MA/IXIFY Dokumentáció

SPECIFIKÁCIÓ, RENDSZERTERV, TELEPÍTÉSI DOKUMENTÁCIÓ

Bordács Gergő (KJB0VB) – Hegedüs Péter (LMZGU4)

Szoftverarchitektúrák | 2016.11.20.

Tartalomjegyzék

Specifikáció

FELADATKIÍRÁS

Az alkalmazás legyen képes több Spotify playlist (lejátszási lista) alapján összeállítani egy valószínűleg minden résztvevő számára elfogadható playlistet (pl. metszetképzéssel). Az így létrehozott playlist-et legyen lehetőség elmenteni az alkalmazást használó user Spotify fiókjába. A felhasználni kívánt playlisteket a felhasználók a playlist URL-je, vagy egy Spotify usernév segítségével a playlistek lekérése után tudják megadni. Fontos, hogy az alkalmazás egyszerű felületen, könnyen kezelhető legyen, több platformon. Bővebb leírás a Részletes feladatleírás részben található.

Az alkalmazás a MaiXIFY projekt-nevet kapta, így a továbbiakban az ilyen nevű hivatkozások alatt a házibuli playlist összeállító rendszert értjük.

A FEJLESZTŐI CSAPAT

A csapat tagjai:

Név	**Neptun**	**E-mail**
Bordács Gergő	KJB0VB	gerawba@hotmail.com
Hegedüs Péter	LMZGU4	hegedus21@gmail.com

A csapat kis létszámának következtében, valamint az egyenlőség biztosításának érdekében nem osztottunk ki külön szerepeket a csapaton belül.

RÉSZLETES FELADATLEÍRÁS

A feladat célja egy webalkalmazás létrehozása, ami lehetővé teszi Spotify lejátszási listák generálását előre megadott Spotify lejátszási listák felhasználásával.

Az alkalmazásnak a legtöbb népszerű webes böngészőben futnia kell, továbbá platform függetlennek kell lennie (egy asztali számítógépen ugyanúgy lehessen használni, mint egy mobil eszközön).

Az alkalmazás használatakor egy felhasználó a következő módon adhatja meg a lejátszási listát (listákat), amiket az alkalmazás figyelembe vesz a közös lejátszási lista generálásakor:

- közvetlenül megadja a lejátszási lista azonosítóját, URL-jét, VAGY
- megad egy Spotify felhasználónevet, majd a Spotify felhasználó lejátszási listái közül kiválasztja a releváns listákat.

Az alkalmazás legfeljebb 100 db lejátszási listát képes figyelembe venni a közös lejátszási lista elkészítésekor.

Az alkalmazás használatakor a következő beállításokat végezheti el a felhasználó:

- Küszöbérték (Threshold) beállítása:
  egy százalékos érték (0-100%), ha a megadott lejátszási listák közül egy zeneszám a listák beállított küszöbérték százaléknál nagyobb részében benne van, akkor a zeneszám bekerül a generált közös lejátszási listába, egyébként nem kerül be.
- Javasolt zenék engedélyezése:
  ha a megadott lejátszási listákban nem szerepel olyan zeneszám, ami közös lenne a lejátszási listákban (illetve a küszöbérték alapján sem kerülne be), akkor a közös lejátszási lista generálásakor az alkalmazás azt veszi figyelembe, hogy a különböző lejátszási listákon melyek a hasonló zeneszámok (stílus, előadó stb.).

A megadott lejátszási listák és a beállítások alapján az alkalmazás olyan lejátszási listát hoz létre, ahol a legtöbb olyan zeneszám szerepel, amit a fentiek alapján a legtöbb felhasználó szívesen hallgat.

A kapott lejátszási listában szereplő zeneszámokat különböző, előre beállított szempontok alapján lehet rendezni:

legtöbb találat/hasonló stílus/véletlenszerű.

A lista generálása után a felhasználónak lehetősége van a lejátszási lista elmentésére, adott néven a saját Spotify fiókjába. Ehhez azonosítania kell a felhasználónak magát a Spotify rendszerbe.

TECHNIKAI PARAMÉTEREK, ARCHITEKTÚRA

Az alkalmazás az ASP.NET platformon fog futni, ehhez egy megfelelő kiszolgáló szükséges: ilyen például egy dedikált IIS (Internet Information Services) webszerver, vagy a mai trendek szerinti webalkalmazás hosztolás a felhőben (pl. Microsoft Azure) - mi ez utóbbi fogjuk használni. Magát a szolgáltatást azonban bármilyen operációs rendszeren lehet majd használni (Windows/Linux/Mac OS X stb.), hiszen annak eléréséhez mindössze egy webböngésző (és természetesen internet kapcsolat) szükséges.

A szoftver klasszikus kliens-szerver architektúrájú lesz, melyben a vékony kliens (azaz a böngésző) HTTP kérésekkel fordul a szerverhez, mely megvalósítja az alkalmazás üzleti logikáját, és a Spotify Web API-ját felhasználva válaszol a kérésre. Az alkalmazás a 2016 nyarán megjelent ASP.NET Core web frameworkre fog épülni, mely az ASP.NET MVC, WebAPI és Web Pages technológiáit “ötvözi”.

Az alkalmazásnak két jól elkülöníthető és különböző felelősségű rétege lesz:

- az **üzleti logika** (vezérlési réteg) C# nyelven kerül implementálásra, és a fent említetteknek megfelelően a Spotify Web API-ját fogja felhasználni, mely segítségével képes lesz playlistekhez, zeneszámokhoz (és az ezek információhoz) hozzáférni

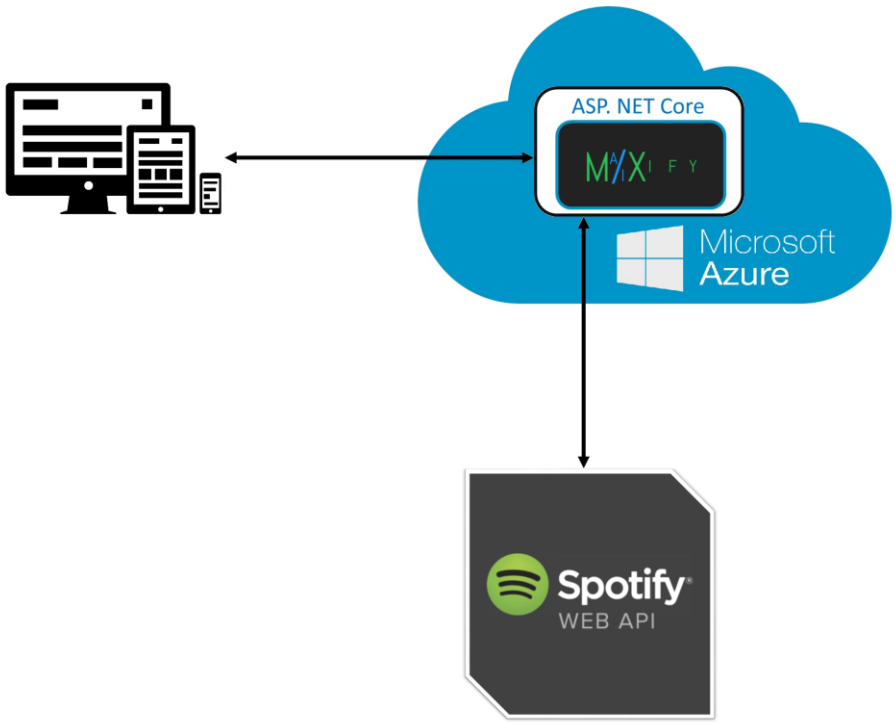
- a **megjelenítés** az alkalmazás webes mivoltából fakadóan egy webböngészőben történik: a felhasználói felület kialakítása során reszponzivitásra törekszünk, így a Bootstrap front-end keretrendszert is felhasználjuk.

Ezen felül, a kliens oldalon szükség szerint JavaScript-et (és/vagy különböző JS könyvtárakat, pl. JQuery) is alkalmazunk az interaktív/reszponzív működés eléréséhez.

Mivel a felhasználók adatait, playlistjeit stb.-t a Spotify API-ján keresztül érjük el, így az alkalmazás megvalósításához nincsen szükség adatbázisra. Az alkalmazás beállításait cookiek (HTTP sütik) segítségével fogjuk a böngészőben tárolni és onnan betölteni, illetve a szervernek elküldeni.

Hasonló okok miatt nem szükséges kitüntetett figyelmet fordítani a biztonsági kérdésekre sem: a felhasználók bejelentkezési adatait stb.-t is a Spotify API-ja kezeli, így az ilyen és ehhez hasonló érzékeny adatokkal az alkalmazásunk nem kerül közvetlen kapcsolatba.

Az alkalmazás architektúrája:



Rendszerterv

Az alábbiakban a MaiXIFY webalkalmazás architektúrája, komponensei, működése kerül részletezésre. Először a teljes alkalmazást vizsgáljuk figyelembe véve az alkalmazott technológiákat, architektúrális kérdéseket, mintákat, majd az egyes komponensek kerülnek ismertetésre.

ÁTTEKINTÉS

Az alkalmazást ASP.NET platformon fejlesztettük, ASP.NET Core web framework segítségével. A keretrendszer fontos tulajdonsága, hogy segítségével MVC szerkezeti mintán alapuló alkalmazásokat lehet fejleszteni, a MaiXIFY webalkalmazás is ezen elv mentén működik.

