­­ TP3

**Changes Implemented**

1. Created new infinity room of disliked songs, scrolling function set
2. Edited UI
3. Added playlist generation function from liked songs

TP2

**Changes Implemented**

1. Design of my game has slightly changed – from the loading screen you now walk into a main room, where you can access different rooms. There is the song recommendation room, the personality results room, and the liked music room.

**Functionalities of each room**

1. Song recommendation room
   1. Recommendations are affected by both the personality you are determined to have and the top 40 tracks you listened to.
   2. Pressing z will refresh the recommendations
   3. Moving the character over the song cover and pressing x will dislike the recommendation. This means that the song will never be displayed in the recommendations list every again.
   4. Pressing space when your character is over the song character will start playing it on your Spotify player. Your player has to be active and on, I used try/except to make sure that an error prompt is printed out instead of the whole program crashing.
   5. Moving the character over the song cover and pressing y will like the recommendation. This means that the song will be displayed in the liked songs music wall.
   6. Likes and dislikes are saved despite the user closing or starting the app again.
2. Liked music room
   1. Pressing M will refresh the wall in case the user liked songs in the current session and wants to see them pop up.
   2. The music room has screens that can go on infinitely, press A to toggle backwards and D to toggle forward between the screens. Press space when your character is on any song cover to start playing it on your Spotify player.
   3. Likes are stored in a file automatically created in the same directory.
3. Personality results room
   1. The results of your personality test are generated, along with a quick snippet of what that personality means.
   2. Pressing space will restart the process with the next closest similarity.

**Algorithms**

1. To match personality
   1. Variances of the different features of user’s top 40 songs versus distribution across the different songs of every playlist is compared, lowest variance wins
2. To match song recommendations
   1. Dataset of songs pulled out depending on random few tracks in the user’s personality playlist. They are then arranged and filtered via the user average feature scores via the lowest Euclidean distance and displayed, then clears out possible overlaps with the disliked songs from the user, also filter outs duplicates.

TP1

**Name:** psst. hey. let me guess.

**Description:** Guessing your MBTI personality based on what you listen to on Spotify, then exploring the results and your music taste via a side-scroller game.

**Competitive Analysis:**

Similar Projects I've seen online:

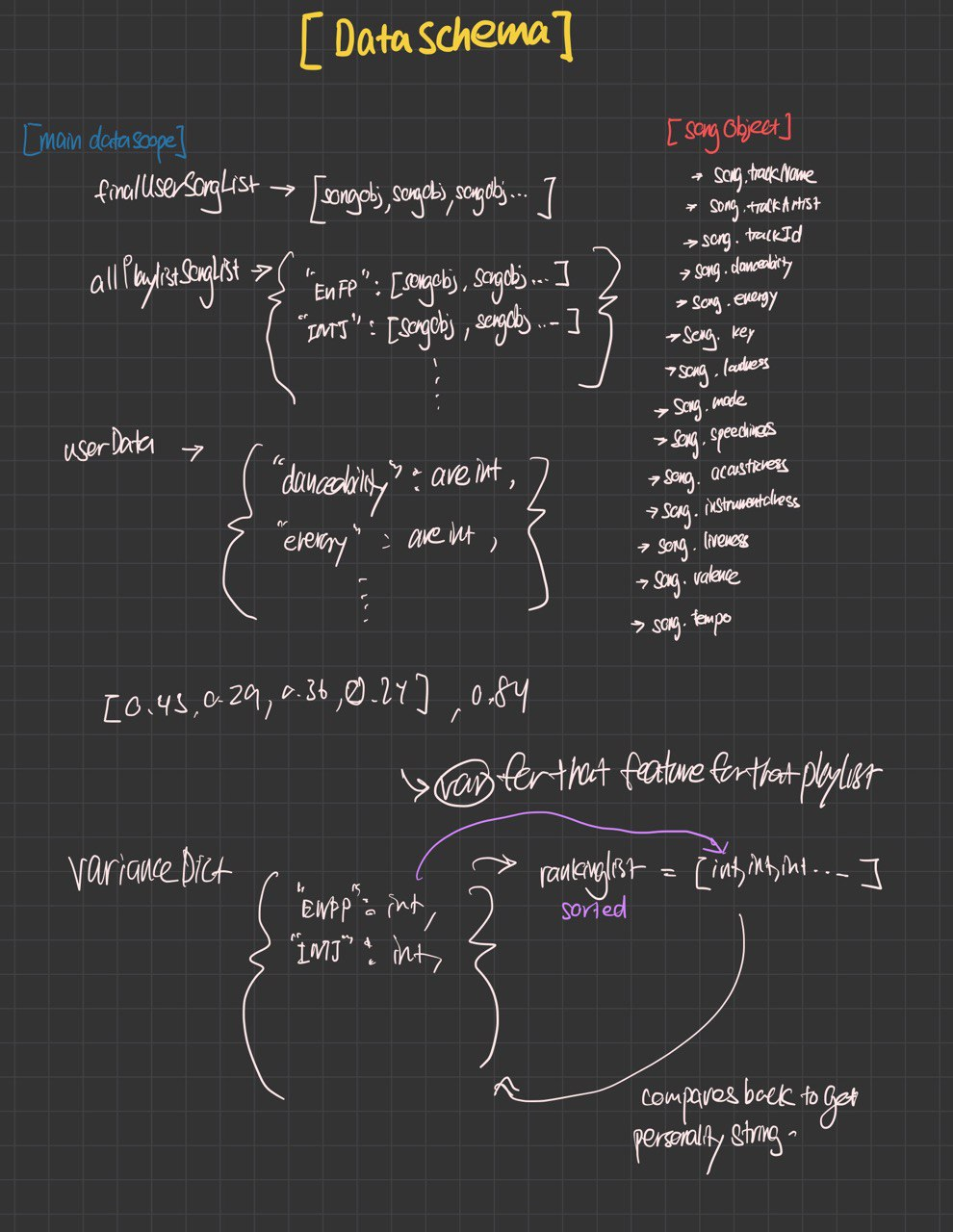
1) There are a bunch of Spotify API based apps which depend on your Spotify data to come to some sort of generalization about your music taste, for example, this nicely done one called How Bad is Your Spotify. https://pudding.cool/2020/12/judge-my-spotify/ These apps use the API provided by Spotify and attempt to come to some sort of conclusion about your music taste.

