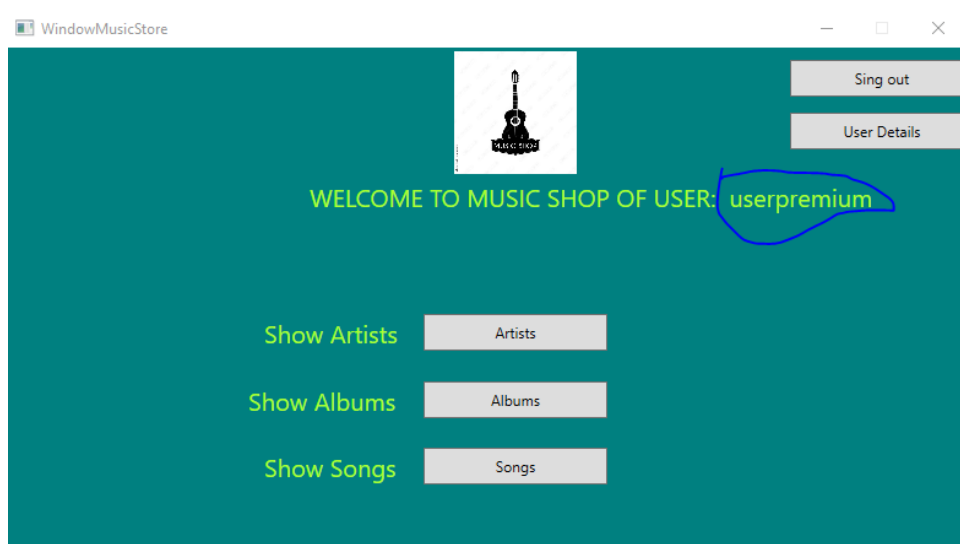
Są 3 opcje jako:

- **USER** -> możliwość tylko przeglądania ogólnego oraz szczególnego artystów, albumów, piosenek.
- **USER_PREMIUM** -> możliwość przeglądania, tworzenia nowych artystów, lecz o unikatowej nazwie, aby się nie powtarzała na liście już stworzonych przez danego usera. Możliwość tworzenia albumów, tylko do artystów, których dany user stworzył. Możliwość tworzenia

piosenek do albumów, które dany user stworzył. Usuwanie i aktualizowanie artystów, albumów i piosenek jedynie stworzonych przez danego usera. Dostęp jedynie ma dany user_premium do danych **ContactEmail** i **ContactNumber** w oknie **Details of Artist** lub oknie **User Details** do danych Artysty, którego stworzył.

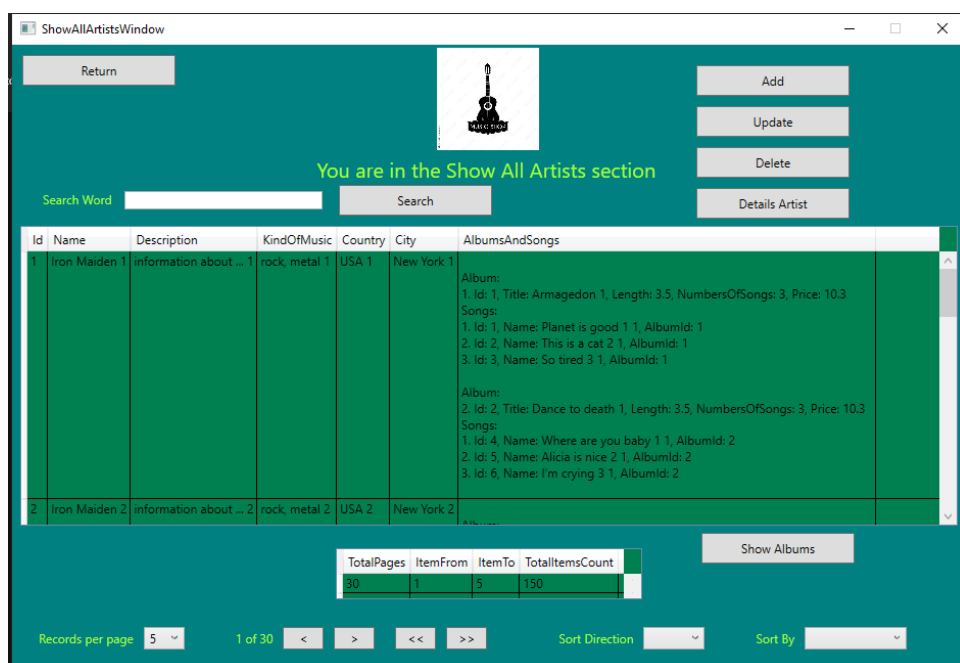
- **ADMIN** -> możliwość przeglądania, brak dostępu ContactEmail i ContactNumber danego Artysty, jeśli go nie stworzył. Jeśli chodzi o tworzenie, aktualizowanie to nazwy mają być unikatowe.