A specifikációban leírt funkcionalitások megvalósításához igénybe kell venni külső szolgáltatásokat. Spotify adatok lekéréséhez igénybe kell venni a Spotify webes szolgáltatását, a Spotify Web API-t. A szerveren futó MaiXIFY webalkalmazás ezért különböző adatokat cserél a Spotify Web API-n keresztül a Spotify szerverrel. A kommunikáció a Spotify szerverrel REST alapon működik JSON formátumú üzenetek segítségével. Mivel több különböző Spotify Web API végponttal történik a kommunikáció, célszerű a funkcionalitásoknak megfelelően az API-val történő kommunikációt struktúrálni az MVC alkalmazáson belül felhasználva **Wrapper Facade** architektúrális mintát. Ez az architektúrális minta a **Szolgáltatás hozzáférés és konfiguráció** architektúrális minták csoportjába tartozik, így segítségével a Spotify szolgáltatáshoz való hozzáférést lehet hordozható és karbantartható objektumorientált osztályba szervezni.

Spotify Web API Wrapper

A Spotify Web API felhasználását két csoportba lehet sorolni:

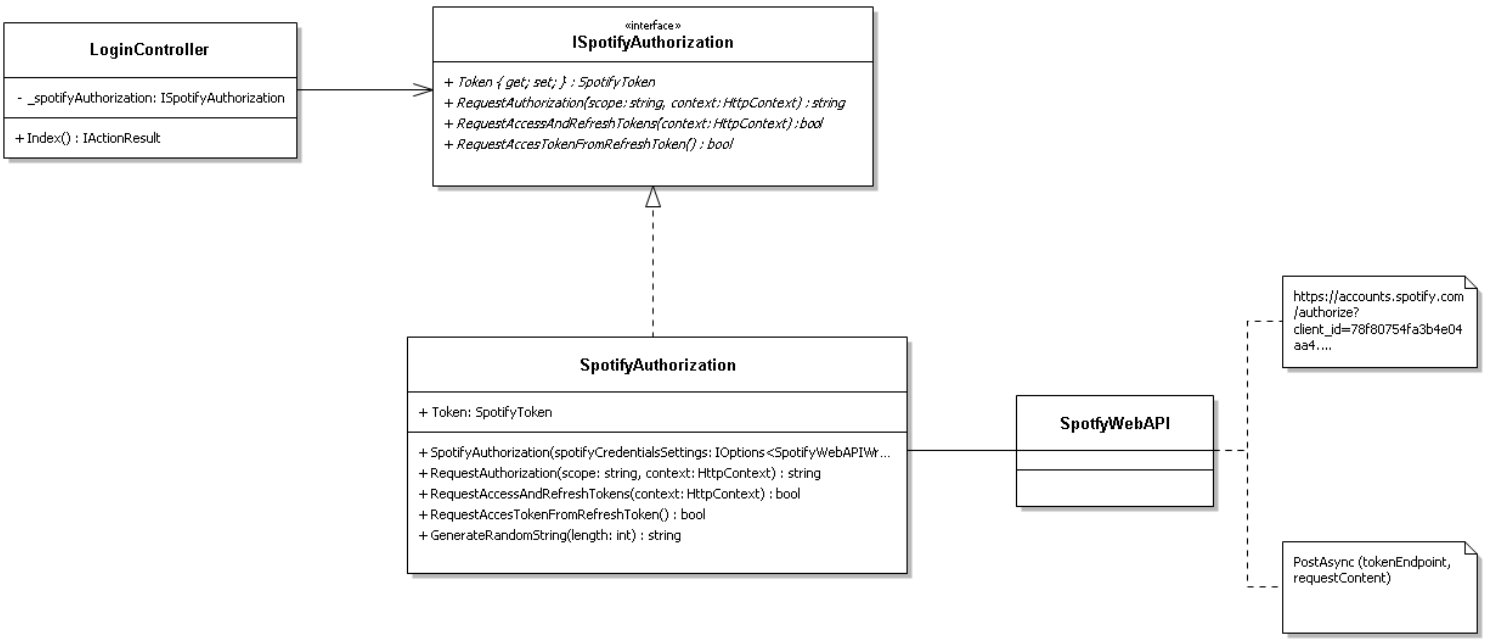
- Spotify-hoz történő csatlakozás, azonosítás
- Spotify adatbázisból adatok lekérdezése

Ehhez a két csoporthoz érdemes két külön csomagoló osztályt létrehozni. A hordozhatóságot és könnyen módosíthatóságot szem előtt tartva érdemes az egyes csoportokba tartozó függvényekhez interfészeket létrehozni és ezeket az interfészeket burkoló osztályokban megvalósítani.

Létrehozunk tehát két interfészt a megfelelő csoportoknak: *ISpotifyAuthorization* Spotify-hoz történő csatlakozáshoz és azonosításhoz a szerver felé, *ISpotifyEndpointAccessor* Spotify adatbázisból történő adatok lekérdezéséhez.

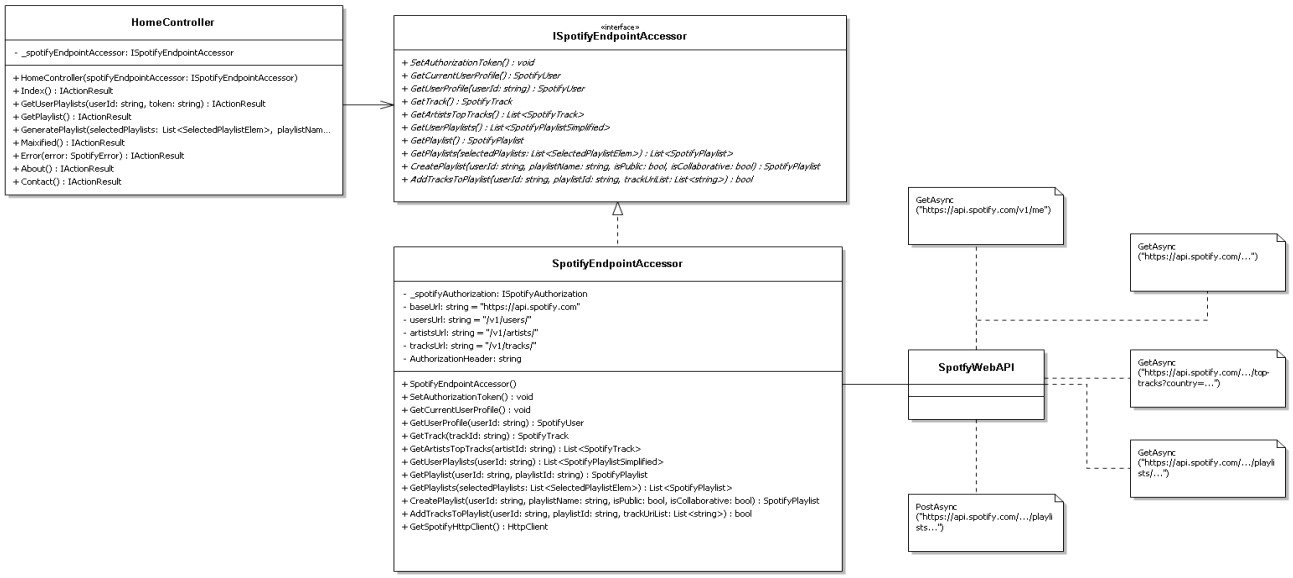
Az ezeket az interfészeket megvalósító burkoló osztályok: *SpotifyAuthorization, SpotifyEndpointAccessor.*

A **Wrapper Facade** architektúrális minta megvalósításához tartozó osztálydiagram a Spotify-hoz történő csatlakozás, azonosítás csoporthoz a következőképpen néz ki:



1. ábra: Wrapper Facade – SpotifyAuthorization

A **Wrapper Facade** architektúrális minta megvalósításához tartozó osztálydiagram a Spotify adatbázisból adatok lekérdezése csoporthoz a következőképpen néz ki:



2. ábra: Wrapper Facade – SpotifyEndpointAccessor

Lejátszási listák generálása

Az alkalmazás legfőbb feladata az, hogy a felhasználó által kiválasztott lejátszási listák alapján egy olyan lejátszási listát hozzon létre, ami a kiválasztott lejátszási listák metszetét tartalmazza, vagy ha ez a metszet üres vagy kevés számot tartalmaz, akkor olyan zenékből álljon, amiket minden felhasználó szívesen hallgatna.

Ennek a funkcionalitásnak a megvalósításához érdemes egy külön osztályt létrehozni, aminek metódusai megvalósítják ezt a feladatot.

