Zain Rashid – PSYZR2 – 20172929 – WORD COUNT (2000)

VOICEFLOW PROTOTYPE REPORT

School of Computer Science
COMP3074

The Prototype

I have designed a voice bot which allows for the creation and management of multiple playlists. The songs in a playlist represent items, where each song is selected through the user specifying multiple details, such as the artist and album associated with it. The user can select a track from an album or add the full album to their playlist if preferred. Alternate paths can be taken where a user can view the top artists at any given time or get the top tracks of a selected artist.

The user can query about an artist or track if they want more information. They can receive an artist's biography or a random fact about them. While if inquiring about a track they can know if it's explicit, get it's runtime and if available, listen to a mp3 preview link.

The playlist is a collection where songs are added or removed, playlists can be deleted and have their name changed. While updating them, the user can view the playlist's content as it changes. This can be done before it's saved to the database and after it's been stored. When viewing the playlist after storage, the user can narrow the query down, allowing for searches of specific artists or albums in the playlist, or a view of the full playlist.

Each track added to a playlist costs £1, where the cost of the playlist changes is tracked throughout the updating of it. When tracks are removed, the song cost is deducted, this allows for the user to be refunded in certain scenarios. If there is no net change in playlist cost, then the user may not incur any transaction refund/cost. If the user is ready to exit, they can save their changes or discard them. Saving will incur a transaction process where the playlist and it's cost will be saved to the Google Sheets database. While discarding will remove all session alterations.

The user cans call on a global cancel intent, allowing them to leave any process, and return to the start or exit completely.