Działanie okna głównego Music Shop of User:



Jeśli się zalogujesz jako np. **User_premium** to u góry zawsze pojawi się twoje **FirstName**.

Działanie okna **Show Artists**:



Od razu generuje się lista wszystkich artystów z bazy danych. Użycie **paginacji** :

Records per page -> wybór **5, 10, 15** rekordów na daną stronę. **NextPage**-> następna strona, **PreviousPage**-> wcześniejsza strona, **FirstPage**-> pierwsza strona, **LastPage**-> ostatnia strona. **Sort Direction**-> **ASC**(rosnąco), **DESC**(malejąco) sortowanie, aby działało to musi być wybrane **Sort By** -> **Name, Description, Kind Of Music**. **TotalPages** -> ilość stron do wyświetlenia, **totalItemsCount**-> wynik wszystkich rekordów. **1 of 30** -> **1**-> strona aktualna, **30** -> ilość stron do przeglądania, podczas zmieniania aktualnej strony wyświetla się, na której stronie jest się obecnie.

Przycisk **Search** -> szukane słowo szuka w wartości Name. Aby ustawić wszystkie zmiany trzeba użyć tego przycisku.

DetailsArtist

Return


You are in the Details of Artist section

Id	Name	Description	KindOfMusic	Country	City	ContactEmail	ContactNumber	AlbumsSongs
95	Sepultura 95	information about ... 95	rock, metal 95	USA 95	New York 95	contact@email95.com	9874238595	Album: 1. Id: 189, Title: Armagedon 95, Length: 3.5, Nun Songs: 1. Id: 565, Name: Planet is good 1 95, AlbumId: 189 2. Id: 566, Name: This is a cat 2 95, AlbumId: 189 3. Id: 567, Name: So tired 3 95, AlbumId: 189 Album: 2. Id: 190, Title: Dance to death 95, Length: 3.5, 1 Songs: 1. Id: 568, Name: Where are you baby 1 95, Albu 2. Id: 569, Name: Alicia is nice 2 95, AlbumId: 19 3. Id: 570, Name: I'm crying 3 95, AlbumId: 190

Przycisk **Details Artist** -> trzeba zaznaczyć jakiś rekord i pokazuje szczegóły danego artysty w nowym oknie.

UpdateCreateArtist

Return



You are in the Create Artist section

NAME

DESCRIPTION

KIND OF MUSIC

CONTACT EMAIL

CONTACT PHONE

COUNTRY

CITY

Save

Clear

Przycisk **Add** -> dodaje nowego artystę.

×

Status Code: 400 -> BadRequest
Errors: "errors":{"City":["The City field is required."],"Name":["The Name field is required."],"Country":["The Country field is required."],"Description":["The Description field is required."],"ContactEmail":["The ContactEmail field is not a valid e-mail address."],"ContactNumber":["The ContactNumber field is not a valid phone number."]}}

OK

Informacje o błędach:

Name -> nie może być puste.

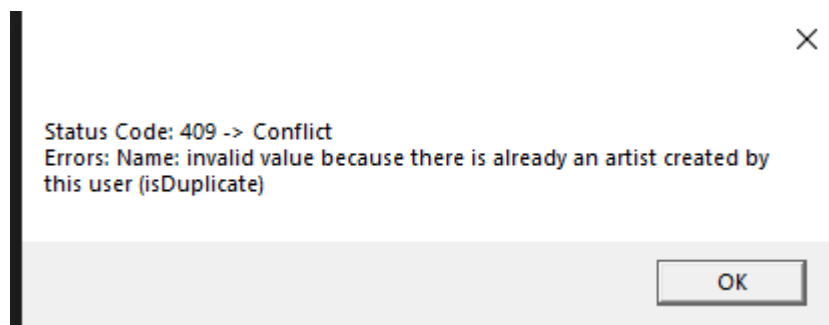
City -> nie może być puste.

Country -> nie może być puste.

