Interlinked

**Team 18-Product Backlog**

*Andrew Furman, Nick Galecki, Kevin Matson, Josh Shepler, and Doug York*

## Problem Statement

Although music is widely available through multiple platforms online, there is not currently a system for managing playlists between platforms that is both easy to use and accessible. Our project aims to allow playlists from different platforms to be exchanged, managed, and modified from one device.

## Background Information:

### Audience

Music consumers have different needs, and therefore use different streaming platforms. Over time, consumers may choose to move to a different online music service. If they are among music listeners who enjoy curating their own playlists with hundreds of songs, then they would face the difficult task of manually adding these songs one-by-one to a new playlist all over again, every time they want to use a different music streaming service.

### Similar Platforms

A similar platform exists called Stamp. Stamp has the ability to transfer song playlists from one platform to another, and they support a multitude of platforms, some more popular than others. However, the free version of this app is incredibly barebones, allowing only 10 songs to be moved from a single playlist per session. The free version of the app provides support for Windows, Mac, iOS, and Android. Unlimited playlist transfers for one specific device comes at a cost of 10 USD, but there is also an option to have unlimited playlist transfers for all of your devices for a one-time payment of 15 USD.

### Limitations

While Stamp has this basic functionality, its main limitation is that it only allows for a limited number of song transfers before requiring a one-time user fee. This severely limits its usability for users on a budget. In addition, its user interface is clunky and is generally slow to transfer music, only being able to move a handful of songs at a time. Our project will contain much more functionality in playlist management, will function without cost to the user, and will have a clean, responsive user interface.

### Functional Requirements:

* 1. As a user, I want to know which music streaming services are supported.
  2. As a user, I want to import a playlist from Amazon Music.
  3. As a user, I want to export a playlist to Amazon Music.
  4. As a user, I want to import a playlist from Apple Music/iTunes.
  5. As a user, I want to export a playlist to Apple Music/iTunes
  6. As a user, I want to import a playlist from Spotify.
  7. As a user, I want to export a playlist to Spotify.
  8. As a user, I want to import a playlist from Youtube.
  9. As a user, I want to export a playlist to Youtube.
  10. As a user, I want to be notified when a playlist is imported successfully.
  11. As a user, I want the transfer of playlists to be continued if it is interrupted.
  12. As a user, I want to be notified when a playlist is exported successfully.
  13. As a user, I want to be notified when a song can't be exported to a site.
  14. As a user, I want to search for songs in the app.
  15. As a user, I want to open a searched song in the corresponding platform for listening.
  16. As a user, I want to choose which platforms I search from.
  17. As a user, I want to filter search results.
  18. As a user, I want to add songs from Amazon Music to my playlist.
  19. As a user, I want to add songs from Apple Music/iTunes to my playlist.
  20. As a user, I want to add songs from Spotify to my playlist.
  21. As a user, I want to add songs from Youtube to my playlist.
  22. As a user, I want to remove songs from my playlist.
  23. As a user, I want to save my playlists.
  24. As a user, I want to delete my playlists.
  25. As a user, I want to revert a playlist back to a state it was in the past.
  26. As a user, I want to organize my playlists by song title.
  27. As a user, I want to organize my playlists by album title.
  28. As a user, I want to organize my playlists by artist name.
  29. As a user, I want to organize my playlists by music genre.
  30. As a user, I want to merge two or more playlists into a new playlist.
  31. As a user, I want the app to listen to a song and put it into my playlist.
  32. As a user, I want to share my playlist with a friend.
  33. As a user, I want to suggest songs to a friend.
  34. As a user, I want the program to suggest songs to add to my playlists.
  35. As a user, I want the program to run on a computer.
  36. As a user, I want the program to run on an Android device.
  37. As a user, I want to use this service on a web browser.
  38. As a user, I want the app playlists to be transferable cross-platform.
  39. As a user, I want the GUI to be intuitive and simple.
  40. As a user, I want to have my playlists automatically synchronized between platforms.
  41. As a developer, I want to ensure security of user data.
  42. As a developer, I want an API that allows for universal importing from sites without needing a separate method for each site.
  43. As a developer, I want an API that allows for universal exporting to sites without needing a separate method for each site.
  44. As a developer, I want the playlists to be independent from their source(s).

### Non-Functional Requirements:

#### Architectural

The program backend is to be a Java-based program running on a device/server. The back end will be capable of handling at least 100 user queries simultaneously, each query being resolved ideally within 500 ms, but this depends on the music platform’s network speed and size of the query. These would allow for at least 17 million queries to be resolved per day.

The frontend will depend on the platform developed on. On Android devices, the Android Studio Layout Editor will be used. On PC, the Java Swing and AWT libraries will be used to interface the back-end with the user.

#### Security

As a user, I want proper data security protocols to be followed to ensure the privacy of my user data from various music streaming services, and also that any data needing to be stored is protected by encryption. The data itself will be stored on the device with the application, therefore, the security is ultimately reliant on the user to not share the files with authentication data.

#### Usability

The user interface(s) are to be both intuitive and responsive, with simple menu layouts that can be easily used by the average music consumers. The interface will be similar across platforms to ensure that the user will have minimal confusion when using a different platform.

#### Legality

The program will function within all current copyright law, especially in regards to music. As a result, playing music will only be accessible through third parties.