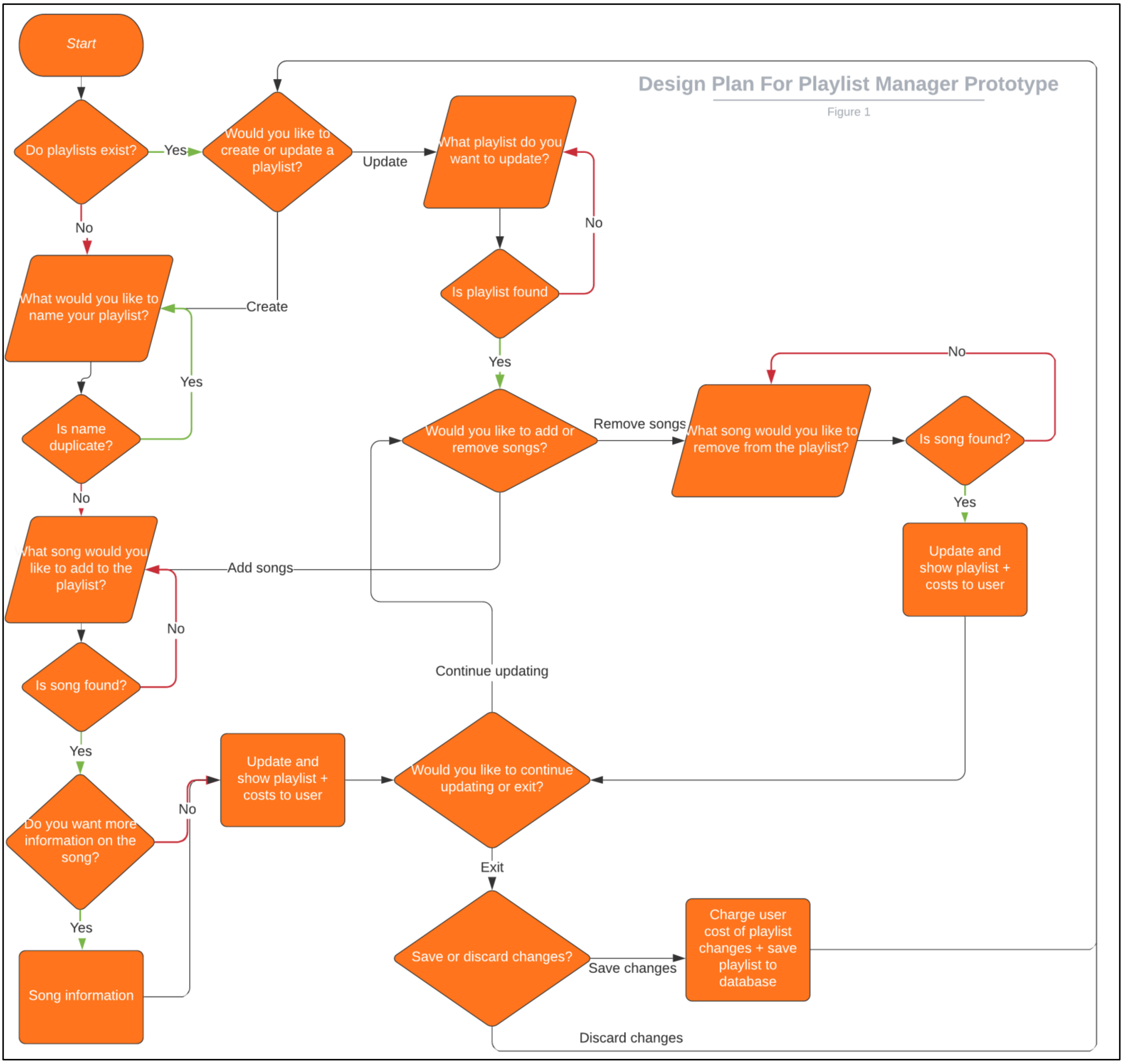
Design Process

I started the design process with a flow[1] to create a clear structure for my prototype. At this stage, the linkage of components hadn't been realised, meaning interactions were simplified. The manor in which I retrieve a track also differed, where instead of gaining information about a song through it's artist and album, I only ask them to give the requested song. In production this method turned out to be unfeasible due to the limitations of the API. Details like depth and personality hadn't yet been thought out.

Next, I created sample dialogs to visualise typical bot conversations. I created a "good[2]" and "mixed[3]" path, the good path entails a flawless interaction, with focus on prompt design and discoverability, in addition to usage of implicit and explicit confirmation. While the mixed path focused on misinputs and the handling of them with re-prompts and alternate pathing. The dialogs let me focus on the user, planning the use of key design principles.

Once I had implemented the API and created a fundamental prototype, I thought how to add depth to the manager, I created a flow[4] which outlined a data interaction process for a returning user. Here, I focused on the key design principles instead of functionality, I made sure every interaction was catered to the user, so the experience felt personalised and not linear.

Zain Rashid – PSYZR2 – 20172929 – WORD COUNT (2000)


Design Plan For Playlist Manager Prototype
Figure 1
Start
Do playlists exist?
Yes
No
Would you like to create or update a playlist?
Update
Create
What playlist do you want to update?
Is playlist found
No
Yes
What would you like to name your playlist?
Is name duplicate?
Yes
No
Would you like to add or remove songs?
Remove songs
Add songs
What song would you like to remove from the playlist?
Is song found?
No
Yes
Update and show playlist + costs to user
What song would you like to add to the playlist?
Is song found?
No
Yes
Do you want more information on the song?
No
Yes
Song information
Update and show playlist + costs to user
Would you like to continue updating or exit?
Continue updating
Exit
Save or discard changes?
Save changes
Discard changes
Charge user cost of playlist changes + save playlist to database

Bot: Hi! What is your name?
User: Call me Zain.
Bot: Zain, you can either create, update, or inspect a playlist. What would you like to do on this fine day?
User: Can I create a playlist.
Bot: What would you like to call your playlist?
User: Call it “my playlist”.
Bot: Great! The playlist “my playlist” has been successfully created.
What artist would you like to add to “my playlist”?
User: Add Drake please.
Bot: Nice! The artist was found.
Would you like to get a Drake album or retrieve his top tracks?
User: Please may I get his top tracks.
Bot: Here’s the list of top tracks by Drake: One Dance, God’s……. What track would you like to add to “my playlist”?
User: Let me add One Dance.
Bot: Would you like to get more info on One Dance?
User: Yes.
Bot: One Dance is a non-explicit song with a playback of 180 seconds, check your device to listen to a preview of the track!
The song has been added to your playlist and will cost you £1.
Would you like to view your newly updated playlist?
User: Yes please.
Bot: “my playlist” contains: One Dance by Drake and has a value of £1.
Would you like to continue updating your playlist or exit?
User: Please may I exit.
Bot: Okay, would you like to save the changes or discard them?
User: Save the changes thanks.
Bot: The playlist will cost you a total of £1 and will contain: One Dance by Drake. Are you sure you want to confirm this transaction?
User: I am, yes.
Bot: The transaction has been completed and “my playlist” has been saved!