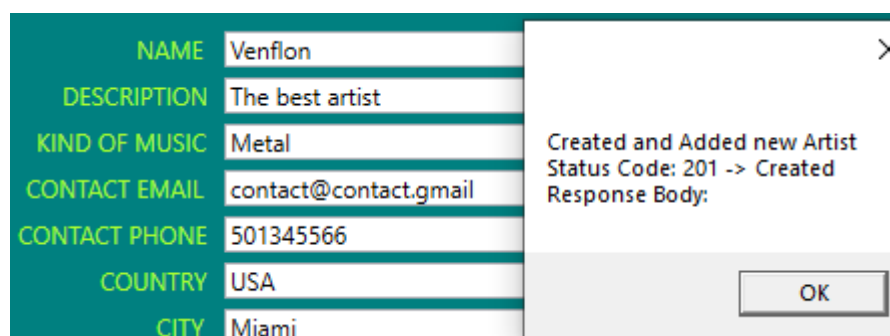
Description -> nie może być puste.

ContactNumber -> jest nieprawidłowy, powinny być liczby.

ContactEmail -> jest nieprawidłowy adres.

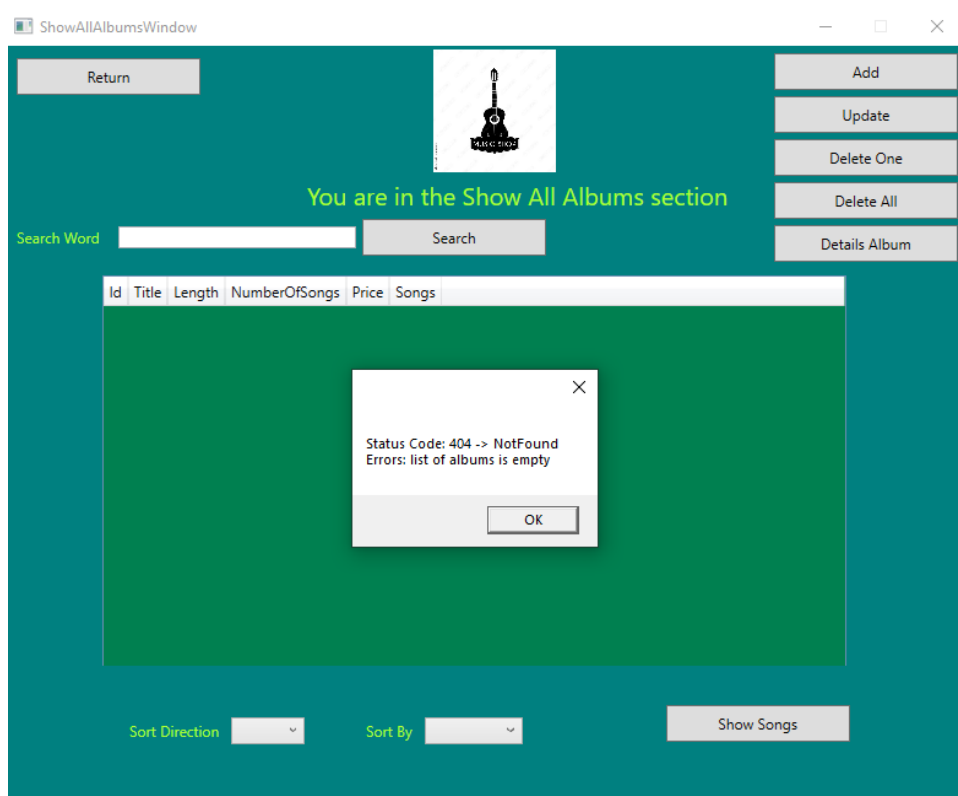


Przy tworzeniu nazwy albumu która już jest na liście danego usera, tworzy duplikat nazwy, czyli wyświetla się błąd.

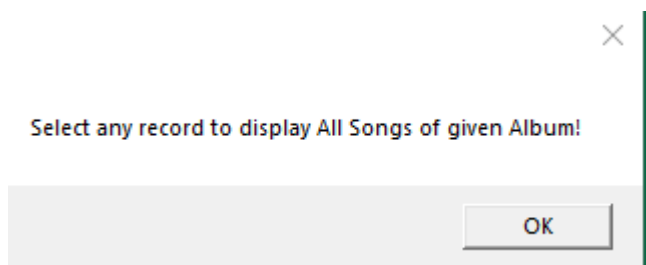


All Artists						
Id	Name	Description	KindOfMusic	Country	City	AlbumsAndSongs
151	Venflon	The best artist	Metal	USA	Miami	

Utworzony prawidłowo, potem przechodzi do okna **All Artists**



Wchodzimy do nowego okna dzięki przyciskowi **Show Songs** i dostajemy informację, że brak listy Albumów.



Komunikat błędu, jeśli wchodzimy w listę piosenek, której jeszcze nie ma w nieistniejącym albumie.

