

1. Introduction.

My chat bot will talk to users about music and help them learn about songs and artists. I really like listening to music and using apps like Spotify and Genius to learn more about music and what the world thinks about certain songs. Spotify and Genius show data for songs like a song's genre, the song's mood, lyric annotations, and much more. I want to use info from their respective API's and datasets to make it easy for people to find out info about certain songs. Some interesting features I want for my chatbot is, based on inputs of songs and artists, to recommend songs for the user to listen to and open Spotify to play them. I want to implement my chat bot as a discord bot simply because I was curious about how it works, and based on tutorials I've researched I believe I will be capable of getting it running.

Part 3 Update: No changes

Part 4 Update: No changes

2. Technical Overview.

I will write my music chat bot using Python with a variety of python friendly API's and datasets for retrieving information, as well as the Discord API to make it function within the app. I'm considering using an online MySql database to work with a Spotify Dataset, but if I don't know if I can get that to work in time, so I'll most likely just keep the data as local Python files. Below is a list of API's and some of the features that they will be used for.

- **Spotify API and Dataset:** finding info about a song like Genre, BPM, Listens, creating recommended Playlists, and more obscure data like Acousticness, Danceability, Energy, Instrumentalness, etc
- **Genius API:** Facts about song lyrics and the artists (if available), including social media handles, real names, etc
- **Lyricsgenius:** Just lyrics for a song since the Genius API doesn't support getting the full lyrics
- **Discord API:** for getting the chat bot running

Part 3 Update: Turns out LyricsGenius is the python library for working with the Genius API so I can retrieve data easier, but i think i'll have to access the API manually for some requests (like getting a song's release date). I've still yet to set up the local database so i

have to research whether sqlite or pandas would be better to use. I'm a little behind schedule though in terms of questions because of trying to figure out what the libraries offered and what they didn't so i can get that data elsewhere

Part 4 update: I did not set up the local database server since i was able to find out how to get features of a certain audio track using the spotipy library like danceability, tempo, song length, and other data that removed the need for the database. I also read through the lyrics genius library more and was able to find out how to get data like the release date data that i thought was previously unreachable, so i didn't have to go through setting it up on my own.

How to run the discord chat bot:

- Pip install the libraries in the requirements.txt
- Create discord accounts at <https://discord.com/> and click the link below to get access to the private discord server for this project:
<https://discord.gg/UCw9mDgnaV>
- Run discord_chat.py to activate the bot and write responses within the text channels

3. Phases.

Phase 1 Set up [PART 3]:

- Download the necessary libraries to work with the above API's and also register any tokens/clients/authorization protocols
- Set up simple discord bot using the link below just with a simple hello and goodbye response system
<https://www.freecodecamp.org/news/create-a-discord-bot-with-python/>
- Learn about implementing a LL Parse
- Create code for making n-grams of input and check for existing artists, songs, albums. Also start researching words to include in the grammar

Phase 2 Experimentation [PART 3]:

- Build simple grammar and lexical rules to see what sentences can be created for parsing, starting with simple sentences like "What are the lyrics for [insert song names here]". Ideally I would get responses to work for questions about lyrics, an artist's currently most played songs, and when it was released with a limited grammar
- Learn how to get data from each of the respective API's to use for the chatbot. All of them have available python libraries that make getting data easier than having to do everything from scratch, but I have to figure out how I will process the local spotify dataset that's stored as a csv file. Based on what others said on Piazza i

might use pandas or sqlite, but that's something I will write in the part 3 report about what works in the end

Phase 3 Creating[PART 4]:

- Create a more flexible grammar that will allow the bot to comprehend getting the following data:
 - **Genius**: An interesting fact about a song's lyrics or the artist, release date, alternative names for artist, producer of song, featured artists
 - **Spotify**: Playing a song, Most played song, First song on spotify, length of a song, energy of the song, acousticness, song id, a song's genre, tempo
 - **LyricsGenius**: search for and get song lyrics
- Assuming I am on good pace with the previous phases there could be more questions I program for but I believe this is a solid, do-able list for this quarter

Part 3 Update: I decided to revert back to a CYK Parse because after researching LL Parse more it seemed like it wouldn't be very flexible with creating sentences easily, especially when I'll have to scale it up to accept many forms of questions. I will discuss this more in depth in the input handling section. In terms of retrieving data, the lyricsgenius and spotify libraries are working and the discord bot responds as it should.

