# Chatbot and Recommendation System – Final Project

Louis TEMPE    Ferdinand VALANCOGNE

## I. Data used

For this project, we decided to use a music API. We took the Spotify API that have lot of information about tracks, albums and artists of the Spotify platform. In order to access the data easier, we used the spotipy library from python. This module permits us to access the data in a few command lines, and a loop permits us to get all first 1000 tracks by year from Spotify, from 2010 to 2021.

```
cid = '9ffe6a6cffed429cac2fa7448844442f'
secret = 'bf748672ad1b4498a3da1097dfd0de91'
client_credentials_manager = SpotifyClientCredentials(
    client_id=cid, client_secret=secret)
sp = spotipy.Spotify(client_credentials_manager=client_credentials_manager)
```

Once the code is executed, all data are saved in a *pandas* dataframe, that we can save then.

```
artist_names = []
artist_lists = []
track_names = []
album_names = []
times = []
popularities = []
track_ids = []
years = []
for year in range(2010, 2022):
    for i in range(0, 1000, 50):
        track_results = sp.search(
            q=f'year:{year}', type='track', limit=50, offset=i)
        for i, t in enumerate(track_results['tracks']['items']):
            artist_names.append([a['name'] for a in t['artists']])
            track_names.append(t['name'])
            album_names.append(t['album']['name'])
            times.append(t['duration_ms'] / 1000)
            track_ids.append(t['id'])
            popularities.append(t['popularity'])
            years.append(year)

df = pd.DataFrame({'artists': artist_names, 'track': track_names, 'album': album_names,
                    'time (s)': times, 'popularity':  popularities, 'year': years, 'id':
track_ids})
df.to_csv('tracks.csv')
```

## II. Method used

This chatbot is implemented in python with a regex method. We have a list of regular expression patterns, and when a message matches a pattern, this one is associated with an intent, and the answer will depend on it. The bot doesn't have only one answer by intents. In fact, we can provide a list of possible answers for each intent, and the program will choose one of them randomly.

Here are some examples of regular expression and the result we have:

```
rf'\W*live\W+in\W+(?P<city>([\w-]+ ?)) b'
```

This pattern matches a city where the user lives. When he says something like "I live in Paris" or "Actually, the truth is that I live in Paris", the bot will save in his memory the city, and will associate it in a dictionary to the username, and now it knows the user lives in Paris. It will then respond something like "Oh, Paris is a great city!".

Another simple example is that regular expression:

```
\bfrom (?P<artist>[a-z -]+)\b
```

Here, the bot will take all letters, space and – between the *from* and the end of the sentence, or the next special character. The program then will propose to listen a music from that artist, if it exists in our datas.

The bot is a discord bot, so we used the *discord* library, that permits us to get all information about a message (the text, the author, the channel, date, etc.), and for each message sent on any channel of the discord group, we will be able to do something.

To create the client, we do the command line

```
client = discord.Client()
```

We also created a dictionary named *users*, that will contain all information about each user. When a user is talking, if the username does not already exist in this dictionary, we put the username as a new key of the dictionary. Then, when the user will give a new information (his real name, the city he lives in, etc.), it will be saved in the dictionary associated with his username in the global dictionary *users*.