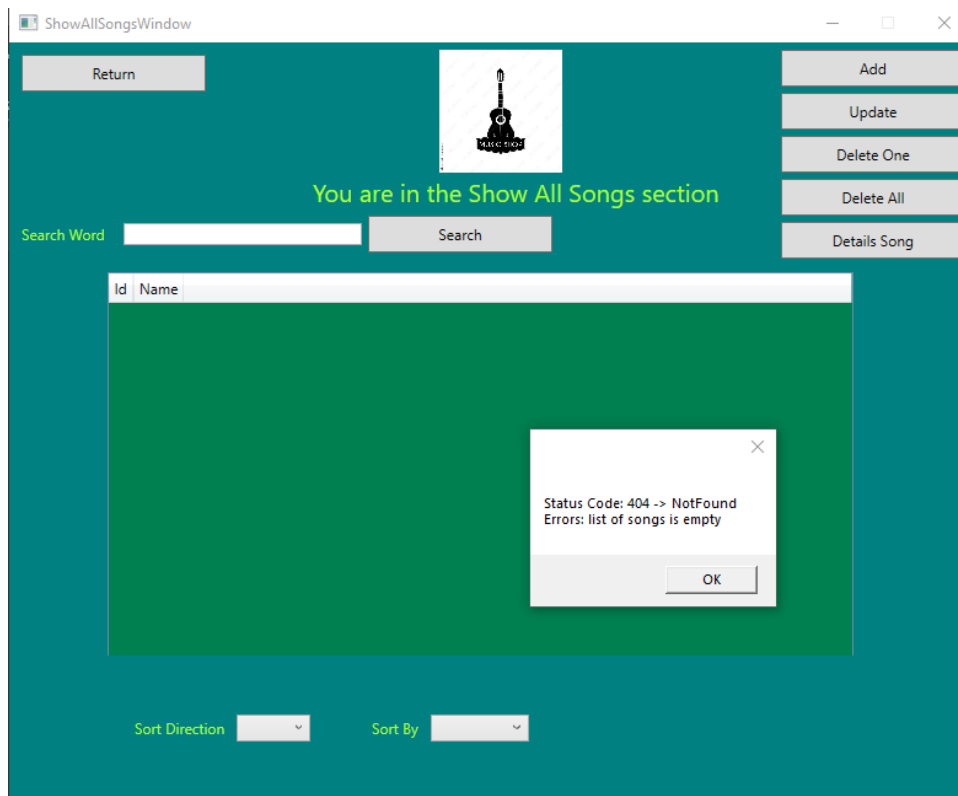
Tworze nowy album przyciskiem **Add**.

Lista błędów -> **Price** : musi być cyfrą: 33 lub 15.5, **Length** -> też musi być cyfrą.

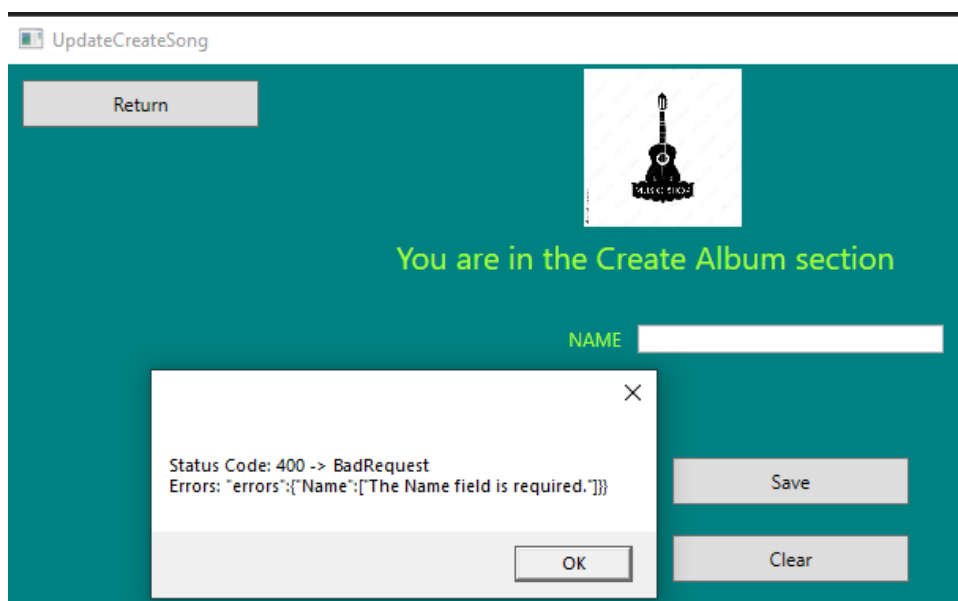
Stworzenie nowej piosenki. Potem wraca automatycznie do okna **All Albums**.