Part 4: the final pieces of data i got from each library include:

- Spotify:
 - for artists: get Top song(s), get follower amount, get genres the artist plays under, get popularity of artist,
 - For songs: get danceability, energy, tempo, happiness, song length
- LyricsGenius: get lyrics, release date, and other artists that had sampled, interpolated (using someone's melody and re-record it instead of sampling it), covered, remixed a song

4. Examples

Person: Hello

Chatbot: Hi! What can I help you with?

Person: What is Drake's Best song

Chatbot: Best songs are subjective, but **Drake**'s most listened to song right now according to Spotify is "Laugh Now Cry Later". Do you want to listen to it?

Person: No

Chatbot: Understood. Would you like to know more about **Drake** or **Laugh Now Cry Later** or something else? (if response was Yes, then it would open spotify and play the song)

Person: Lyrics for One Dance by Drake

Chatbot: Here they are. [insert lyrics here]. Would you like to know more about **Drake** or **One Dance** or something else?

Person: What's interesting about One Dance

Chatbot: "One Dance" also features Nigerian singer Wizkid—he and Drake teamed up in 2015 to remix his track "Ojuelegba. Would you like to know more about **Drake** or **One Dance** or something else?

Person: When was One Dance released?

Chatbot: **One Dance** was released by **Drake** on April 5th, 2016. Would you like to know more about **Drake** or **One Dance** or something else?

Person: What was Drake's first song

Chatbot: **Drake**'s first official song was "Over". Would you like to listen to it on Spotify?

Person: Yea

Chatbot: [Spotify opens and plays it] Understood. Would you like to know more about **Drake** or **One Dance** or something else?

Person: No

Chatbot: End using the chatbot?

Person: Yes

Chatbot: Goodbye!

Part 3 Update: No changes

Part 4: not including an artist's first song, and playing spotify songs from discord's builtin spotify player instead of the spotify app. Also ending the bot conversation is done just by sending a "Goodbye!" message

5. Input Handling.

My current plans are to implement an LL Parsing algorithm over CYK Parse for analyzing input. I do not believe it's too necessary for calculating the probability of each grammar since it's more important to find what the keywords are and determine what the user will ask. Before it does any of that, we'll have to create n grams of the input to determine what songs and/or artists the user might be talking about. We also have to be wary of punctuation and make copies with and without it. For example, the sentence "What is Drake's best song" requires us to make the n-gram list:

["What is Drake's best song", "What is Drake's best", "What is Drake's", "What is Drake", "What is", "is Drake's best song", "is Drake's best", "is Drake's", "is Drake", "Drake's best song", "Drake's best", "Drake's", "Drake", "best song", "best", "song"]

leaving out stop words like "what" and "is", and over generating strings that end in "s" for all possible inputs. This is to account for the fact it's more likely for users to write something like "what is Drake's best song" over "what is Drake best song" because it makes more sense grammatically. Once it creates this list, it looks for an exact match in the spotify data and if it finds an artist, song, or album then it stores it in a dictionary with key value pairs of strings to lists. This is in case there are multiple choices to consider for each song, artist, or album. It chooses the longest string from the list with an exact match(es) and then adds itself to the lexical rules to look for during the LL Parse. During the parse, the question words will help determine what the user wants. If the phrase starts with "How", it's most likely a question about how long a song or album would be. "When" would entail a question about when a song or album was released. "What" will entail many possibilities like getting song lyrics, song statistics, and an artist's most popular song.

Part 3 Update: Since I've switched over to using CYK Parsing again it's made handling easier. I was worried about what words to consider keywords and how that could affect retrieving music information. I didn't want to send a ton of api requests for a bunch of slightly different phrases, so i pulled words relevant to the existing lexicon to help. When a word not found in the lexicon is seen, it's substituted with a "MusicNoun" string and we start compiling the rest of the

MusicNoun until we see another keyword or the input ends. Right now it only works for one MusicNoun but i will try to store multiple music Nouns in the next phase. An example of this in action can be shown with the input: “what are Drake’s most played songs.”

- It matches “what” to its grammar rule, then “are”, then it sees “drake’s”. This word isn’t in the lexicon so we replace it with the placeholder “[MusicNoun]” value in our list of words to be parsed instead of “drake’s”. “Drake’s” meanwhile gets stored in a MusicNoun variable. The next word “most” is in our lexicon so we say “Drake’s is our MusicNoun” and add the rest of the words to sentence to be parsed, with a final input sentence to parse of [“what”, “are”, “[MusicNoun]”, “most”, “played”, “song”].