Az implementáció során létrehozott osztály neve *PlaylistMixerCoreLogic*, ami a következő elemeket tartalmazza:

- *ISpotifyEndpointAccessor* típusú *_spotifyEndpointAccessor* privát tagváltozó, a Spotify Web API végpontokhoz való hozzáféréshez
- *PlaylistMixerSettings* típusú *Settings* tulajdonságot, ami a beágyazott *Settings* osztály egy példánya, ami az algoritmushoz használt beállításokat tartalmazza:
    - *Threshold:* double típusú 0 és 1 közötti érték
    - *RecommendedMusic:* bool típusú tagváltozó, amivel jelezni lehet ajánlott zenék hozzáadását a lejátszási listához a generálásnál
    - *SortOption:* felsorolt típust, amivel a generált lejátszási listát lehet rendezni *MostHit* (legtöbbször fordult elő)/*Popularity* (népszerűség)/*Random* (véletlenszerű) tulajdonságok alapján
- *PlaylistMixerCoreLogic* konstruktort, amivel végpontot lehet beállítani és az alapbeállításokat létrehozni
- *GenerateMaixifyPlaylist* metódust, ami a megadott lejátszási listák és beállítások alapján lejátszási listát hoz létre
- *GenerateRecommendedPlaylist* metódus, ami ajánlott zenékből álló listát hoz létre

A *GenerateMaixifyPlaylist* metódus által implementált algoritmus:

1. Ha túl kevés lejátszási listát adtak meg vagy a *Threshold* beállítása nem megfelelő -> *return null*;
2. Eltároljuk a lejátszási listák számát (*playlistNumber*) és két segéd Dictionary-t hozunk létre (*tracksFrequency* – string kulcs és *TrackInfo* értékek; *artistsFrequency* – string kulcs és int értékek)
3. Végighaladunk a kiválasztott listák összes zeneszámán és a *tracksFrequency*-ben eltároljuk a zene azonosítóját (ha még nem tartalmazza), a zene népszerűségét leíró adatot és azt, hogy hányszor fordult elő a kiválasztott lejátszási listákban a zeneszám. Ha a *RecommendedMusic* beállítást is aktiválta a felhasználó, akkor az *artistsFrequency*-ben eltároljuk az előadó azonosítóját és a feldolgozás során folyamatosan növeljük, hogy hány zeneszámot ad elő a kiválasztott lejátszási listákban található számok közül.

4. Egy listában a megadott rendezési feltételnek megfelelően rendezzük a *tracksFrequency*-ben található elemeket,
5. Létrehozunk egy listát ajánlott zenékhez és az előbb létrehozott rendezett listán végigmegyünk: ha egy szám legalább kétszer előfordul és legalább annyiszor mint a lejátszási listák számának és a *Threshold* beállítás szorzata, akkor a zeneszám bekerül az ajánlott zenék listájába.
6. Ha az előző lépés után az ajánlott zenék listája üres vagy 10-nél kevesebb szám van és a *RecommendedMusic* beállítás aktiválva van, akkor a következő lépések következnek:
    a. Ha a *RecommendedMusic* nincs aktiválva -> *return null*
    b. Meghívjuk az ajánlott zenékből lejátszási listát generáló függvényt, majd az így kapott számokat hozzáadjuk az ajánlott zenék listájához az esetlegesen többször előforduló számokat kiszűrve.
7. Az aktuálisan bejelentkezett felhasználó Spotify fiókjába létrehozunk egy lejátszási listát a megadott néven és beállításokkal (publikus-e, kollaboratív-e) az ajánlott zenék listájából.

A *GenerateRecommendedPlaylist* metódus által implementált algoritmus:

1. Az algoritmus bemenete az a rendezett lista, amit az előző algoritmus hoz létre a 4. pontban (*tracksFrequency*, ahol a zenék azonosítója szerepel a zenék népszerűségével és előfordulásainak számával) és *artistsFrequency*, ahol az előadókat tároljuk és azt, hogy a kiválasztott zenék közül az egyes előadók hány számot adnak elő.
2. Ha bármelyik bemenő paraméter null -> *return null*
3. Létrehozunk egy listát az ajánlott zenéknek, majd a paraméterként kapott zenéket tartalmazó listát szűrjük, hogy csak különböző zenék szerepeljenek benne és ezt rendezzük a zenék népszerűsége alapján.
4. Végigmegyünk ezen a listán, és amelyik számnak a népszerűsége nagyobb egy meghatározott értéknél (77), azt hozzáadjuk az ajánlott zenékhez.
5. Egy HashSet-ben eltároljuk az előadókra vonatkozó adatokat
6. A *topArtistsFrequencyLimit* változóban eltároljuk a legnépszerűbb előadókat meghatározó mutatót a következőképpen:
    a. Ha több mint két szám szerepel tőle, akkor az artistsFrequency listát az előadó előfordulásainak száma alapján rendezzük csökkenő sorrendbe és kiválasztjuk a harmadik legmagasabb előfordulási mutatót.
    b. Ha pontosan két szám szerepel tőle, akkor először szintén rendezzük az artistsFrequency listát az előadó előfordulásainak száma alapján, majd kiválasztjuk a második legmagasabb előfordulási mutatót.
    c. Ha csak olyan előadók vannak, akiknek egy-egy száma fordul elő a listákban, akkor az *topArtistsFrequencyLimit* értékét egyre állítjuk.

7. Ha a *topArtistsFrequencyLimit* értéke kevesebb, mint kettő, akkor visszatérünk az ajánlott zenék listájával.
8. Egyébként az *artistsFrequency* és a *topArtistsFrequencyLimit* segítségével kiválasztjuk a legnépszerűbb előadókat.
9. A Spotify adatbázisból lekérdezzük ezeknek az előadóknak a három legnépszerűbb zeneszámát és ezeket a számokat hozzáadjuk az ajánlott zenék listájához.

BACKEND

A M^A/_IXIFY webalkalmazás architektúráját a különböző komponensek alapján két részre osztottuk: az MVC architektúrális mintának megfelelő modell és vezérlő komponensek alkotják az alkalmazás backend-jét, míg a megjelenítésért felelős nézetek az alkalmazás frontend-jét. A Spotify Web API-t használó csomagoló osztályok szintén a BACKEND részei. Ezekben a csomagoló osztályokban található metódusokat egyrészt a vezérlők metódusai, másrészt a lejátszási listák generálását végző osztály metódusai használják.

Spotify Web API

A Spotify Web API segítségével az alkalmazásunk képes lesz arra, hogy a Spotify zenei adatbázisból adatokat kérjen le, valamint testre szabja egy-egy felhasználó elmentett zenéit, lejátszási listáit.

A REST alapokon működő Web API a végpontokról JSON formátumban szolgáltat adatokat a Spotify adatbázisból például előadókról, albumokról, lejátszási listákról és zeneszámokról.

Az API ezen a címen érhető el: https://api.spotify.com. Az API több végpontot tartalmaz, melyek közül vannak szabadon elérhető nyitott végpontok, de a privát adatok (felhasználói fiók adatok, lejátszási listák…stb.) eléréséhez azonosításra van szükség, amit az *Authorization Code* folyamat valósít meg.

Alkalmazás regisztrálása a Spotify-nál

Amennyiben alkalmazásunk privát adatokkal is dolgozik (felhasználói adatok, lejátszási listák) az alkalmazást először regisztrálni kell a Spotify rendszerben. Ehhez egy létező Spotify fiók szükséges.

Az alkalmazás regisztrációjánál meg kell adni az alkalmazás nevét (M^A/_IXIFY), leírást az alkalmazásról, valamint Redirect URI-kat, amiket a Spotify rendszer fog meghívni, amikor az azonosítási folyamat befejeződik.

A regisztráció után a Spotify rendszer generál számunkra egy ügyfél azonosítót és egy titkos kulcsot, amiket az azonosítási folyamathoz kell felhasználni, valamint abban az esetben, ha biztonságos hívásokkal vesszük igénybe a Spotify Web API-t.

Spotify URI-k és azonosítók

A Spotify Web API hívásoknál, és az onnan érkező válaszoknál jellegzetesen a következő paraméterek használatasok:

PARAMETER	DESCRIPTION	EXAMPLE
Spotify URI	The resource identifier that you can enter, for example, in the Spotify Desktop client's search box to locate an artist, album, or track. To find a Spotify URI simply right-click (on Windows) or Ctrl-Click (on a Mac) on the artist's or album's or track's name.	`spotify:track:6rqhFgbbKwnb9MLmUQDhG6`
Spotify ID	The base-62 identifier that you can find at the end of the Spotify URI (see above) for an artist, track, album, playlist, etc. Unlike a Spotify URI, a Spotify ID does not clearly identify the type of resource; that information is provided elsewhere in the call.	`6rqhFgbbKwnb9MLmUQDhG6`
Spotify category ID	The unique string identifying the Spotify category.	`party`
Spotify user ID	The unique string identifying the Spotify user that you can find at the end of the Spotify URI for the user. The ID of the current user can be obtained via the Web API endpoint https://api.spotify.com/v1/me.	`wizzler`
Spotify URL	An HTML link that opens a track, album, app, playlist or other Spotify resource in a Spotify client (which client is determined by the user's device and account settings at play.spotify.com).	`http://open.spotify.com/track/6rqhFgbbKwnb9MLmUQDhG6`