Id	Title	Length	NumberOfSongs	Price	Songs
301	Armagedon	3.5	0	15.8	
302	Second album	22	0	44	
303	second	213	0	22	

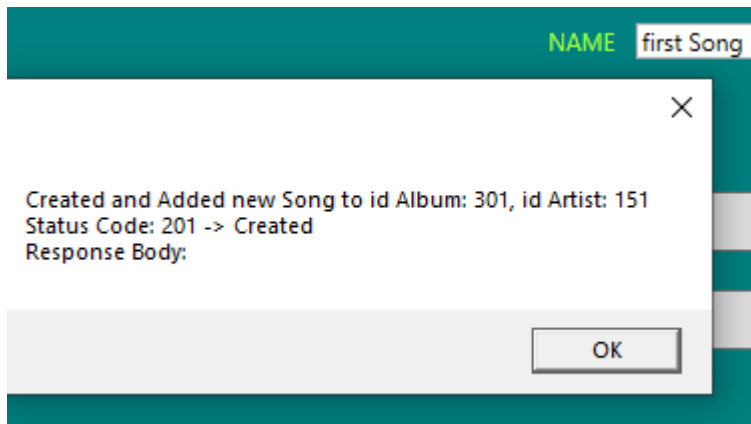
W oknie **All Albums** można sortować **ASC** ub **DESC** za pomocą **Sort By** -> wartości **Title**. Przycisk **Search** -> szuka po nazwie **Title**.



Przycisk **Show Songs** -> przechodzi do okna piosenek danego albumu i informacja, że brak listy piosenek.

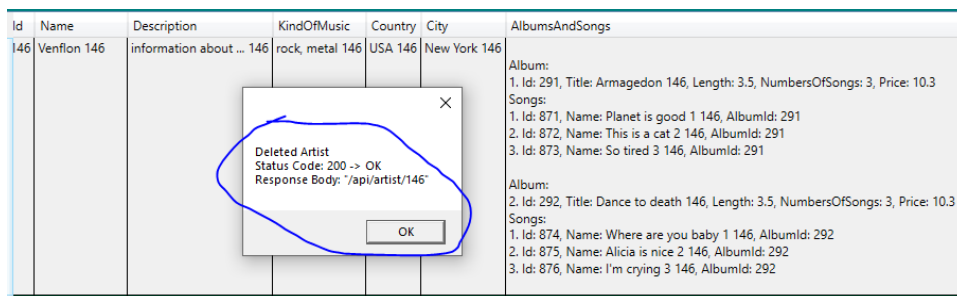


Przy tworzeniu nowej piosenki informacja błędu, że **Name** -> nie może być puste.

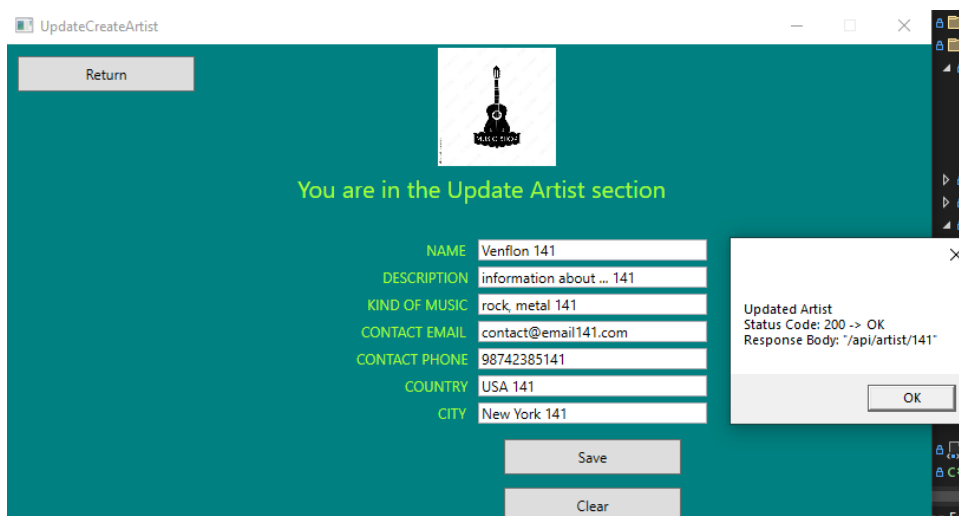


Id	Name	
901	first Song	
902	second song	
903	third song	

Utworzenie nowej piosenki i wraca do okna **All Songs**. Sortowanie działa tak samo jak w oknie **All Albums**.



Działanie przycisku **Delete** w oknie **Show All Artists**, informacja o usunięciu i danej ścieżce, gdzie się wcześniej znajdował artysta.