Part 4 update: As I suspected I had to mess around with the keywords, like using the last “**by**” in my input to separate a song from its artist and collecting all available **Music Nouns** in a **list**. I also checked for the first time the word “**for**” appears to help determine when the user will talk about music. An example of it in action would be “what are the lyrics for Drowning by Jai Wolf” getting parsed like above to make the list [“Drowning”, “Jai Wolf”] to be used when we look for data through the API’s. We see the word for so we start checking the next words as music nouns, and when we’ve found the last by, we separate those strings. This however causes a problem if the user inputs solely a song with no artist, because if it contains the word **by**, then it will continue to separate the words. An example of this would be for the song “hand crushed by a mallet”, where the music nouns equals [“hand crushed”, “mallet”], removing the “a” because of my stop word processing. This means it will try to search for a song “hand crushed mallet”, as the music nouns list is joined into a string before it’s used with the API’s. If you add on “ by 100 geecs”, it parses correctly and music nouns become [“hand crushed by a mallet”, “100 geecs”].

To end the chatbot, simply say bye, goodbye, or farewell. Case doesn’t matter since it all gets put into lowercase in the end. Example sentences on using the chatbot are commented within the main function of the Create_Message.py file.

I added a form of memory retention for when a user doesn't appear to specify a MusicNoun, and uses that data in its place. I did this by implementing a simple list that stores the previously used data. Here are the functions that can work alongside each other:

- Given an Artist:
 - i. What is their most listened to song(s)
 - ii. What is their popularity
 - iii. What is their follower count
- Given a Song [and Artist]:
 - i. What is the [energy, length, tempo, happiness, danceability]
 - ii. What are the lyrics

- iii. Who did a remix
- iv. Who did [covers, interpolations] for the song
- v. Who has sampled the song

6. Internal Representations and Data Sources.

The grammar would for the most part be hard coded to fit possible inputs from the user. The spotify dataset will be the only database I work with which will be used to find potential music data to add lexical rules during run time. I'm considering using sqlite or pandas for reading the data from the local csv files. This stores Spotify info on songs from 1920 to January 2021 with statistical information like energy, acousticness, and more. The API's will be used for retrieving information like lyrics, an artist's most listened to songs, facts not available in the database. After each input it will keep track of the song, artist, or album being talked about to help with next potential questions during use.

Part 3 Update: I have still yet to figure out how to retrieve the local spotify data. It's most likely too big to be stored as a pickle file, so i must decide on whether to use sqlite or pandas. Pandas appears less complex than sqlite but i haven't dived too deep into example code

Part 4 update: I was able to get the data i was interested in solely through more documentation reading and did not need to create a local database.

7. Output handling.

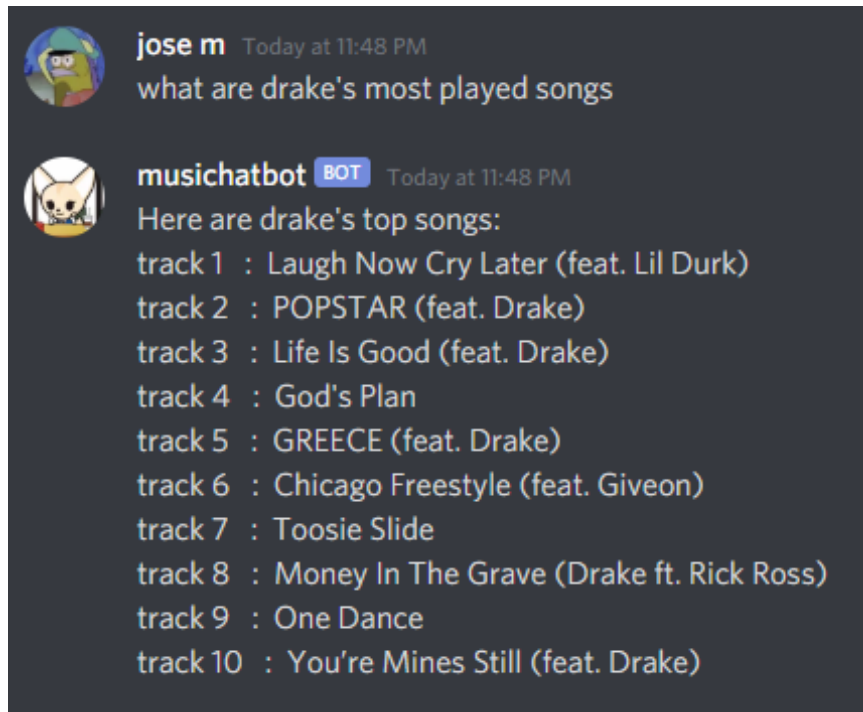
Below are examples of how the chatbot should respond to a given sentence. Not all potential responses are listed but this will be the general form of it:

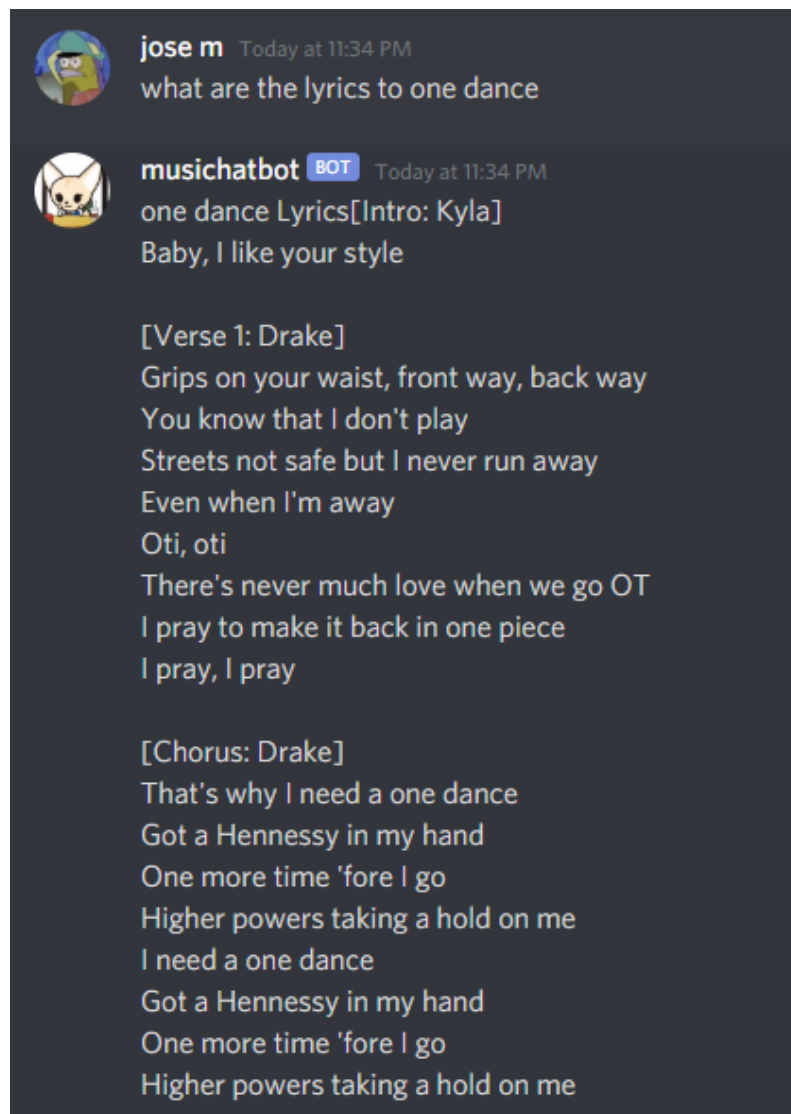
- If the user Greets the chat bot, the chat bot will say hi back
- If the user asks "What is [artist]'s best song", the chatbot will output "Best songs are subjective, but [artist]'s most listened to song right now is [most_listened_to_song]. Do you want to listen to it?"
 - If "yes" open Spotify
- If the user asks "What are the lyrics for [song] by [artist]", the chatbot will reply "Here they are [song lyrics]"
- If the user asks "What's interesting about [song]", the chatbot will reply with a random fact about the song from Genius, if it's available
- If the user asks "When was [song] released", the chatbot will reply "[song] was released on [release date]"
- If the user asks "When was [song] released", the chatbot will reply "[song] was released on [release date]"

- At the end of each chatbot response the chat bot will ask if it wants to learn more about what the user inputted as “Would you like to know more about [current_artist] or [current_song]?”
 - If no, then it will ask “Do you want to end using the chat bot?”. If the user enters “yes” then the chatbot says goodbye