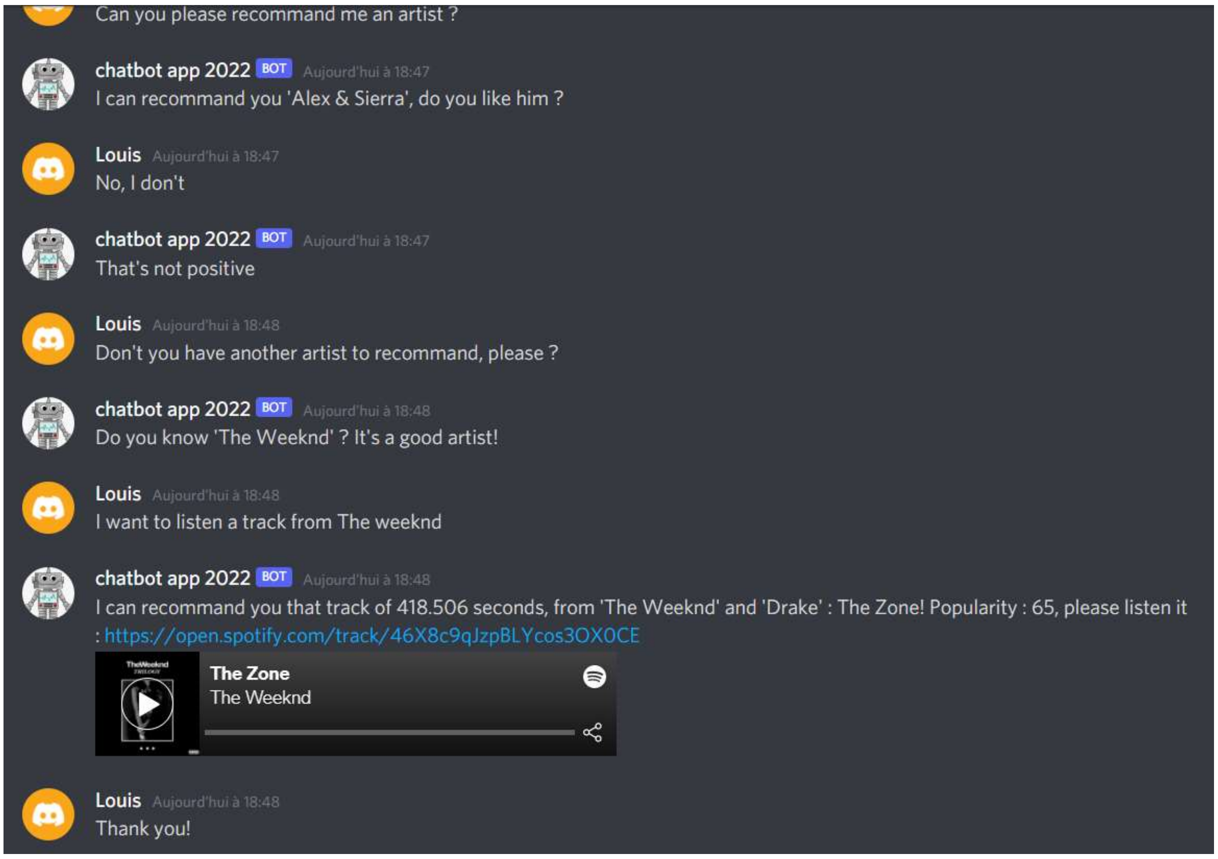
The chatbot will answer only if the message was sent by anyone else than himself, if the message was sent in *chatbot* channel, or if the message starts with "*$chatbot*" in another channel. If there is not any matched pattern, the bot will answer with default sentences such as *I don't know*, or *What do you mean by "…" ?* If the user message ends with a ?, we added some other default answers, such as *That's a really good question…*, to make it more natural.

After the program checked if the message was for the bot, it will try to match each programmed patterns of regex, and if there is a match, we have a set of *if* and *elif*, that works like a *switch*, for example:

```
    elif intent == 'from':
        artist = res.group('artist')
        ts = df[df['artists'].apply(
            lambda x: artist in x.lower())]
        if len(ts) > 0:
            _id = choice(list(ts.loc[:, 'id']))
            t = ts[ts['id'] == _id]
            artists = list(t['artists'])[0].replace(',', ' and ')
            track = list(t["track"])[0]
            popularity = list(t['popularity'])[0]
            time = list(t['time (s)'])[0]
            album = list(t['album'])[0]
            ans = [f'You can listen that song from {artists}, "{track}", from the album
{album}. The track lasts {t["time (s)"]} seconds, and has a score of {t["popularity"]} of
popularity. Listen it here : https://open.spotify.com/track/{_id}',
                   f'What about {track}, by {artists} ? A track that have {popularity} of
popularity, from {album}\'s album. https://open.spotify.com/track/{_id}',
                   f'I can recommand you that track of {time} seconds, from {artists} :
{track}! Popularity : {popularity}, please listen it :
https://open.spotify.com/track/{_id}']
        else:
            ans = ['I do not know that artist',
                   f'Who is {artist}?']
```

This is the result of a message such as "*Can I please listen a music from Selena Gomez?*". The program first get all the tracks in the *ts* dataset. If the length of ts is > 0, it means there are musics from that artist. We then get some information on the music, such as the id, the name of the artists in the music, the album, etc. And it send him back one of the sentence in *ans* array to give information on a music, and the link to listen the music.

Here is an example of discussion we can have with that bot



Can you please recommand me an artist ?

chatbot app 2022 BOT Aujourd'hui à 18:47
I can recommand you 'Alex & Sierra', do you like him ?

Louis Aujourd'hui à 18:47
No, I don't

chatbot app 2022 BOT Aujourd'hui à 18:47
That's not positive

Louis Aujourd'hui à 18:48
Don't you have another artist to recommand, please ?

chatbot app 2022 BOT Aujourd'hui à 18:48
Do you know 'The Weeknd' ? It's a good artist!

Louis Aujourd'hui à 18:48
I want to listen a track from The weeknd

chatbot app 2022 BOT Aujourd'hui à 18:48
I can recommand you that track of 418.506 seconds, from 'The Weeknd' and 'Drake' : The Zone! Popularity : 65, please listen it : https://open.spotify.com/track/46X8c9qJzpBLYcos3OX0CE

Louis Aujourd'hui à 18:48
Thank you!

You can see more in the video.