1. táblázat: Spotify URI-k és ID-k

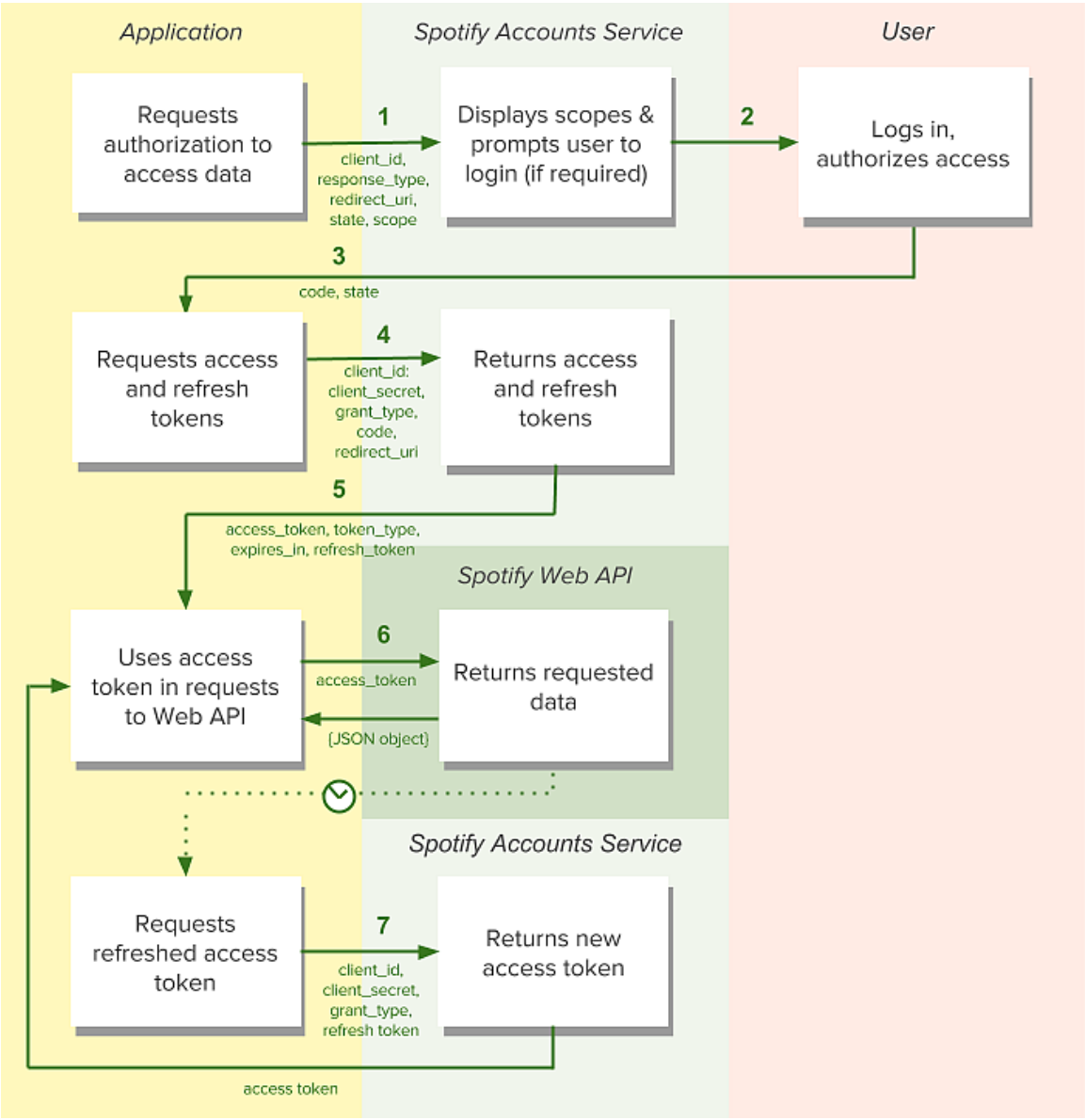
Authorization Code azonosítási folyamat

A folyamat először egy kódot kap a Spotify rendszertől, majd ezt a kódot hozzáférési és frissítési tokenekre cseréli. Mivel ehhez a cseréhez szükség van a titkos kulcsra, ezt a lekérést szerver oldalon kell megvalósítani, hogy a kulcs integritása ne sérüljön. A folyamat előnye, hogy a frissítési token felhasználásával meg lehet hosszabbítani a hozzáférési token érvényességét.

Az Authorization Code azonosítási folyamat a következő tulajdonságokkal rendelkezik:

- felhasználói adatokhoz történő hozzáférés engedélyezése
- kérések kiszolgálásának priorizálása
- hozzáférési token meghosszabbíthatósága

Az Authorization Code folyamat lépései a következő ábrán látszanak:



3. ábra: Authorization Code Flow

Felhasznált Spotify Web API végpontok

A M^A/_IXIFY webalkalmazás az alábbi végpontokat használja fel működése során:

- https://accounts.spotify.com/authorize:
  A *LoginController Index* metódusa erre a címre irányítja át a felhasználót. Ezt a címet a *SpotifyAuthorization* osztály *RequestAuthorization* metódusa állítja össze. Az URL-ben meg kell adni a kliens azonosítót, válasz típusát (code), redirect uri-t, scope-ot (user-read-private playlist-read-private playlist-modify-public playlist-modify-private), state-et.
  Példa:
  https://accounts.spotify.com/authorize/?client_id=5fe01282e44241328a84e7c5cc169165&response_type=code&redirect_uri=https%3A%2F%2Fexample.com%2Fcallback&scope=user-read-private%20user-read-email&state=34fFs29kd09

- https://accounts.spotify.com/api/token:
  A *SpotifyAuthorization* osztály *RequestAccessAndRefreshTokens* metódusából érjük el ezt a végpontot. A POST kérés törzsében a következő paramétereket kell tartalmaznia a hívásnak: grant_type (“authorization_code” az értéke), code (a *RequestAuthorization* metódus meghívása során kapott kód értéke), redirect_uri (csak validáláshoz használt callback függvény URI-ja, meg kell egyeznie azzal a redirect_uri-val, amit előzőleg az authorize/ végpontnak adtunk meg).
  A fejlécben át kell adni Base64 kódolásban a kliens azonosítót és a titkos kulcsot.
  Sikeres kérés esetén a következőhöz hasonló válaszüzenet érkezik a végponttól:

```
{
   "access_token": "NgCXRK...MzYjw",
   "token_type": "Bearer",
   "scope": "user-read-private user-read-email",
   "expires_in": 3600,
   "refresh_token": "NgAagA...Um_SHo"
}
```

- A *SpotifyAuthorization* osztály *RequestAccesTokenFromRefreshToken* metódusa egy hasonló kérést küld a végpontnak. Ebben az esetben a *grant_type* parameter lrtéke “*refresh_token*”, valamint a többi parameter helyett egy *refresh_token* paramétert kell átadni, aminek az értéke az előbbi lekérés során kapott *refresh_token*. A fejlécben hasonlóan kell elküldeni a kliens kódot és a titkos kulcsot, mint az előző pontban.
  Példa sikeres kérés esetén a válaszra:

```
{
   "access_token": "NgA6ZcYI...ixn8bUQ",
   "token_type": "Bearer",
   "scope": "user-read-private user-read-email",
   "expires_in": 3600
}
```

- https://api.spotify.com/v1/me:
  A *SpotifyEndpointAccessor* osztály *GetCurrentUserProfile* metódusa használja ezt a végpontot. Az éppen bejelentkezett felhasználóhoz tartozó adatokat lehet lekérdezni vele. A *HomeController Index* metódusa használja például ahhoz, hogy a kezdőoldalon az üdvözlő üzenetben megszólítsa az éppen bejelentkezett felhasználót. Továbbá a fent ismertetett algoritmus is használja ezt a metódust, amikor egy-egy előadó legnépszerűbb számait kérdezi le a felhasználó országából. Végül, amikor a generált lejátszási listát létrehozzuk, szükség van erre a lekérdezésre ahhoz, hogy a megfelelő felhasználó fiókjába mentsük el a generált lejátszási listát.

- https://api.spotify.com/v1/users/{userId}:
  Egy paraméterben megadott felhasználó adatai kérhetőek le a végponttól. A *SpotifyEndpointAccessor* osztály *GetUserProfile (string userId)* metódusa hajtja végre a kérést az API felé. A *HomeController GetUserPlaylists (string userId, string token)* metódusa használja fel a felhasználó által megadott Spotify azonosítóhoz tartozó fiókhoz tartozó adatok lekérdezéséhez.

- https://api.spotify.com/v1/tracks/{trackId}:
  Egy paraméterben megadott zeneszám adatai kérdezhetőek le a végponttól. A *SpotifyEndpointAccessor* osztály *SpotifyTrack GetTrack (string trackId)* metódusa hajtja végre a kérést az API felé.

- https://api.spotify.com/v1/artists/{artistId}/top-tracks?country=...:
  Egy paraméterben megadott előadó legnépszerűbb zenéi kérdezhetőek le a végponttól. A *SpotifyEndpointAccessor* osztály *GetArtistsTopTracks (string artistId)* metódusa hajtja végre a kérést az API felé. Ezt a lejátszási listát generáló algoritmus használja fel ajánlott zenék kiválasztásához.

- https://api.spotify.com/v1/users/{userId}/playlists:
  Egy paraméterben megadott felhasználó lejátszási listái kérdezhetőek le a végponttól. A *SpotifyEndpointAccessor* osztály *GetUserPlaylists (string userId)* metódusa hajtja végre a kérést. A *HomeController GetUsersPlaylists (string userId)* metódusa használja ezt a lekérést ahhoz, hogy a MA/IXIFY felhasználója által megadott Spotify felhasználó lejátszási listáit lekérdezze a végponttól.

- https://api.spotify.com/v1/users/{userId}/playlists/{playlistId}:
  Egy konkrét felhasználóhoz tartozó konkrét lejátszási lista adatai kérdezhetőek le a segítségével. A felhasználó és a lejátszási lista azonosítója paraméterként adható meg. A *HomeController GetPlaylist (string userId, string playlistId, string token)*

metódusa használja ezt, amikor az alkalmazás felhasználója egy konkrét lejátszási listát ad meg URL-ként. Továbbá ugyanezen controller *Maixified (string userId, string playlistId, string token)* metódusa is felhasználja ezt a lekérést, amikor a generált lejátszási listát kérdezi le a szervertől és jeleníti meg a weblapon a felhasználónak.