Part 3 Update: In my response module I make sure to keep track of key phrases like lyrics, songs, and popular to help my chatbot decide on how to respond. some new questions i want to add include:

- what are [genre]’s top songs
- What are my most listened to songs right now (based on user spotify authentication)





This continues for the rest of the song

Part 4 Update: the main edit i added was just giving users the spotify link if the question the user asked used spotify data. Asking whether they wanted to see it or not was going to mess with the way I parsed responses so adding it to the bot reply seemed like the best option. I didn't have time to implement the "What's interesting about the song" feature but the new ones i mentioned in the input handling work. Also, instead of asking if a user wants to end the chat bot, they just need to write bye, goodbye, or farewell and the bot will stop running.



jose m Today at 10:06 PM

what is the most played song by dua lipa



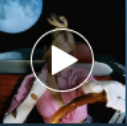
musichatbot BOT Today at 10:06 PM

Here is Dua Lipa's top song:


track 1 : Levitating (feat. DaBaby)





Here's a spotify link with music you might like <https://open.spotify.com/artist/6M2wZ9GZgrQXHCfjv46we>

Would you like to know more about Dua Lipa?



Top tracks for Dua Lipa



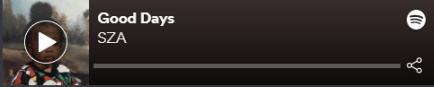


| | | |
|---|--------------------------------|------|
| 1 | Levitating (feat. DaBaby) | 3:23 |
| 2 | We're Good | 2:45 |
| 3 | UN DIA (ONE DAY) (Feat. Tainy) | 3:52 |
| 4 | Prisoner (feat. Dua Lipa) | 2:49 |
| 5 | Don't Start Now | 3:03 |
| 6 | Break My Heart | 3:41 |
| 7 | One Kiss (with Dua Lipa) | 3:34 |

More examples of responses below:

jose m Today at 10:00 AM
what is the tempo of good days by sza

musichatbot BOT Today at 10:00 AM
Good Days by SZA has a tempo of 121.002.
Here's a spotify link with music you might like <https://open.spotify.com/track/3YJJQPAbDT7mGpX3WtQ9A>
Would you like to know more about Good Days or SZA?

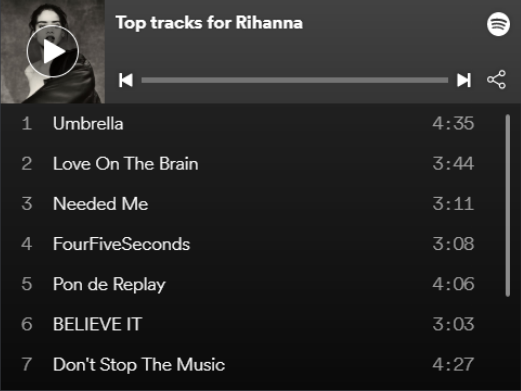


jose m Today at 10:00 AM
what are the lyrics

musichatbot BOT Today at 10:00 AM
Okay here are the lyrics for Good Days by SZA
[Verse 1: SZA]
Good day in my mind, safe to take a step out
Get some air now, let yo edge out
Too soon, I spoke
You be heavy in my mind, can you get the heck out?
I need rest now, got me bummed out
You so, you so, you
Baby, baby, babe

jose m Today at 10:01 AM
what is the follower count for rihanna

musichatbot BOT Today at 10:01 AM
Rihanna has 41713833 many followers
Here's a spotify link with music you might like <https://open.spotify.com/artist/5>
Would you like to know more about Rihanna?



| Track | Duration |
|------------------------|----------|
| 1 Umbrella | 4:35 |
| 2 Love On The Brain | 3:44 |
| 3 Needed Me | 3:11 |
| 4 FourFiveSeconds | 3:08 |
| 5 Pon de Replay | 4:06 |
| 6 BELIEVE IT | 3:03 |
| 7 Don't Stop The Music | 4:27 |

jose m Today at 10:01 AM
what are her top songs

musichatbot BOT Today at 10:02 AM
Here are Rihanna's top songs:
track 1 : Umbrella
track 2 : Love On The Brain
track 3 : Needed Me
track 4 : FourFiveSeconds
track 5 : Pon de Replay
track 6 : BELIEVE IT
track 7 : Don't Stop The Music
track 8 : Disturbia
track 9 : Desperado
track 10 : Work
Here's a spotify link with music you might like <https://open.spotify.com/artist/5>

