Clef
CPSC 2350 Project

Instructor:
Parsa Rajabi

Team:
Arshpreet Kaur
Chao Xu
Darrick He

Github Repository: https://github.com/DivoHub/2350_project

Table of Contents

Overview

With the growing user-base and variety of music streaming services, users can often switch companies and memberships without any transitional support.
Members who spend hours curating their playlists can find themselves trapped in their respective companies - deterred by having to start over, or painstakingly copying playlists song by song.

Clef is a web application that helps users transfer playlists between their favourite music services by automating that process.
Clef utilises public APIs to interact with each music streaming service to pull data and create playlists on the user's behalf.

SDLC Model

The Software Development Life Cycle model we chose was Agile Kanban.

We chose Agile because of its flexibility for feature changes, as well as its aptitude for smaller teams. In order for Clef to reach a larger user-base, additional music streaming services will need to be included. In turn, more APIs will be implemented with additional features to be added. Agile's structure will help us iterate these features by segmenting each feature into 'deliverables'.
Furthermore, we chose Agile due to its current relevance and ubiquity in contrast to other plan-driven methodologies which are considered outdated, or for larger teams.

We chose Kanban as our Agile framework because it allows us to see project progress in visuals. It also appeared to be the most appropriate for collaborating remotely since the daily meetup structure of Scrum may not be feasible. Kanban allows us to work asynchronously without compromising on efficiency.

User Stories

User Persona: Kate Watson

Name: Kate Watson
Age: 28 years old
Location: Vancouver, BC
Description: Kate works as a Cast Member at Cineplex. She loves to hear different genres of music and always recommends others her findings. Kate believes music is good for the soul, and looks forward to the weekly recommendations provided by Spotify. However, Kate is finding that Spotify has been recommending the same music repeatedly, and she wants to move to a new music streaming service to get new experiences.

As a Cast Member at Cineplex, Kate wants to be able to close the theatre while listening to music. Since she would be busy working, she would not be able to constantly change the music herself. Instead, she wants a service that can play recommendations on her behalf.

User Persona: Loren Smith

Name: Loren Smith
Age: 40 years old
Location: Burnaby, BC
Description: Loren Smith lives with her husband and her two kids. Loren works as a general physician at a public hospital. As a medical professional, Loren believes living a healthy lifestyle is paramount. Every morning before work, she likes to go on morning runs while listening to podcasts. Recently, she purchased an Apple Watch to help her consolidate fitness statistics, as well as providing her a means to listen to podcasts while running. Since Loren is now a member in the Apple ecosystem, Apple Music seemed like the most appropriate choice for a music streaming service. However, Loren finds that the selection of podcasts in Apple Music is quite limited. Thus, she is open to the idea of changing streaming services to accommodate her podcast needs, but is fearful that she will lose all of her playlists on Apple Music.

As a busy medical professional and podcast enthusiast, Loren wants an application that can help expedite the transfer of her playlist data from Apple Music so that she does not have to spend a lot of her time moving playlists individually.

User Persona: David Jones

Name: David Eric Jones
Age: 15 years old
Location: Calgary, ON
Description: David is a high school student who lives with his parents. David's interests are listening to music, technology, and socialising with friends. David has been sharing an Apple Music account with his parents. David recently got a job at a local clothing store and is now making his own disposable income. All of David's friends use Spotify, and he would like to switch from Apple Music so that he and his friends can collaborate on playlists. Now that David can afford to pay for his own subscription, he wants to change services. Although David is tech-savvy, he can't seem to find a way out of Apple Music without finding a program to help him.

As a student, David is looking for an application which will help him to transfer all of his playlists from his Apple Music account to Spotify account so that he will save a lot of time for his study and job.

User Persona: Izumi Yo

Name: Izumi Yo
Age: 18 years old
Location: Portland, Oregon
Description: Izumi is a new college student who recently travelled to Portland from Kyoto, Japan to study English. He likes listening to all kinds of music, but has recently developed a taste for Western music. Like most of the youth in Japan, Izumi is a Spotify user. However, Izumi is now a college student and is very cautious about his spending. He does not mind changing from Spotify as long as he can find a good promotional deal for students in order to save money. Despite being a teenager, Izumi is not very well versed with computers and technology, and will always gravitate towards programs that have intuitive and clean designs.

As a non-tech-savvy college student, Izumi wants a program that is easy to understand and use in the event that he has to change music streaming services to save money and time.