W oknie **All Artist** przycisk **Update** -> aktualizuje i zaciąga aktualne danego artysty i wraca do okna **All Albums**, gdy wszystko przeszło bez problemów.

You are in the Update Artist section

NAME	Iron Maiden 2
DESCRIPTION	information about ... 1
KIND OF MUSIC	rock, metal 1
CONTACT EMAIL	contact@email1.com
CONTACT PHONE	987423851
COUNTRY	USA 1
CITY	New York 1

Status Code: 409 -> Conflict
 Errors: Name: invalid value because there is already an artist created by this user (isDuplicate)

OK

Jeśli wystąpi próba duplikatu nazwy albumu, który występuje już na liście danego usera, to pojawi się powiadomienie.

DetailsArtist

Return

You are in the Details of Artist section

Id	Name	Description	KindOfMusic	Country	City	ContactEmail	ContactNumber	AlbumsSongs
1	Iron Maiden 1	information about ... 1	rock, metal 1	USA 1	New York 1	contact@email1.com	987423851	<p>Album:</p> <p>1. Id: 1, Title: Armagedon 1, Length: 3.5, NumbersO</p> <p>Songs:</p> <p>1. Id: 1, Name: Planet is good 1 1, AlbumId: 1</p> <p>2. Id: 2, Name: This is a cat 2 1, AlbumId: 1</p> <p>3. Id: 3, Name: So tired 3 1, AlbumId: 1</p> <p>Album:</p> <p>2. Id: 2, Title: Dance to death 1, Length: 3.5, Numbe</p> <p>Songs:</p> <p>1. Id: 4, Name: Where are you baby 1 1, AlbumId: 2</p> <p>2. Id: 5, Name: Alicia is nice 2 1, AlbumId: 2</p> <p>3. Id: 6, Name: I'm crying 3 1, AlbumId: 2</p>

W oknie **All Artist** przycisk **Details Artist** -> otwiera nowe okno **Details of Artist**. Kolumny **ContactEmail** i **ContactNumber** -> są wyświetlane tylko dla usera, który stworzył dany album, dla innych są zastrzeżone.

W oknach **All Albums** i **All Songs** wszystko działa podobnie, jedynie dodatkowo dodany jest przycisk **Delete All** -> który usuwa wszystkie dane albumy lub piosenki, potem pojawia się informacja że lista albumów, piosenek jest pusta.

Id	Name
1	Planet is good 1 1
2	This is a cat 2 1
3	So tired 3 1

Deleted Songs
 Status Code: 200 -> OK
 Response Body: "Deleted all songs: /api/artist/1/album/1"

OK

Przed użyciem przycisku **Delete All** w oknie **All Songs**.

Id	Name

Status Code: 404 -> NotFound
 Errors: list of songs is empty

OK

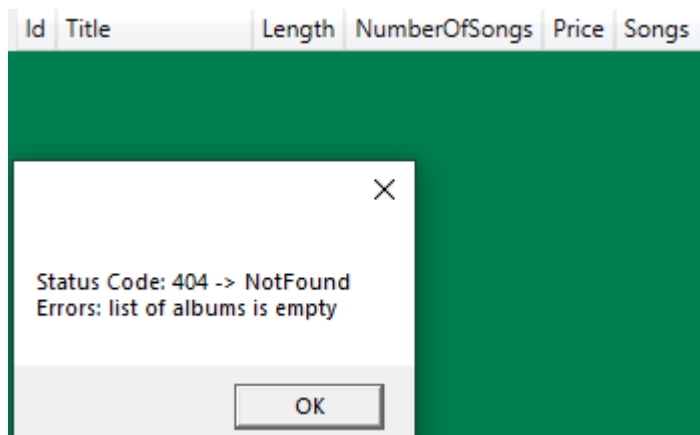
Po użyciu.

Id	Title	Length	NumberOfSongs	Price	Songs
1	Armagedon 1	3.5	0	10.3	
2	Dance to death 1	3.5	3	10.3	1. Id: 4, Name: Where are you baby 1 1, AlbumId: 2 2. Id: 5, Name: Alicia is nice 2 1, AlbumId: 2 3. Id: 6, Name: I'm crying 3 1, AlbumId: 2

Deleted Albums
 Status Code: 200 -> OK
 Response Body: "Deleted all albums: api/artist/1"