And as some significant coding accomplishments, here is my reworked parsing algorithm that helped all this be possible.

```
364 user_input = "What are the lyrics for and july by heize"
365
366 cykp, musicnouns = compile_msg(user_input)
367 # getSentenceParse(t["CYK_PARSE"][0])
368 # print(cykp[0])
369 print(musicnouns)
370
371
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

2: Python

```

ree object at 0x01706408', 'S/0/5': <Tree.Tree object at 0x01706508>, 'VP/1/6': <Tree.Tree object at 0x01706568>, 'NP/2/7': <Tree.Tree object at 0x01706538>, 'S/0/6': <Tree.Tree object at 0x01706598>, 'VP/1/7': <Tree.Tree object at 0x017065C8>, 'S/0/7': <Tree.Tree object at 0x01706628>] ['and july', 'heize']
PS C:\Users\josed & C:\Users\josed\AppData\Local\Programs\Python\Python38-32\python.exe "c:\Users\josed\OneDrive\Desktop\uci master\sum 20 - spr 22\wint 21\CS 17 1/project/chat bot - musicchat\Parsing.py"
T: [S[Question what], 'Verb/1/1: [Verb are]', 'Article/2/2: [Article the]', 'Noun/3/3: [Noun lyrics]', 'Prep/4/4: [Prep for]', 'MusicNoun/5/5: [MusicNoun [musicNoun]]', 'Prep/6/6: [Prep by]', 'MusicNoun/7/7: [MusicNoun [musicNoun]]', 'S/0/1: [S[Question what] [Verb are]]', 'VP/1/2: [VP[Verb are] [Article the]', 'NP/2/3: [NP[Article the] [Noun lyrics]', 'PP/4/5: [PP[Prep for] [MusicNoun [musicNoun]]]', 'PP/6/7: [PP[Prep by] [MusicNoun [musicNoun]]]', 'S/0/2: [S[Question what] [VP[Verb are] [Article the]]]', 'VP/1/3: [VP[Verb are] [NP[Article the] [Noun lyrics]]]', 'NP/2/4: [NP[NP[Article the] [Noun lyrics]] [Prep for]]', 'S/0/3: [S[Question what] [VP[Verb are] [NP[Article the] [Noun lyrics]]]', 'VP/1/4: [VP[Verb are] [NP[NP[Article the] [Noun lyrics]] [Prep for]]]', 'NP/2/5: [NP[NP[Article the] [Noun lyrics]] [PP[Prep for] [MusicNoun [musicNoun]]]', 'S/0/4: [S[Question what] [VP[Verb are] [NP[NP[Article the] [Noun lyrics]] [Prep for]]]', 'VP/1/5: [VP[VP[Verb are] [NP[NP[Article the] [Noun lyrics]] [Prep for]]] [MusicNoun [musicNoun]]]', 'NP/2/6: [NP[NP[NP[Article the] [Noun lyrics]] [PP[Prep for] [MusicNoun [musicNoun]]] [Prep by]]]', 'S/0/5: [S[Question what] [VP[VP[Verb are] [NP[NP[Article the] [Noun lyrics]] [Prep for]]] [MusicNoun [musicNoun]]]', 'VP/1/6: [VP[Verb are] [NP[NP[NP[Article the] [Noun lyrics]] [PP[Prep for] [MusicNoun [musicNoun]]] [Prep by]]]', 'S/0/6: [S[Question what] [VP[Verb are] [NP[NP[NP[Article the] [Noun lyrics]] [PP[Prep for] [MusicNoun [musicNoun]]] [Prep by]]]', 'VP/1/7: [VP[VP[Verb are] [NP[NP[NP[Article the] [Noun lyrics]] [PP[Prep for] [MusicNoun [musicNoun]]] [Prep by]]] [MusicNoun [musicNoun]]]', 'S/0/7: [S[S[Question what] [VP[VP[Verb are] [NP[NP[NP[Article the] [Noun lyrics]] [Prep for]]] [MusicNoun [musicNoun]]] [PP[Prep by] [MusicNoun [musicNoun]]]]]']
['and july', 'heize']

```

In this photo we see the input “what are the lyrics for and july by heize” successfully finding the music data we want and creating the music nouns list equaling [“and july”, “heize”]. The spots where they used to be have also been replaced with the word “[MusicNoun]” that denotes it’s a placeholder for data.