- POST https://api.spotify.com/v1/users/{user_id}/playlists:
  A lejátszási listát generáló algoritmus használja fel ezt a végpontot akkor, amikor egy lejátszási listát hoz létre egy paraméterben megadott felhasználó fiókjában. A kérés törzsében kerülnek átadásra a lejátszási listára vonatkozó beállítások (a lejátszási lista neve, publikus-e, kollaboratív-e).

- POST https://api.spotify.com/v1/users/{user_id}/playlists/{playlist_id}/tracks:
  A lejátszási listát generáló algoritmus használja fel ezt a végpontot akkor, amikor az előzőleg létrehozott lejátszási listához zeneszámokat ad hozzá. A felhasználó, akinek a fiókjába mentettük a lejátszási listát és a lejátszási lista azonosítója paraméterként kerül átadásra. Query stringben egy listában kell átadni a zeneszámok Spotify Track azonosítóját vesszővel elválasztva őket egymástól. (pl.: `uris=spotify:track:4iV5W9uYEdYUVa79Axb7Rh,spotify:track:1301WleyT98MSxVHPZCA6M`)
  Sikeres lekérés esetén a következőhöz hasonló üzenetet kell kapjunk a szervertől: `HTTP/1.1 201 Created (valamint egy "snapshot_id-t" tartalmazó JSON objektumot)`

Modelek

Az alkalmazás jellegéből adódóan többfajta adatot kezel, például Spotify felhasználókat, lejátszási listákat, zeneszámokat stb. Ezek mind összetett adatok, amik több mezőből állnak és különböző típusú információkat tartalmaznak.

A végpontokkal történő kommunikáció során JSON formátumban történik az adatok cseréje, így célszerű olyan modelleket létrehozni a fent felsorolt adatokhoz, amiket könnyű JSON formába (és vissza modellekké JSON formátumból) alakítani. A különböző Spotify-os entitásokhoz a következő modelleket célszerű létrehozni:

- **SpotifyArtist:**
  - *Id : string – előadó azonosítója*
  - *Name : string – előadó neve*
  - *Popularity : int – előadó népszerűségét jelző érték*
  - *Genres : List<string> - műfajokat tartalmazó lista*
  - *Uri : string – előadó URI-ja*

- **SpotifyError:**
  - *ErrorCode : int – hibakód*
  - *ErrorMessage : string – hibaüzenet*

- **SpotifyExternalUrl:**
  - *Url : string – külső Spotify-os URL cím*

- **SpotifyPaging:**
  - *TotalItemNumbers : int – összes elem száma*
  - *Items : List<T> - generikusan megadott elemek listája*

- **SpotifyPlaylist:**
  - *Id : string – lejátszási lista azonosítója*
  - *Name : string – lejátszási lista neve*
  - *IsPublic : bool – lejátszási lista publikus tulajdonságát tároló bit*
  - *OwnerUser : SpotifyUser – lejátszási lista Spotify felhasználó tulajdonosa*
  - *Tracks : SpotifyPaging<SpotifyPlaylistTracks> - lejátszási lista zeneszámai*
  - *SpotifyExternalUrl : SpotifyExternalUrl – külső Spotify-os URL cím*
  - *Uri : string – lejátszási lista URI-ja*

- **SpotifyPlaylistSimplified:**
  - *Id : string – lejátszási lista azonosítója*
  - *Name : string – lejátszási lista neve*
  - *IsPublic : bool – lejátszási lista publikus tulajdonságát tároló bit*
  - *OwnerUser : SpotifyUser – lejátszási lista Spotify felhasználó tulajdonosa*
  - *Tracks : SimplifiedTrack - lejátszási lista zeneszámai*
  - *SpotifyExternalUrl : SpotifyExternalUrl – külső Spotify-os URL cím*

- **SimplifiedTrack:**
  - *Href : string – hivatkozás a zeneszámra*
  - *TotalItemNumbers : int – hány elem szerepel*

- **SpotifyPlaylistTrack:**
  - *Track : SpotifyTrack – Spotify-os zeneszám*
  - *IsLocal : bool – lokális tulajdonságot tároló bit*

- **SpotifyTrack:**
  - *Id : string – Spotify zeneszám azonosítója*
  - *Name : string – Spotify zeneszám neve*
  - *Artists : List<SpotifyArtist> - zeneszámot előadók listája*

- *Popularity : int – zeneszám népszerűségét tükröző mérőszám*
- *Uri : string – lejátszottZeneszámURI-ja*

- **SpotifyTracksList:**
  - *Tracks : List<SpotifyTrack> - Spotify zeneszámokat tartalmazó lista*

- **SpotifyUser:**
  - *Id : string – Spotify felhasználó azonosítója*
  - *DisplayName : string – Spotify felhasználó által megjelenítendő név*
  - *Country : string – Spotify felhasználó országa*

Két további osztályt érdemes bevezetni:

- **SpotifyCredentialsSettings**

  Ebben az osztályban érdemes tárolni az alkalmazás regisztrációja során kapott kliens azonosítót, titkos kulcsot és a regisztráció során megadott callback függvény URI-ját, amiket egy külső json fájlban (appsettings.json) tárolunk permanensen.
  - *ClientId : string – alkalmazás azonosítója*
  - *ClientSecret : string – alkalmazás regisztrációja során kapott titkos kulcs*
  - *RedirectURI – string : callback függvény*

- **SpotifyHelpers**

  Egy segédosztály, ami az alkalmazás működéséhez szükséges adatokat tartalmaz, valamint segédfüggvényeket és beágyazott segédosztályokat.

  Legfontosabb itt tárolt adatok:

  Hozzáférési token cookie-jának kulcsa, refresh token cookie-jának kulcsa, threshold beállítás cookie-jának kulcsa, ajánlott zene beállítás cookie-jának kulcsa, rendezési szempont beállítás cookie-jának kulcsa, állapot cookie kulcsa, alapértelmezett generált lejátszási lista nevének előtagja.

  Legfontosabb segédfüggvények:

  Megadott playlist URI alapján Spotify widgetet készítő függvény, lejátszási listához nevet készítő függvény.

  Legfontosabb segédosztályok:

  TrackInfo *segédosztály, a generáló algoritmusnál használt, egy zenére vonatkozó mérőszámok tárolásához,* ***RequestContentCreatePlaylist*** *segédosztály a létrehozandó lejátszási listára vonatkozó adatok tárolásához,* ***SelectedPlaylistElem*** *a kiválasztott lejátszási listához tartozó adatok tárolásához,* ***RequestContentAddTrackToPlaylist*** *a lejátszási listához hozzáadandó zeneszámok listája.*

Kontrollerek

A M^A/_IXIFY webalkalmazás két vezérlőt tartalmaz: ***LoginController***-t a Spotify rendszer felé történő azonosításhoz, belépéshez, ***HomeController***-t a különböző felhasználói műveletek csoportosításához.

LoginController

Ez a vezérlő egy tagváltozót tartalmaz (*_spotifyAuthorization*), aminek a segítségével az alkalmazás felhasználója azonosítani tudja magát a Spotify rendszer felé.

Az *Index* metódus a *"…/Login/Index*" oldal lekérése esetén hívódik meg és a *spotifyAuthorization* tagváltozó segítségével átirányítja a felhasználót arra a helyre, ahol azonosítani tudja magát.

HomeController

A Home vezérlő egy *ISpotifyEndpointAccessor* típusú *_spotifyEndpointAccessor* nevű tagváltozót tartalmaz, aminek a segítségével az alkalmazás használatba tudja venni a Spotify Web API különböző végpontjait.

Home/Index

A *Home* vezérlő Index oldalának felkeresésekor az URL-ben meg kell adni a Spotify tokent, ahhoz, hogy minden felhasználó a saját munkafolyamatának megfelelő adatokat lássa a weboldalon és a saját adataival dolgozzon. Ha a megadott token nem null, akkor a válasz cookie-jai közé hozzáadjuk a tokennel kapcsolatos adatokat. Egyébként más esetben a cookie-k tartalmazzák a tokenre vonatkozó információkat és ekkor a tokenhez ezeket az adatokat rendeljük. A folyamat végén a metódus visszaadja válaszként a hozzá tartozó oldalt.

Home/GetUserPlaylists

A vezérlő *GetUserPlaylists* oldalának lekérésekor az URL-ben meg kell adni a tokent és a felhasználó azonosítóját. Ha bármelyik hiányzik, akkor a metódus hibaüzenettel tér vissza. Egyébként a tokent és a felhasználói azonosítót felhasználva először a felhasználóhoz tartozó adatokat kérdezzük le a Spotify adatbázisból majd, ha valóban létezik ilyen felhasználó, akkor lekérdezzük a lejátszási listáit és JSON formátumban visszaadjuk ezeket.

Home/GetPlaylist

A *Home* vezérlő *GetPlaylist* oldalának lekérésekor az URL-ben meg kell adni paraméterként a felhasználó azonosítóját, a lejátszási lista azonosítóját és a tokent. Ha ezek közül valamelyik null értékű, akkor hibaüzenettel válaszol a metódus. Egyébként a

paraméterek felhasználásával a Spotify adatbázisból lekérdezzük a megadott lejátszási listát, és ha valóban létezik ilyen, akkor JSON formátumban visszaadjuk.