Technology Stack

For our technology stack, we chose the following languages and technologies:

Backend: **Javascript/Node.js, Python**

Frontend: **CSS, HTML**

Frameworks: **Express.js**

APIs: **Spotify API, Apple Music API, (Deezer API if not time constrained)**

In an attempt to avoid learning new languages/technologies to save time and resources, it was most appropriate to choose common languages between our team. These languages include the following: Javascript, Java, C++, HTML, and CSS.

Despite Java and C++ being common languages between us, we chose to exempt both since there is no need to use lower-level, compiled languages. This will save us from having to learn new technologies and implementation methods to save time.

Since most of the backend programming is centred around JSON data creation/manipulation, it was apt for us to implement our web application in Javascript to mitigate any portability/compatibility issues. Consequently, we will be using Node.js to run our server-side applications for data retrieval, and processing.

Due to previous experience working with Python libraries such as Requests, Beautiful Soup, and Selenium, Darrick will be using Python 3 if circumstances show that any of the mentioned libraries need to be utilised.

We decided to use Express.js as our framework due to Chao, and Arshpreet's knowledge of it and its ubiquity.

APIs that we are implementing from Apple Music and Spotify are all public web services/APIs provided by the respective companies themselves. These APIs share similar features that include:
Retrieve contents of playlists
Search for album, song, or artist
Retrieve chart information regarding song/artist statistics
User ID/URL information

API

Chosen APIs

API (Application Programming Interface) is a set of protocols that allow two different programs to interact with each other.

The two starting APIs used in Clef are:

- Spotify API
- Apple Music API

Spotify and Apple Music are two of the largest platforms for music streaming. Conveniently, both Apple and Spotify provide public APIs for developers based on REST principles.
Clef will utilise these APIs to implement the following features:

Features

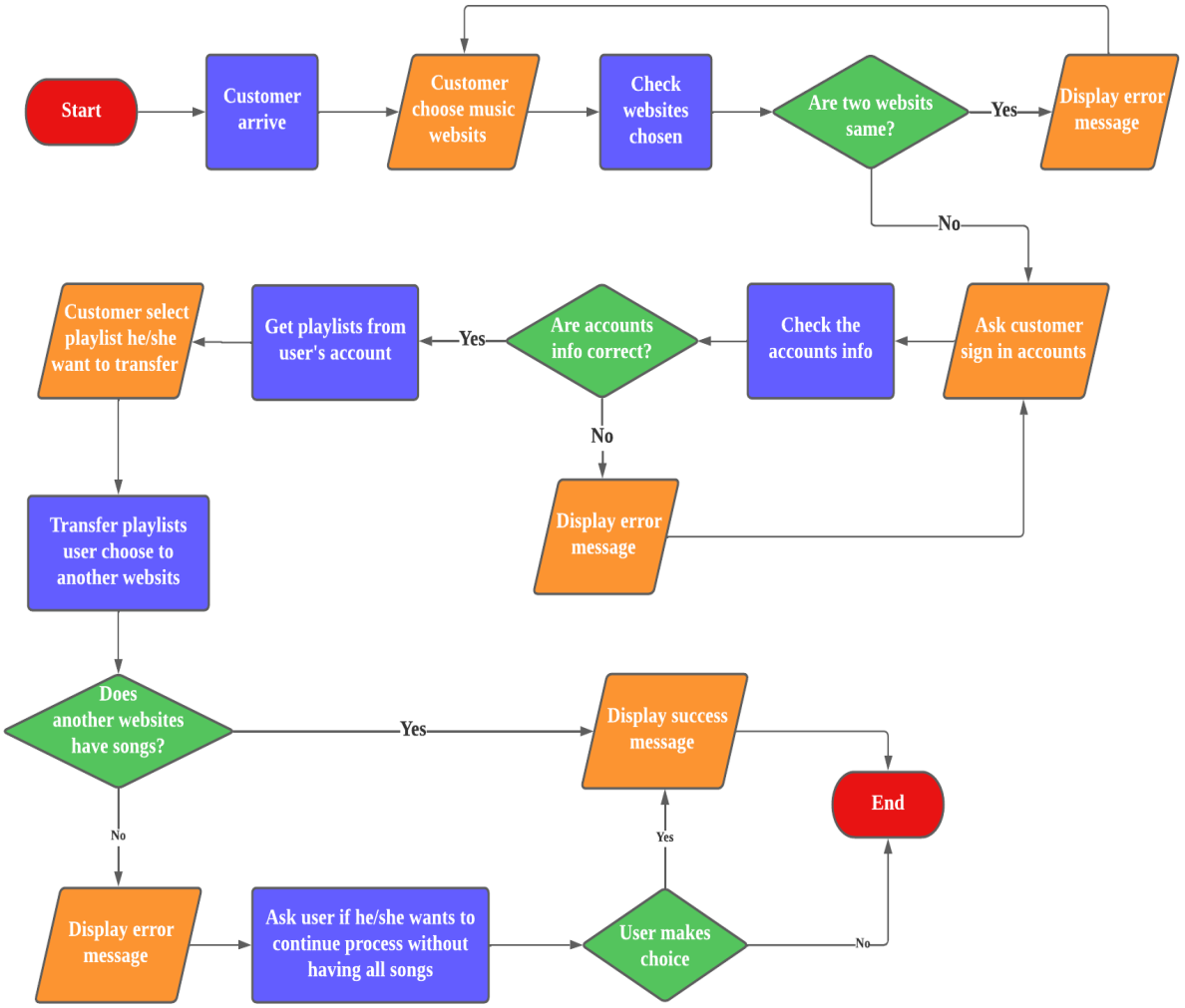
Spotify API:

- Clef will pull/retrieve song data (GET) from a Spotify playlist including song name, artist, album, year, and genre, and will consolidate the data into a more portable format such as JSON.
- Expediting data between music services can often be a challenge due to many songs having similar information. To clarify that the music is correct, the user should be able to play the song from the web browser using the Web Playback SDK.
- Clef will parse through a JSON file, and search for songs within the Spotify registry for matching information. If a song matches the information provided by the JSON file, Clef will place the songs into a new playlist which can then be pushed to a new/existing playlist in the target account (PUT).

Apple Music API:

- Apple Music API already provides resource information in JSON format which includes song name, album, artist, and playlist. Clef will utilise this information by sending a (GET) request.
- Use Apple Music API to preview songs before adding/removing/transferring them onto target playlists.
- Provide authentication tokens to allow users to login to their Apple account if playlist information, or playlist editing permissions is not accessible to the public.

Data Flow Diagrams



Task #	Task	Assigned To	Estimated Hours	Actual Hours	Start Date	Due Date
1	**Planning**		13.25	14.25	01/22/2022	02/12/2022
1.1	Brainstorm ideas for Project	Team	1	3	01/22/2022	01/23/2022
1.1.1	User stories	Arshpreet	1	1	02/11/2022	02/12/2022
1.2	Determine appropriate SDLC	Team	2.5	0.5	02/07/2022	02/08/2022
1.2.1	Research different SDLC	Team	2	1.5	02/07/2022	02/08/2022
1.2.2	Setup Kanban interface	Darrick	0.5	0.5	02/08/2022	02/09/2022
1.3	Choose technology stack	Team	3	3	02/10/2022	02/12/2022
1.3.1	Choose programming languages	Team	0.5	1	02/10/2022	02/11/2022
1.3.2	Find compatible APIs for project	Team	0.5	0.5	02/10/2022	02/11/2022
1.3.3	Research dependencies/licenses	Team	2	2	02/11/2022	02/12/2022
1.4	Setup Github Repository	Darrick	0.25	0.25	02/02/2022	02/03/2022
1.5	Team work plan	Team	0.5	0.5	02/02/2022	02/03/2022
1.5.1	Assign roles amongst team	Team	0.25	0.25	02/02/2022	02/03/2022
1.5.2	Determine communication tools	Team	0.25	0.25	02/02/2022	02/03/2022
2	**GUI**		15	12	02/12/2022	02/25/2022
2.1	Draw layout	Darrick	0.5	0.5	02/12/2022	02/13/2022
2.1.1	Draw rough draft	Darrick	0.5	0.5	02/12/2022	02/13/2022
2.1.2	Research drawing tools	Chao	2	2.5	02/13/2022	02/14/2022
2.1.3	Create Wireframe	Chao	2	3	02/15/2022	02/17/2022
2.1.4	Improve fidelity of prototype	Chao	2	3	02/16/2022	02/18/2022
2.2	Style	Team	3		02/16/2022	02/24/2022
2.2.1	Determine flow of website	Chao, Darrick	0.5	1	02/16/2022	02/17/2022
2.2.2	Create data flow diagram	Chao	1.5	1.5	02/16/2022	02/17/2022
2.2.3	Column structure	Team	0.5		02/22/2022	02/23/2022
2.2.4	Colour scheme	Team	0.25		02/23/2022	02/24/2022
2.2.5	Determine appropriate font-family	Arshpreet	0.25		02/23/2022	02/24/2022
2.3	Graphics	Chao, Arshpreet	1		02/23/2022	02/25/2022
2.3.1	Source logos/pictures	Arshpreet	0.5		02/23/2022	02/24/2022