OK

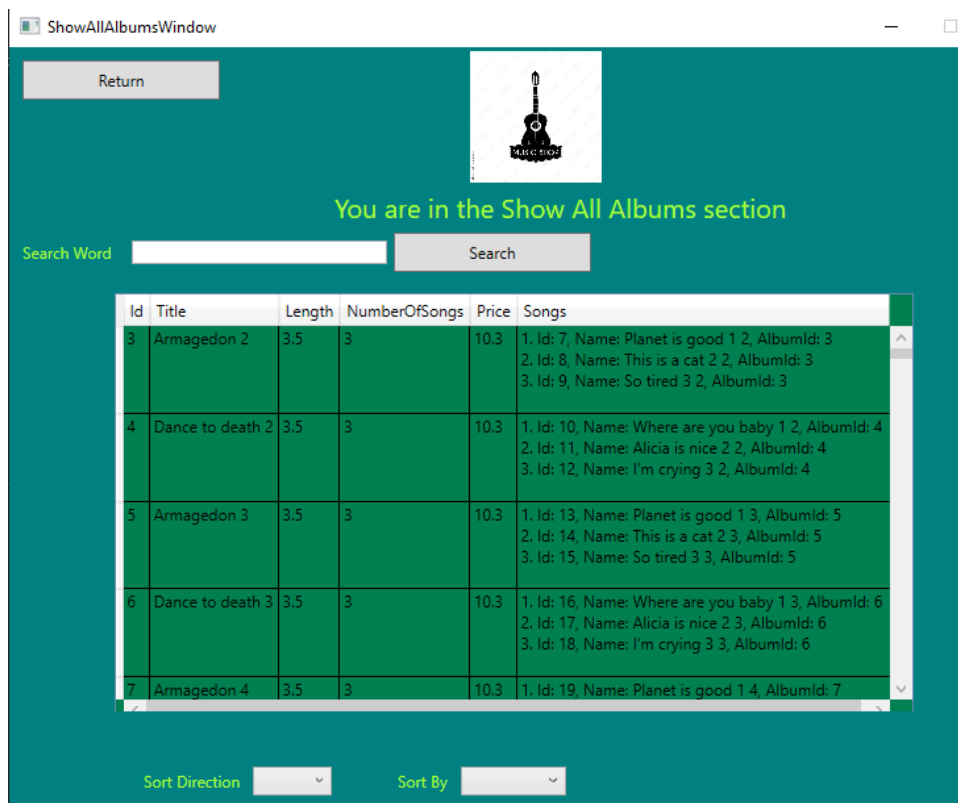
Przed użyciem przycisku **Delete All** w oknie **All Albums**.



Po użyciu.

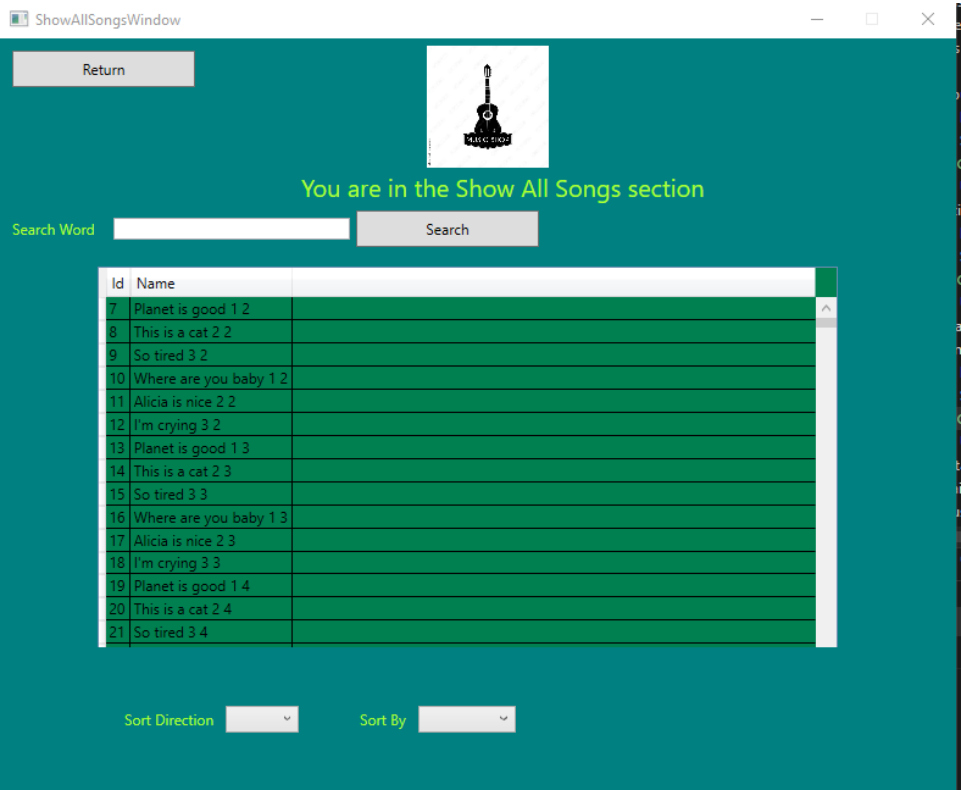


W oknie **Music Shop** są dwa przyciski:

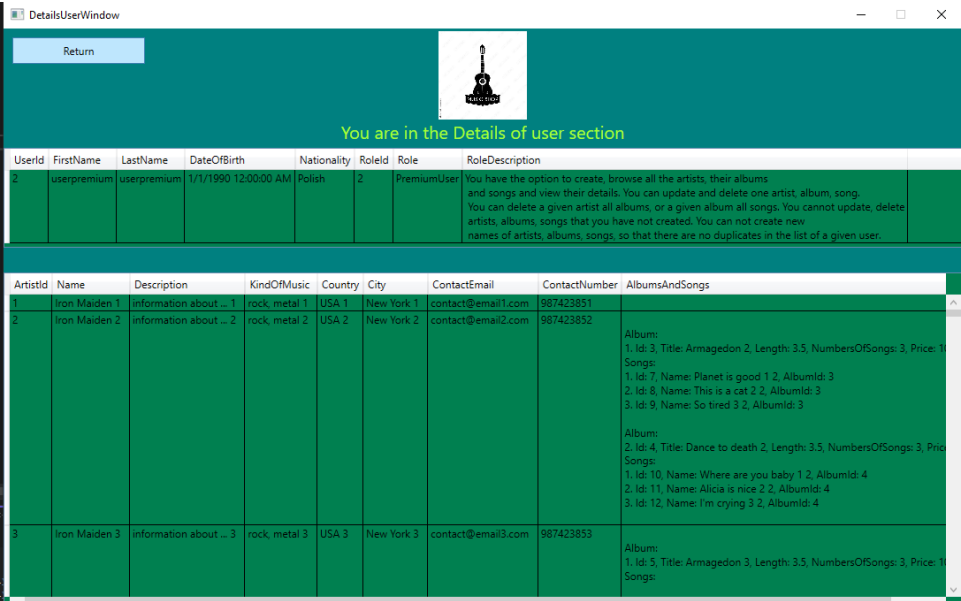


Przycisk **Albums** -> wchodzi w nowe okno, które pokazuje wszystkie albumy z bazy danych.

Przycisk **Search** -> wyszukuje po nazwie **Title**, **Sort Direction** -> **ASC**, **DESC** sortuje. **Sort By** -> sortuje po wartości **Title**.



Przycisk **Songs**-> wchodzi w nowe okno, które pokazuje wszystkie piosenki z bazy danych oraz można sortować po kolumnie **Name**.



W oknie Music Shop znajduje się też dodatkowy przycisk **User Details**-> który otwiera nowe okno **Details of User** i pokazuje szczegóły danego usera, czyli w pierwszej tabelce są informacje

szczegółowe danego usera , a na drugiej są wszyscy Artyści, ich albumy, ich piosenki stworzone przez tego usera.

Dodatkowo utworzone / zmienione rzeczy w Aplikacji Backend :

1. Dodanie nowego endpointa -> **GET api/login/user/{email}** -> aby mieć dostęp do danych szczegółowych użytkownika i wygenerowany **Token JWT**, po zalogowaniu do głównego okna Music Store.
2. Dodanie nowego endpointa -> **GET api/login/user/{userId}/artist** -> aby mieć dostęp po zalogowaniu do głównego okna Music Store do wszystkich artystów stworzonych przez danego użytkownika oraz szczegółowych informacji do wyświetlenia.
3. Dodanie nowego endpointa -> **GET api/song** -> pobiera wszystkie piosenki z bazy danych.
4. Dodanie nowego endpointa -> **GET api/album** -> pobiera wszystkie albumy z bazy danych.
5. Zmiana sortowania danych, jeśli nie wybierze się **ASC (1)**-> rosnąco lub **DESC (2)** malejąco oraz słowa szukanego **SearchWord**, to nie segreguje.
6. Dodanie nowego endpointa -> **GET api/artist/{artistId}/album/{albumId}/song/details/{songId}** -> aby mieć dostęp do danych szczegółowych danej piosenki o dodatkowe pola: **AlbumTitle, ArtistId, ArtistName**.
7. Dodanie nowego endpointa -> **GET api/artist/{artistId}/album/details/{albumId}** -> aby mieć dostęp do danych szczegółowych danego albumu o dodatkowe pola: **ArtistName, ArtistId**.
8. Dodanie nowego endpointa -> **GET api/artist/details/{Id}** -> aby mieć dostęp do danych szczegółowych danego artysty o dodatkowe pola: **ContactEmail, ContactNumber**.