Home/GeneratePlaylist

Az oldal lekérésekor az URL-ben meg kell adni a tokent, a felhasználó által korábban kiválasztott lejátszási listákat, a lejátszási lista nevét és a lejátszási listára vonatkozó beállításokat (publikus-e, kollaboratív-e). Amennyiben a token null értékű, vagy a felhasználó kevesebb, mint két lejátszási listát adott meg, akkor hibaüzenettel válaszolunk. Egyébként a paraméterekben megadott token és beállítások, valamint a cookie-kban tárolt generálási algoritmusra vonatkozó beállítások segítségével lejátszási listát generálunk, ezt elmentjük a bejelentkezett felhasználó Spotify fiókjába és átirányítjuk a felhasználót a *Maixified* oldalra, ahol a felhasználó meg tudja nézni az imént generált lejátszási listát.

Home/Maixified

Az oldal lekérésekor meg kell adni a tokent, a felhasználói azonosítót és lejátszási lista azonosítóját. A paraméterek felhasználásával lekérjük a lejátszási listát és megjelenítjük az ehhez tartozó nézetet.

Home/Error

Az oldal a paraméterként megadott hibát dolgozza fel és jeleníti meg egy weboldalon.

Home/About

Az alkalmazásra vonatkozó információkat tartalmaz. Bővíthető a jövőre nézve.

Home/Contact

A fejlesztők elérhetőségét jeleníti meg egy nézetben.

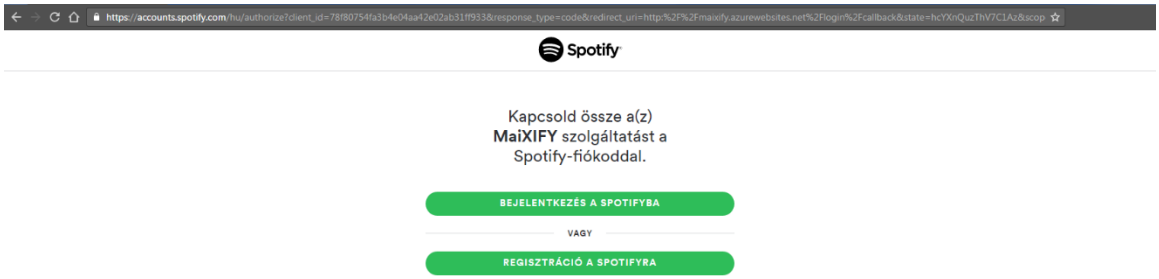
FRONTEND

A webalkalmazás frontend-je tartalmazza a különböző nézeteket (weboldalakat). A weboldalak JavaScript-et használnak kliens oldali logika megvalósításához. Az általános JavaScript nyelvi elemeken kívül a jQuery JavaScript könyvtárat is használják a weboldalak.

A weboldalak design-ját Spotify stílusú CSS segítségével alakítottuk ki.

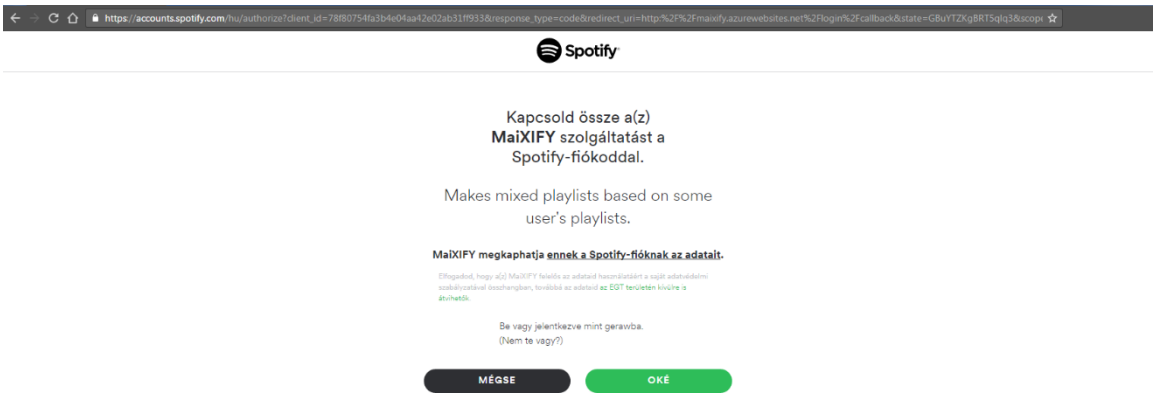
Index.cshtml

Azonosított felhasználó esetén ez az alkalmazás kezdőoldala. Amennyiben még nem azonosította magát a felhasználó először a bejelentkezéshez szükséges oldal jelenik meg:



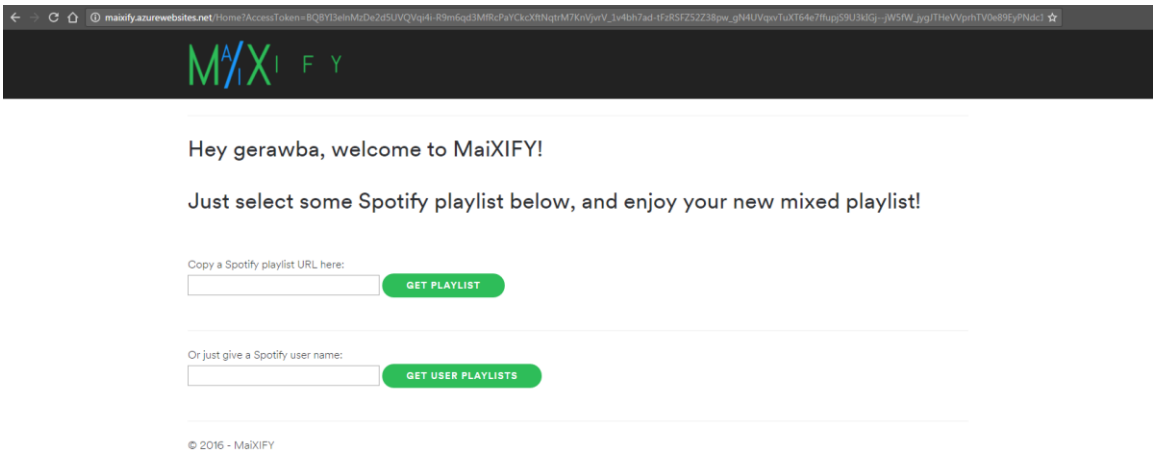
4. ábra: Bejelentkezés a Spotify-ba

Ha még nem használta a felhasználó a MaiXIFY alkalmazást korábban, akkor erre engedélyt kell adni, amihez egy külön oldal jelenik meg:



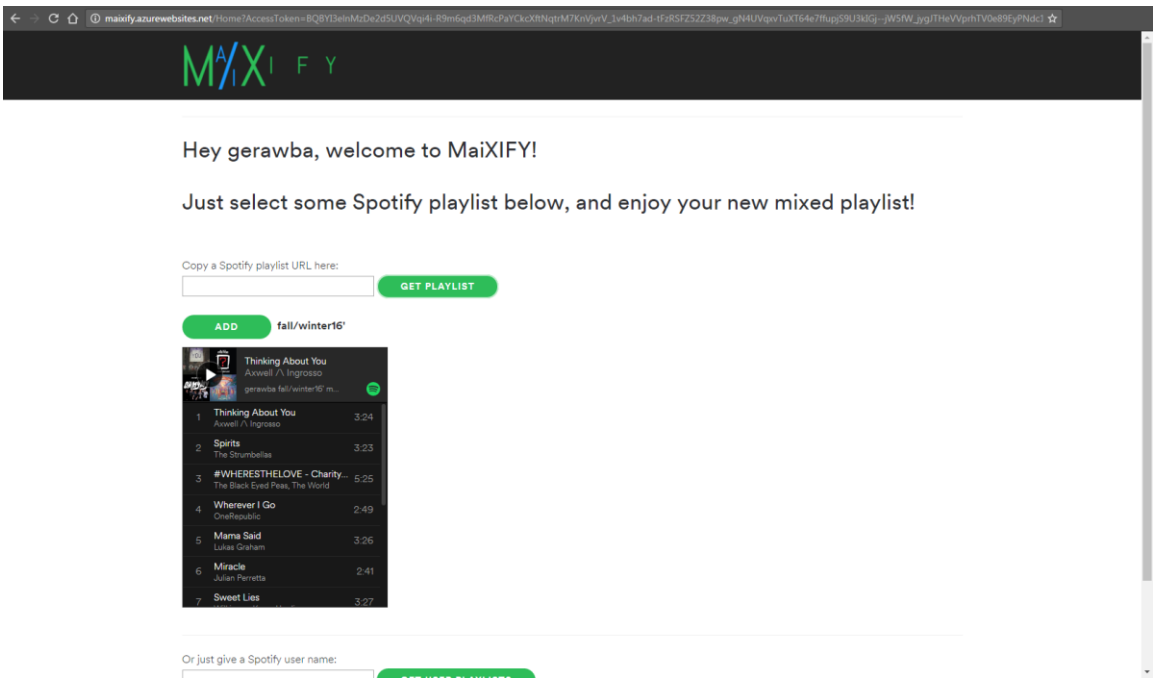
5. ábra: MaiXIFY alkalmazás engedélyezése