2.3.2	Overview of picture licenses	Chao	0.5		02/24/2022	02/25/2022
3	**Implementation**		52		02/25/2022	03/29/2022
3.1	Prepare resources	Team	4		02/25/2022	02/29/2022
3.1.1	Install appropriate software	Team	1		02/25/2022	02/26/2022
3.1.2	Research API implementation	Team	2		02/25/2022	02/27/2022
3.1.3	Research server authentications	Arshpreet	1		02/27/2022	02/29/2022
3.2	Create basic HTML/CSS index	Chao	5.5		02/28/2022	03/01/2022
3.2.1	Build overall framework	Chao	2		03/01/2022	03/02/2022
3.2.2	Add specific elements	Chao	2.5		03/03/2022	03/04/2022
3.2.3	Resizing elements	Chao	1		03/03/2022	03/06/2022
3.3	Create Node.js scripts for website	Team	19		03/07/2022	03/19/2022
3.3.1	Code function to convert playlist data into JSON	Darrick	2		03/07/2022	03/09/2022
3.3.2	Code function to convert JSON into playlist data	Darrick	2		03/09/2022	03/11/2022
3.3.3	Code function to traverse Spotify registry	Darrick	2		03/11/2022	03/13/2022
3.3.4	Code function to traverse Apple Music registry	Darrick	2		03/13/2022	03/14/2022
3.3.5	Create function to Create/Append songs to Target	Arshpreet	1.5		03/13/2022	03/14/2022
3.3.6	Create function to play music on web page	Arshpreet	1.5		03/12/2022	03/14/2022
3.3.7	Create User-facing Output Log	Darrick	4		03/14/2022	03/16/2022
3.3.8	Manage Event Handler	Chao	1		03/15/2022	03/16/2022
3.3.9	Create/Manage Exception Handler/Error Logging	Arshpreet	3		03/17/2022	03/19/2022
3.4	Link appropriate URL/Directories	Arshpreet	1.5		03/18/2022	03/19/2022
3.5	Finalize demo of Web App	Team	3		03/19/2022	03/21/2022
3.6	Testing	Team	14		03/21/2022	03/30/2022
3.6.1	Identify all user cases	Team	2		03/21/2022	03/22/2022
3.6.2	Build test cases for Application	Team	1		03/21/2022	03/22/2022
3.6.3	Stress test application with test cases	Chao, Arshpreet	1		03/22/2022	03/24/2022
3.6.4	Create tickets for found bugs	Chao, Arshpreet	2		03/22/2022	03/25/2022
3.6.5	Debug	Darrick	4		03/24/2022	03/27/2022
3.6.6	Refactor	Darrick	4		03/26/2022	03/28/2022
3.7	Research deployment	Chao	2		03/28/2022	03/30/2022
3.7.1	Obtain appropriate resources for deployment	Team	1		03/28/2022	03/30/2022
3.7.2	Mirror application onto hosting service	Chao	4		03/28/2022	03/30/2022
4	**Closure/Maintenance**		22		03/28/2022	04/02/2022
4.1	Documentation	Team	4		03/28/2022	03/29/2022
4.1.1	Create Documentation	Darrick	3		03/28/2022	03/29/2022
4.1.2	Edit README file	Darrick	0.5		03/29/2022	03/30/2022
4.1.3	Final edits after customer feedback	Team	0.5		03/29/2022	03/30/2022
4.2	SDLC breakdown	Team	2		02/16/2022	02/18/2022
4.3	WBS	Team	4	5	02/16/2022	02/18/2022
4.4	Schedule/Timing	Team	4	5	02/16/2022	02/18/2022
4.5	Sample Output	Team	2		03/30/2022	03/31/2022
4.6	Presentation preparation	Team	3		03/30/2022	03/31/2022
4.6.1	Create Slides	Team	1.5		03/31/2022	04/01/2022
4.6.2	Adding citations	Team	0.5		03/31/2022	04/01/2022
4.7	Finalize documentation	Team	4		04/01/2022	04/02/2022
4.8	Finalize push to alpha stage for application	Team	4		04/01/2022	04/02/2022

