Spotify to YTMusic converter

Arsenii Pogodin
ČVUT–FIT
pogodars@fit.cvut.cz

December 9, 2023

1 Introduction

This report outlines the development of a Spotify to Youtube Music converter. The main objective of this project was to eliminate the need for time-consuming manual searches, making the transfer process a breeze, gain experience from first major project using GUI libraries, multithreading and API calls

2 Methods/procedures/algorithms

The first big challenge for me was Spotify API calls. As I had not previously worked with the requests [3] library, there was a learning curve involved. Additionally, I recognized the need to establish my own server for receiving the access token from the Spotify callback. I used Flask [6] , which is micro web framework written in Python. Upon successful token retrieval, I faced the challenge of securely storing this sensitive information. Presently, the tokens are stored in a text file, a solution that warrants improvement for enhanced security. After then it went smoothly, leveraging my familiarity with the requests [3] library and I implemented functions for retrieving all logged in user playlists and songs in those playlists. Notably, there was no necessity to create a YTMusic API Handler, as I opted for the use of the open-source library ytmusicapi[5]. The graphical interface design was inspired by examples provided in the customtkinter GitHub repository[4].

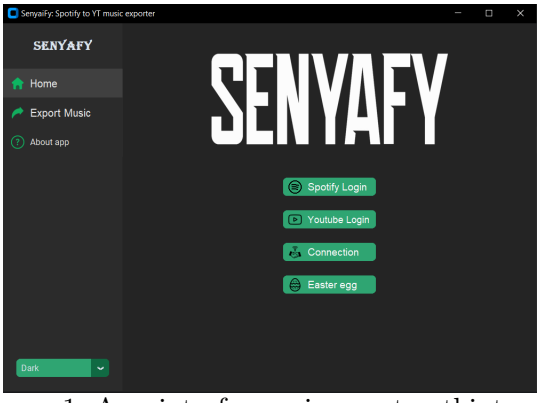

Figure 1: App interface using customtkinter [4]

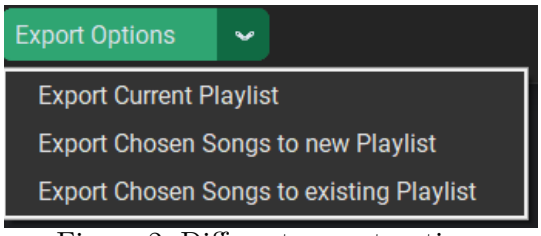

Figure 2: Different export options

3 Results

The implementation of the application has proven to be successful in transferring playlists or selected songs from Spotify to YouTube Music. The tests were implemented using pytest [1] library and encompassing real-life scenarios. Personally I exported a total of three playlists, comprising approximately 700 songs. During the testing phase, peculiar behavior was observed when searching for songs with names in the Cyrillic alphabet, leading to unexpected conflict error responses from the server. While the overall expectations were met, it's worth noting that the process of exporting more than 200 songs took longer than initially anticipated.

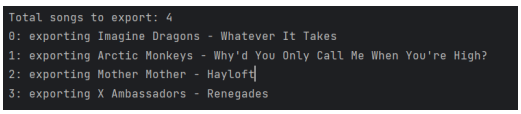

Figure 3: Exporting process

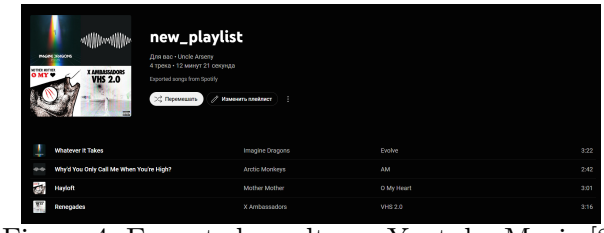

Figure 4: Exported results on Youtube Music [2]

4 Conclusion

The implementation is a useful tool providing a welcome alternative to the tedious process of exporting music. Several enhancements could be considered to further elevate the application's utility. Some

potential avenues for improvement include expanding support for additional music platforms, adding secure way to store user information and api keys, optimizing the transferring algorithm for enhanced efficiency, and introducing multithreading capabilities to enable simultaneous exports.

Reflecting on the project, I am genuinely pleased with the outcomes achieved. I eagerly anticipate the opportunity to refine and extend its capabilities in the future.

References

[1] Krekel et al. Python testing framework. online, 2023. [cit. 2023–08–12] https://pytest.org/.

[2] Google. music streaming service. online, 2023. [cit. 2023–08–12] https://music.youtube.com/.

[3] Kenneth Reitz. Http library for python, built for human beings. online, 2023. [cit. 2023–08–12] https://docs.python-requests.org/en/master/.

[4] Tom Schimansky. A modern and customizable python ui-library based on tkinter. online, 2023. [cit. 2023–08–12] https://github.com/TomSchimansky/CustomTkinter.

[5] sigma67. Unofficial api for youtube music. online, 2023. [cit. 2023–08–12] https://github.com/sigma67/ytmusicapi.

[6] Pallets Team. Flask documentation. online, 2023. [cit. 2023–08–12] https://flask.palletsprojects.com/en/2.3.x/.