Figure 2: Happy path, sample dialog

Bot: Hi! What is your name?
User:
Bot: Please can you give a valid name.
What is your name?
User: Zain
Bot: Zain, you can either create, update, or inspect a playlist. What would you like to do on this fine day?
User: Cna craeet a listplay.
Bot: Sorry, I didn't understand you, can you repeat yourself.
User: Check the weather.
Bot: Please choose one of the three provided options.
User: No, I will not.
Bot: Say either "create", "update" or "inspect".
User: update
Bot: The playlists you can choose include: playlist1, playlist2.... What playlist would you like to update?
User: "playlist1"
Bot: Great! The playlist "playlist1" has been successfully selected.
Would you view the songs in "playlist1"?
User: Yes.
Bot: The playlist "playlist1" contains: Blinding Lights by The Weeknd, Man on the Moon by Kid Cudi. It is worth £2.
Would you like to add or remove songs from "playlist1"?
User: Remove songs please.
Bot: The songs you can remove are: Blinding Lights by The Weeknd, Man on the Moon by Kid Cudi.
What song would you like to remove?
User: Blinding Lights by The Weeknd.
Bot: The track "Blinding Lights by The Weeknd" has been removed. The changes to your playlist currently will refund you £1.
Would you like to continue updating your playlist or exit?
User: Let me continue updating.
Bot: Would you like to add or remove songs?
User: Add.
What artist would you like to add to "my playlist"?
User: I do not know.
Bot: Please could you enter a valid artist.
To help you out here are the current trending artists: Drake, Mariah Carey, Ed Sheeran...
What artist would you like to add?
User: Ed Sheeran.
Bot: The artist was found! Would you like to get Ed's albums or retrieve their top tracks?
User: album please.
Bot: What album would you like to get?
User: Sorry, I'm stuck.
Bot: To help you out here's the list of top tracks by Drake: One Dance, God's....... What track would you like to add to "playlist1"?
User: Let me add God's Plan.
Bot: Would you like to get more info on God's Plan?
User: Yes.
Bot: God's Plan is a non-explicit song with a playback of 210 seconds, unfortunately there is no preview link available!
The song has been added to your playlist and will cost you nothing.
Would you like to view your newly updated playlist?
User: No.
Bot: Would you like to continue updating your playlist or exit?
User: Please may I exit.
Bot: Okay, would you like to save the changes or discard them?
User: Save the changes thanks.
Bot: The playlist will cost you nothing and will contain: Man on the moon by Kid Cudi, God's Plan by Drake. Are you sure you want to save the playlist?
User: I am, yes.
Bot: "playlist1" has been saved!