Project Schedule/Timeline

Clef

Task #	Task	Jan W3	Jan W4	Feb W1	Feb W2	Feb W3	Feb W4	Mar W1	Mar W2	Mar W3	Mar W4	Apr W1	Apr W2
1	**Planning**												
1.1	Brainstorm ideas for Project												
1.1.1	User stories												
1.2	Determine appropriate SDLC												
1.2	Research SDLCs												
1.2.1	Setup Kanban interface												
1.3	Choose technology stack												
1.3.1	Choose programming languages												
1.3.2	Find compatible APIs for project												
1.3.3	Research dependencies/licenses												
1.4	Setup Github Repository												
1.5	Team work plan												
1.5.1	Assign roles amongst team												
1.5.2	Determine communication tools												
2	**GUI**												
2.1	Draw layout												
2.1.1	Draw rough draft												
2.1.2	Research drawing tools												
2.1.3	create wireframe												
2.1.4	improve fidelity of prototype												
2.2	Style												
2.2.1	determine flow of website												
2.2.2	create data flow diagram												
2.2.3	column structure												
2.2.4	colour scheme												
2.2.5	determine appropriate font-families												
2.3	graphics												
2.3.1	source logos/pictures												
2.3.2	overview of picture licenses												
3	**Implementation**												
3.1	prepare resources												
3.1.1	install appropriate software												
3.1.2	research API implementation												
3.1.3	research server authentications												
3.2	create basic HTML/CSS index												
3.2.1	build overall framework												
3.2.2	add specific elements												
3.2.3	resizing elements												
3.3	create Node.js scripts for website												
3.3.1	Code func to convert playlist data into JSON												
3.3.2	Code func to convert JSON into playlist data												
3.3.3	Code func to traverse Spotify registry												
3.3.4	Code function to traverse Apple Music reg.												
3.3.5	Create function to Create/Append songs												
3.3.6	Create function to play music on web page												
3.3.7	Create User-facing output Log												
3.3.8	Manage Event Handler												
3.39	Create exception handler/Error logging												
3.4	Link appropriate URL/Directories												
3.5	Finalize Demo of Web App												
3.6	Testing												
3.6.1	Identify all user cases												
3.6.2	build test cases for application												
3.6.3	stress test application with test cases												
3.6.4	create tickets for found bugs												
3.6.5	debug												
3.6.6	refactor												
3.7	research deployment												
3.7.1	obtain appropriate resources for deployment												
3.7.2	Mirror application onto hosting service												
4	**Closure/Maintenance**												
4.1	Documentation												
4.1.1	Create documentation												
4.1.2	Edit README file												
4.1.3	Final edits after feedback												
4.2	SDLC breakdown												
4.3	WBS												
4.4	Schedule/Timing												
4.5	Sample Output												
4.6	Presentation												
4.6.1	Slideshow												
4.6.2	Citations												
4.7	Finalize Documentation												
4.8	Finalize push to alpha stage												

Legend: Arshpreet Chao Darrick Team

Clef

Easy way to transfer your playlists between Spotify and Apple Music

Get Start

Clef

Easy way to transfer your playlists between Spotify and Apple Music

Playlists from source:

Source:

- Spotify
- Apple Music

Target:

- Spotify
- Apple Music

Options

- [ ] Convert chosen playlist(s)
- [ ] Convert all playlist(s)

Start

Log:

Playlists from target:

Doucumentation

About

Contact

Clef

Easy way to transfer your playlists between Spotify and Apple Music

Playlists from source:

Spotify

Account:

PIN:

Log in

Apple Music

Account:

PIN:

Log in

Options

Convert chosen playlist(s)

Convert all playlist(s)

Start

Log:

Playlists from target:

Doucumentation

About

Contact

Clef

Easy way to transfer your playlists between Spotify and Apple Music

Playlist(s) from source:

- [ ] Playlist 1
- [ ] Playlist 2
- [ ] Playlist 3
- [ ] Playlist 4

Spotify

Username: User 1

Apple Music

Username: User 1

Options

- [ ] Convert chosen playlist(s)
- [x] Convert all playlist(s)

Start

Log:

User successfully logged in!

Playlist(s) found.

Start convert...

Playlist 1 finished...
Playlist 2 finished...
Playlist 3 finished...
Song 1 in playlist 4 not found in target service...
Playlist 4 finished...

All playlist(s) transferred!