2)I've also seen something like a project trying to compare whether two playlists are similar, like this one. https://medium.com/@yurs/how-similar-my-boyfriend-top-tracks-on-spotify-compare-to-mine-bf754622d4d6 However, this seems to be more data analysis focused, and she seemed to just have used the basic functions of MATLAB to visualize the different data, but didn't come to a complete conclusion about it.

My project is different as firstly, it compares similarities in playlists more accurately. I compared the variance of each feature individually across the distribution of each playlists' songs to the mean of each feature for the user's top 40 listened tracks for two years. This gave me a more accurate similarity comparison, which I then used to determine which personality was the more accurate one for the user. I also haven't seen this specific variant of a personality quiz using your Spotify data. I mean this form of testing for MBTI personalities are definitely not an accurate measure by any means, but it is fun!

Next, the medium in which the results are shown in my project is also different and quite fun to play around in. I also am able to control the user’s active player.

**Structural Plan:** The project will be organized according to different modes -> there will be landing page, loading page, results, and different modes for the game itself. The main calculations are done in the large backend function, and my data is sorted and parsed into different dataclasses that I require. The dataclass dictionary is shown below.



**Algorithmic Plan:** I believe the trickiest part of my project algorithmically will be the calculations of the similarity between the user’s top songs and each playlist to determine which personality the user is at. What I did was to take the average of every feature Spotify provides for a song (such as danceability, tempo, valence etc) for the top 40 most listened songs of a user over a year. Then, I took those average values, and took the variance of that value against the distribution of values for each feature on every song for each playlist, making sure to standardize the values (in a trial-and-error way) as each feature had a different range of values possible. Lastly, I added all the variances for each feature up for each playlist, then took the one with the smallest variance possible, before storing them all in relevant data structures to be stored in my animation project.

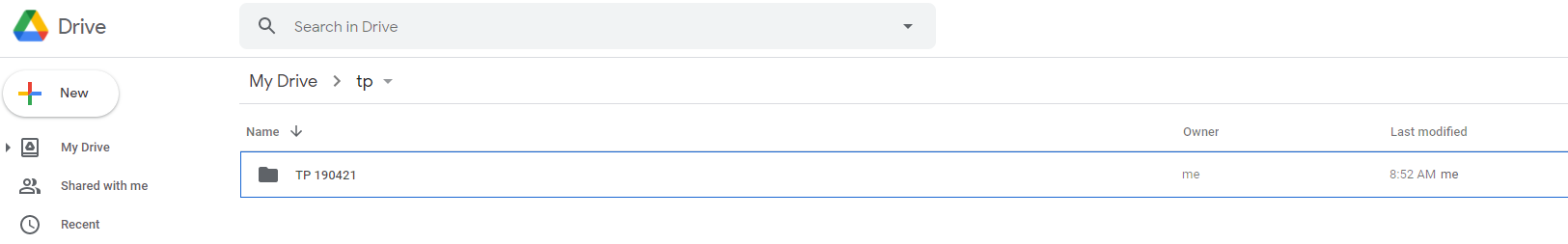
**Timeline Plan:**

26th April 2021: Finish coding backend and setting up the necessary structure and datasets.

28th April 2021: Finish the rough skeleton of each slides.

1st May 2021: Finish aesthetics and polishing.

**Version Control Plan:** My whole project will be backed up after every major change in Google Drive in different versions, as exemplified by my first version below.



**Module List:** Spotipy’s API (tech demo approved)