Figure 3: Following mixed path, sample dialog

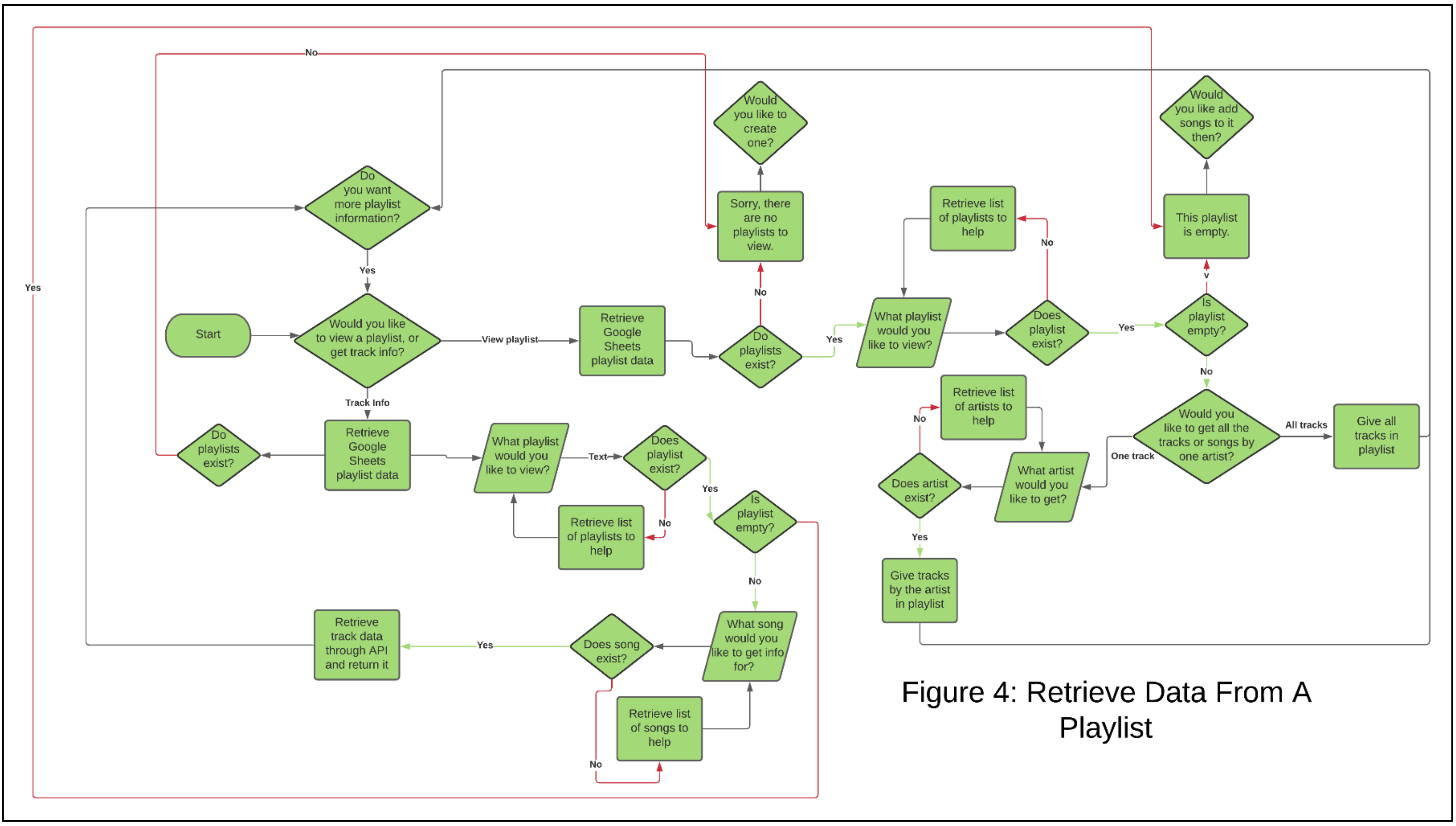

Start
Would you like to view a playlist, or get track info?
View playlist
Retrieve Google Sheets playlist data
Do playlists exist?
No
Sorry, there are no playlists to view.
Would you like to create one?
Yes
What playlist would you like to view?
Does playlist exist?
No
Retrieve list of playlists to help
Yes
Is playlist empty?
This playlist is empty.
Would you like add songs to it then?
No
Would you like to get all the tracks or songs by one artist?
All tracks
Give all tracks in playlist
One track
What artist would you like to get?
Does artist exist?
No
Retrieve list of artists to help
Yes
Give tracks by the artist in playlist
Do you want more playlist information?
Yes
Track Info
Retrieve Google Sheets playlist data
Do playlists exist?
What playlist would you like to view?
Text
Does playlist exist?
No
Retrieve list of playlists to help
Yes
Is playlist empty?
No
What song would you like to get info for?
Does song exist?
No
Retrieve list of songs to help
Yes
Retrieve track data through API and return it
Yes
No

Figure 4: Retrieve Data From A Playlist

VUI Design Principles

Prompt Design & Discoverability

With my prompts I narrowed down the types of questions given. Instead of giving open-ended questions such as “what do you want to do?” that can have an interminable amount of choices; I focus on close-ended questions like “would you like to save or discard changes?”, giving the user two choices, where each choice is unique and distinctive. This lets the user know all the possible paths they can take; in addition to giving greater context clarity and providing discoverability so the user won’t forget or have to wonder what choices they have. The distinctive phrasing is also utilised in calling the global intents.

Secondly, I controlled the placement of questions in a prompt. Instead of asking the question first then supplying options, I provide the options then the question. This makes sure the question is still present in the user’s short-term memory when they try to give a response. Else the question may be forgotten after all options have been listed.

Next, I limited the amount of options given to three. If too many choices are provided the user can lose track of them. A limit lets the user more easily recall choices and make a more informed decision. However, due to the usage of lists, certain prompts require more than three options, this can be justified due to how lists in Voiceflow operate, as they can only be utilised within a custom code block (JavaScript). In addition, due to the nature of the manager there’s a non-finite amount of data that can be gotten from the API, so the data received shouldn’t be limited, as this could hamper the user experience.

Finally, I interweave user responses into follow-up prompts. An example would be “would you like to get (artist’s name)’s album or their top tracks?”. Here I am echoing their previous choice of selected artist, so they know the exact context of the question and are refreshed on the previous selection they made.