Playlist(s) from target:

- [ ] Playlist 1
- [ ] Playlist 2
- [ ] Playlist 3
- [ ] Playlist 4

Clef

Easy way to transfer your playlists between different music services

Start

Clef

Easy way to transfer your playlists between different music services

Playlist(s) from source:

Source:

- Spotify
- Apple Music

Target:

- Spotify
- Apple Music

Next

Options

- [ ] Convert chosen playlist(s)
- [ ] Convert all playlist(s)

Start

Log:

Playlist(s) from target:

Doucumentation

About

Contact

Clef

Easy way to transfer your playlists between different music services

Playlist(s) from source:

Playlist(s) from target:

Error

You cannot choose the same music service for both source and target. Please click Back button to choose services again.

Back

Options

- [ ] Convert chosen playlist(s)
- [ ] Convert all playlist(s)

Start

Log:

Clef

Easy way to transfer your playlists between different music services

Playlist(s) from source:

Source:

- Spotify
- Apple Music

Target:

- Spotify
- Apple Music

Next

Options

- [ ] Convert chosen playlist(s)
- [ ] Convert all playlist(s)

Start

Log:

Playlist(s) from target:

Doucumentation

About

Contact

Clef

Easy way to transfer your playlists between different music services

Playlist(s) from source:

Spotify

Account:

PIN:

Log in

Music

Apple Music

Account:

PIN:

Log in

Options

Convert chosen playlist(s)

Convert all playlist(s)

Start

Log:

Playlist(s) from target:

Doucumentation

About

Contact

Clef

Easy way to transfer your playlists between different music services

Playlist(s) from source:

- [ ] Playlist 1
- [ ] Playlist 2
- [ ] Playlist 3
- [ ] Playlist 4

Spotify

Spotify

Username:
User 1

Log out

Music

Apple Music

Username:
User 1

Log out

Playlist(s) from target:

- [ ] Playlist 1
- [ ] Playlist 2
- [ ] Playlist 3
- [ ] Playlist 4

Options

- [ ] Convert chosen playlist(s)
- [x] Convert all playlist(s)

Start

Notification

Song 1 and song 2 are not found in target music service, do you want to continue process without these songs?

Yes

No

Log:

User successfully logged in!

Playlist(s) found.

Start convert...

Playlist 1 finished...
Playlist 2 finished...
Playlist 3 finished...
Song 1 in playlist 4 not found in target service...
Song 2 in playlist 4 not found in target service...

Clef

Easy way to transfer your playlists between different music services

Playlist(s) from source:

- [ ] Playlist 1
- [ ] Playlist 2
- [ ] Playlist 3
- [ ] Playlist 4

Spotify

Spotify

Username:
User 1

Log out

Music

Apple Music

Username:
User 1

Log out

Playlist(s) from target:

- [ ] Playlist 1
- [ ] Playlist 2
- [ ] Playlist 3
- [ ] Playlist 4

Options

- [ ] Convert chosen playlist(s)
- [x] Convert all playlist(s)

Start

Notification

Song 1 and song 2 are not found in target music service, do you want to continue process without these songs?

Yes

No

Log:

User successfully logged in!

Playlist(s) found.

Start convert...

Playlist 1 finished...
Playlist 2 finished...
Playlist 3 finished...
Song 1 in playlist 4 not found in target service...
Song 2 in playlist 4 not found in target service...
Playlist 4 finished without Song 1 and Song 2...

All playlist(s) transferred!

Doucumentation

About

Contact

Clef

Easy way to transfer your playlists between different music services

Playlist(s) from source:

- [ ] Playlist 1
- [ ] Playlist 2
- [ ] Playlist 3
- [ ] Playlist 4

Spotify

Spotify

Username:
User 1

Log out

Music

Apple Music

Username:
User 1

Log out

Playlist(s) from target:

- [ ] Playlist 1
- [ ] Playlist 2
- [ ] Playlist 3
- [ ] Playlist 4

Options

- [ ] Convert chosen playlist(s)
- [x] Convert all playlist(s)

Start

Notification

Song 1 and song 2 are not found in target music service, do you want to continue process without these songs?

Yes

No

Log:

User successfully logged in!

Playlist(s) found.

Start convert...

Playlist 1 finished...
Playlist 2 finished...
Playlist 3 finished...
Song 1 in playlist 4 not found in target service...
Song 2 in playlist 4 not found in target service...
Stop converting Playlist 4...

All playlist(s) transferred except playlist 4!