Az alkalmazás *Index.cshtml* kezdőoldala ezek után a következőképpen néz ki:

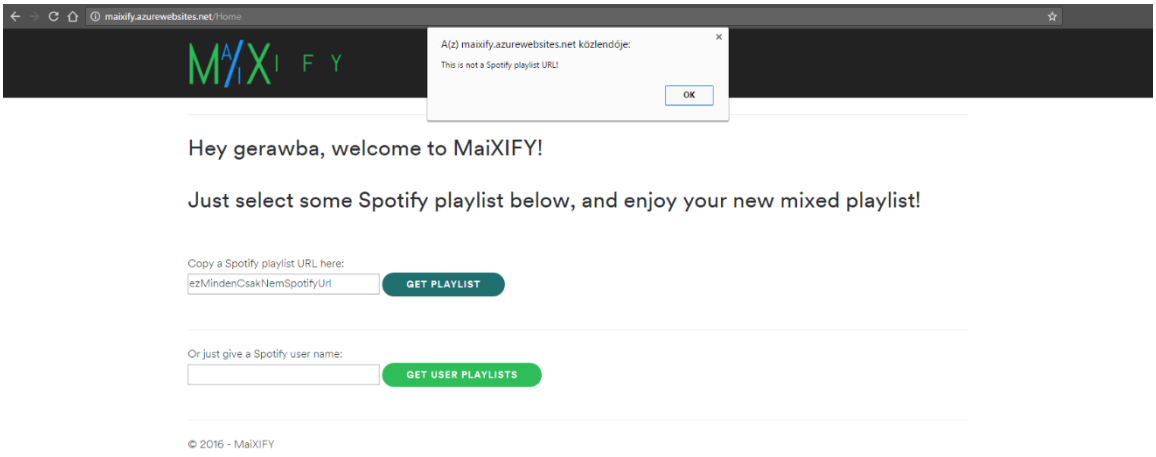


6. ábra: MaiXIFY főoldal

Érvényes URL-el megadott lejátszási lista esetén kliensoldalon az URL-ből kinyerjük a Spotify felhasználó azonosítóját és a lejátszási lista azonosítóját, majd ezeket az adatokat felküldjük (AJAX hívással) a szerver *Home/GetPlaylist* metódusának, ami válaszként visszaküldi a lekérdezett lejátszási listát. Ezt egy widget segítségével jelenítjük meg:

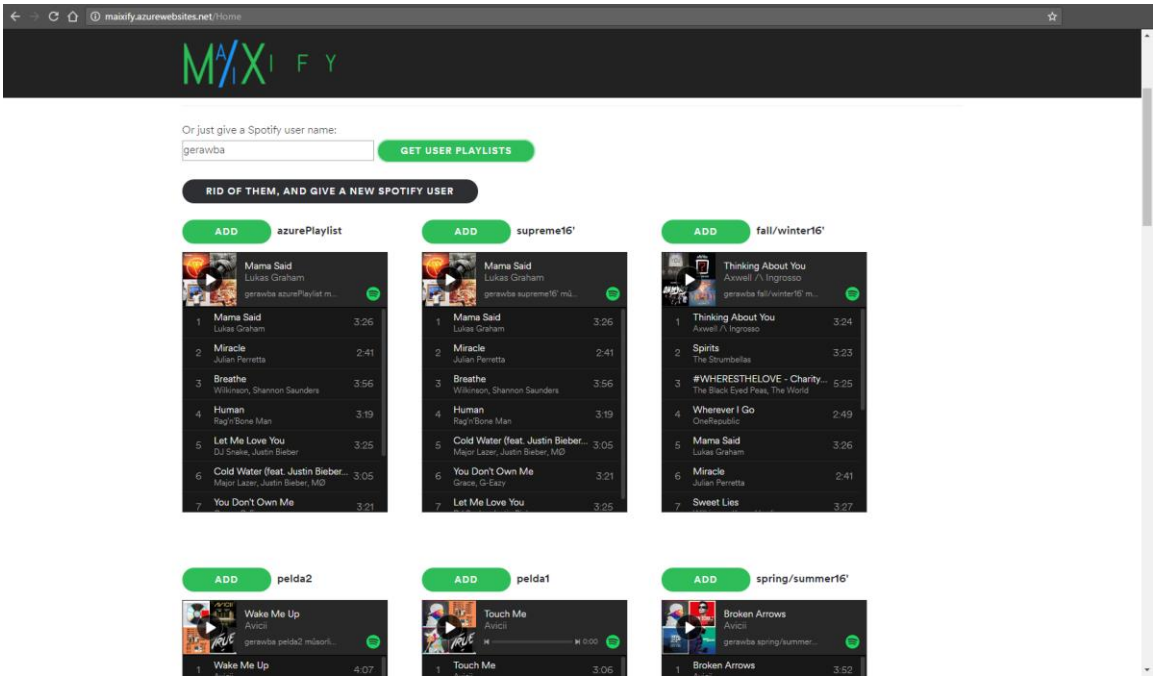


7. ábra: Lejátszási lista lekérése URL alapján



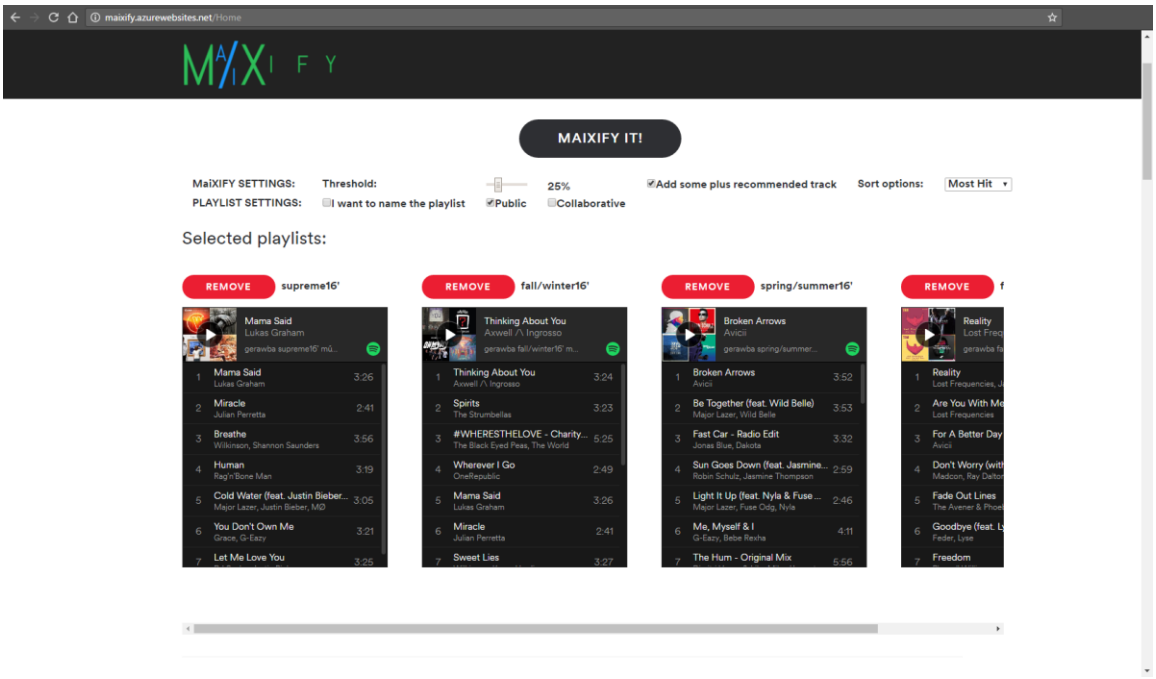
8. ábra: Hibásan megadott lejátszási lista URL

Ha egy felhasználó lejátszási listáit kérdezzük le, akkor a megadott felhasználó azonosítóját AJAX hívással felküldjük a szerver *Home/GetUserPlaylists* metódusának és amennyiben valóban létező felhasználót adtunk meg, külön widgetekben megjelenítjük a lejátszási listákat:



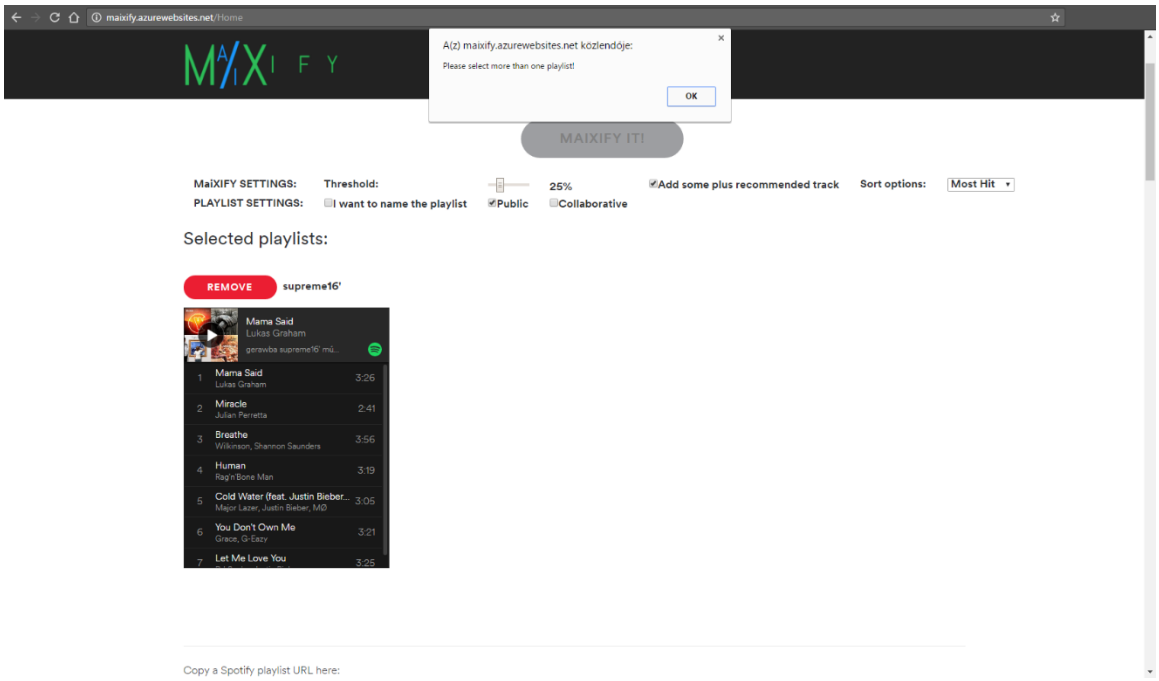
9. ábra: Felhasználóhoz tartozó lejátszási listák

A lejátszási listákat az „ADD” gombra kattintva tudjuk kiválasztani. Ekkor az üdvözlőszöveg alatt megjelennek a kiválasztott lejátszási listák, a generálásra vonatkozó beállítások valamint a lejátszási listára vonatkozó beállítások. A „MAIXIFY IT!” feliratú gomb segítségével generálhatunk lejátszási listát a kiválasztott lejátszási listák és beállítások alapján.



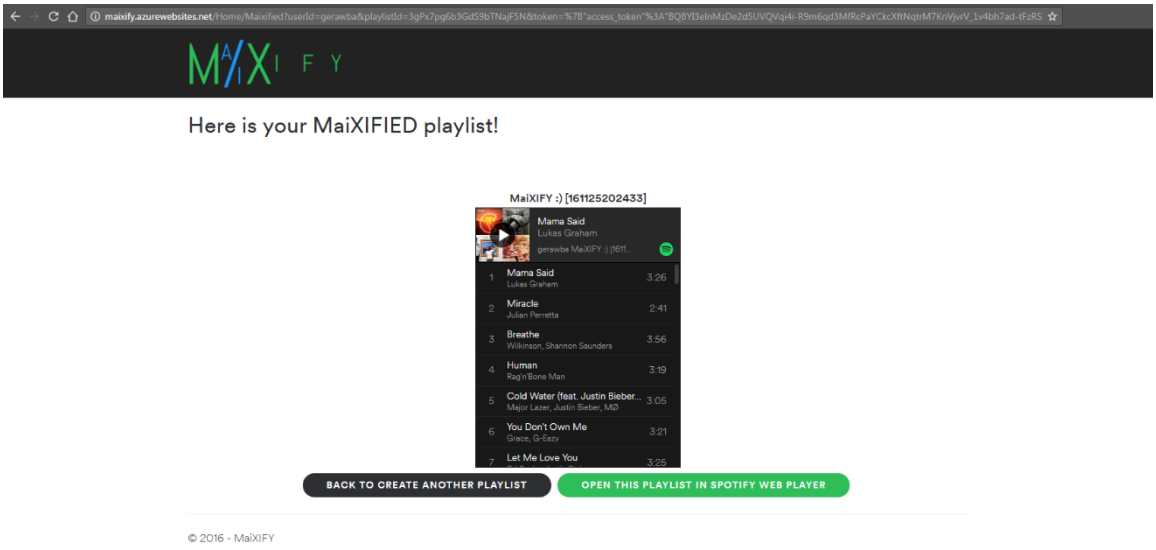
10. ábra: Kiválasztott lejátszási listák és beállítások

Amennyiben csak egy lejátszási listát adtunk meg és így szeretnénk lejátszási listát generálni hibaüzenetet kapunk és az oldal átirányít egy hibaüzenetet megjelenítő oldalra:

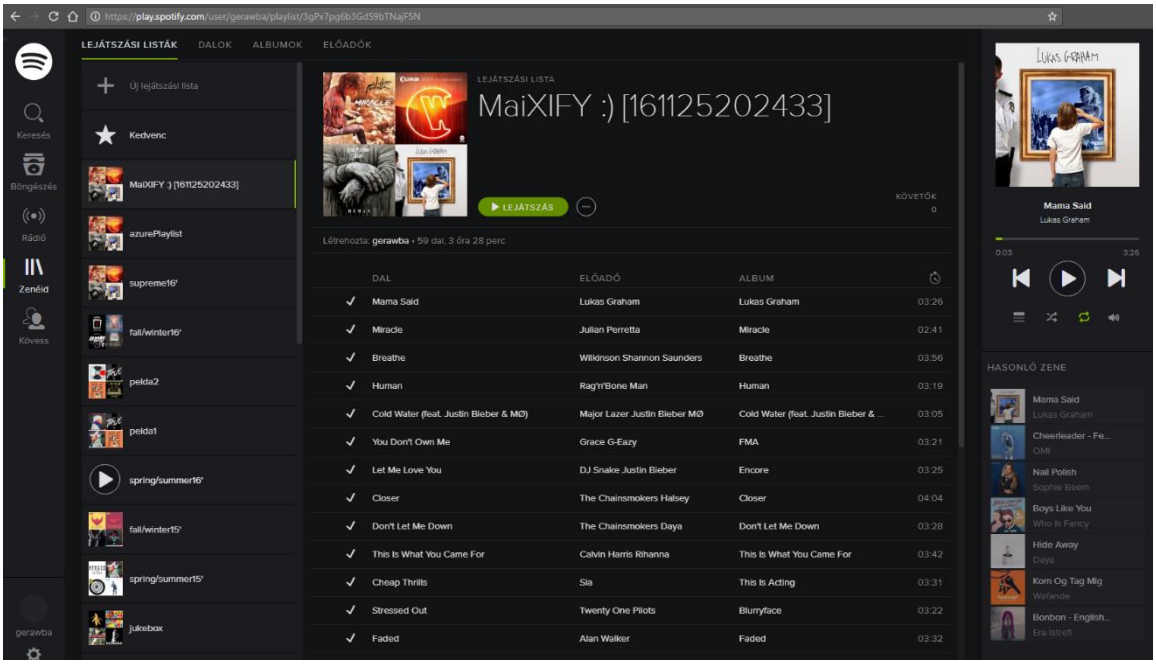


11. ábra: Túl kevés kiválasztott lejátszási lista

Amennyiben a kiválasztott lejátszási listák és a megadott beállítások alapján az alkalmazás képes lejátszási listát generálni, akkor az a „MAIXIFY IT!” gombra kattintva megjelenik a *Maixified.cshtml* oldalon – ezt az “OPEN THIS PLAYLIST IN SPOTIFY WEB PLAYER” gombra kattintva egy új fülön is megtekinthetjük a Spotify web playeren:



12. ábra: Maixified.cshtml a generált lejátszási listával



13. *ábra: Lejátszási lista megnyitása a Spotify oldalon*

AZURE WEB SITES

Az elkészített MA/IXIFY webalkalmazást az AZURE felhőbe telepítettük ki. A szerver megfelelő beállításai után az alkalmazás a következő címen érhető el:

http://maixify.azurewebsites.net

SECURITY

Az alkalmazás használatához a felhasználónak azonosítania kell magát a Spotify rendszer felé. Az autentikáció érvényes OAuth hozzáférési tokenek segítségével történik. Az azonosítási folyamat részletei megtalálhatóak a *Spotify Web API fejezet Authorization Code azonosítási folyamat* alfejezetében. Továbbá, amikor az alkalmazás átirányítja a felhasználót az autentikációt végző végpontra, a felhasználói adatok biztonságos https kapcsolaton keresztül kerülnek továbbításra. Az egyes lekérdezések fejlécében szereplő bizalmas adatok kódolva kerülnek átadásra.

GITHUB

A fejlesztés GitHub webes alapú repository segítségével történt. A fejlesztés alatt hasznosnak bizonyult a szolgáltatás verzió követésre szolgáló része. A GitHub repository a következő címen érhető el:

https://github.com/gerawba/MaiXIFY

Telepítési dokumentáció

Az alkalmazás nem igényel telepítést. Az alkalmazás használatához internetkapcsolatra és egy korszerű böngészőre van szükség. Támogatott böngészők: Internet Explorer 10+, Mozilla Firefox 24.0+, Google Chrome, Safari 9, Opera 30+.

Az alkalmazás használatához a JavaScript-nek telepítve és engedélyezve kell lennie (azon böngészők esetén, amelyeknél ezt külön meg kell tenni).

Továbbfejlesztési lehetőségek

Az alkalmazás többféle módon is továbbfejleszthető

- GUI továbbfejlesztése (mobil eszközökön szebb megjelenítés)
- frontend átalakítása (template-k bevezetése és használata az egyes nézetekben)
- lejátszási listát generáló algoritmus továbbfejlesztése (több szempont figyelembevétele a lista generálásánál, kifinomultabb algoritmus)