Error Handling

Firstly, I employ ‘advanced fallbacks’ where given an unknown input, the user is given multiple re-prompts and if those fail, they are rerouted back to the original prompt till a valid input is supplied. There are three tiers to the re-prompts:

- The first implies that the user made an unintentional input error, such as a mispronunciation. A generic “didn’t understand you, try again” prompt is given.
- The second tier gives more of a focused response, and it now can be interpreted that they don’t fully understand the prompt. A re-prompt such as “please say one of the two options provided” is given.
- The final re-prompt is the bluntest, where the user is told exactly what to say to trigger a choice. An example would be “Say either ‘create’, ‘update’ or ‘info’.”

Secondly, when the user tries to enter or retrieve data for an artist, album, or playlist name; errors are handled differently. Now capture blocks receive the input, where it is pre-processed, removing punctuation, whitespaces, and more. The inputs need to be compared to real data for validity, meaning fetching data from either Google Sheets or the API to authenticate a user’s response. If the response is invalid due to it being a duplicate, unknown input or having another issue, the user is prompted to enter a new input. For certain inputs, more advanced redirects can give the user an alternative route to help. For example, when an artist enters an unknown artist, they are able to get the top artists at the moment to help their decision.

Personalisation

The user can assign themselves a username, which is frequently referred to. For certain prompts there can be randomly chosen alternatives and when creating a playlist they are able to uniquely name it. Furthermore, the inherent design of a playlist manager acts as a personalised experience, as each song/ playlist represents the user’s own taste. Lastly, conditional branching prevents the user from being constrained down a linear path. This allows each interaction / path to be based on the user’s personal choice, not the designers.

Confirmation

When the user has replied to a prompt which required the input of data such as an artist’s name, implicit confirmation is used, this entails the use of statements to confirm the user’s choice, in this example, confirming what artist they have selected. This is effective as since the user attention span is short, they need a summary of how much the system knows. While, when a user is making a vital decision, like exiting a process or solidifying a transaction, explicit confirmation is required. This is shown in the form of a prompt such as “Are you sure? (yes/no)”, which is given to confirm the user’s intent.

Context

Context tracking is performed through the use of components like variables, booleans, and conditional branching. For example, if a playlist is empty a user isn’t prompted to view their playlist, or depending on the changes made in a playlist, a user may either pay a fee, pay nothing, or get a refund. These are a few examples of context tracking in action.

Global intents have different paths based on where they're called in the lifecycle. For example, if the user tries to exit while updating a playlist, they are prompted to save or discard changes. They then are allowed to either exit completely or return to the start. However, if they try to exit at the start, the user is only allowed to leave the bot, as they can't return to where they already are.

Integration

The bot utilises Napster's API which allows for the retrieval of music data. In this project I retrieve artist, album, and track data in addition to global charts such as the top artists in the world. Each data type has subcategories allowing for further information to be provided to the user, if need be. This includes details such as an artist's biography or how long a selected track is.

The API data is retrieved through JSON requests, where I save the desired data to Voiceflow variables and lists containing subcategories. These lists are updated and interacted with through the custom code feature (JavaScript). This allows for dynamic data storage where the amount of data being received is unknown.

The playlists are saved to a Google Sheet Database where they are sorted by their uniquely assigned names. Each playlist contains lists of songs, artists and albums and the total cost of the playlist, these are the only necessary details required as all other data can be retrieved via the API at will. Whenever, a user selects a playlist the data is imported and stored to corresponding lists until they're finished updating or utilising a playlist.

Conclusion

Overall, I believe I made a solid prototype which tries to minimise the downfalls of a voice bot and add as much depth and personality as it can. I am proud of my API implementation in this project, it allowed me to create a unique premise where the user has complete control over the items they put in their curated playlists. I focused on creating a streamlined experience, where the user had many features at their disposal to inspect and update their playlists, adding depth. An example of this would be the ability to preview a selected track.

Additionally, I feel I followed the VUI Design Principles well, making sure to follow each point. The project is flexible allowing it to be ported over for use in a variety of systems such as a movie library or booking manager. This is possible due to the dynamic data storage with Google Sheets and the breakdown of the project into distinct flows which each serve a role, allowing components to be swapped out.

To improve I would have liked to link the manager to a music account (Spotify). The bot would then resemble the industry standard more closely, being usable in a real-world application rather than for a university project. Further, I would have liked to add entities when selecting data like names, albums, or tracks. I originally tried to add these to create a more dynamic prototype, however, with the nature of the data fetching where any input could be valid, utilising entities caused certain inputs to be misunderstood, so I had to use captures instead. I currently allow users to get a link to preview a track, but would have liked the audio to be played through the bot. This wasn't possible due to the "stream" function being unfortunately deprecated. Finally, instead of using Google Sheets as data storage I could switch to a proper database management system such as Airtable.