Mood Magic Therapy

A multimodal music therapy app which **truly** matches users' moods to the emotional tone used by the artist

Marcus Watts 1343313
Supervisor: Dr. Phillip Smith

Abstract

Mood Magic Therapy is a Google Assistant application which uses Sentiment Tone Analysis to match the user's mood with the emotions conveyed by musical lyrics. The application acts as a Music Therapy provider, recommending suitable pre-tone-analysed songs to the user based upon their mood. As a multimodal song analyser, the application also provides melodic analysis in addition to lyrics analysis. The objective of the application is to provide a music recommendation system which places lyrics in the forefront on analysis whilst also providing more information and freedom to the user to choose what they would like to listen to whilst speaking to a Google Assistant device such as a mobile phone or a smart speaker.

Acknowledgements

I would firstly like to thank my project supervisor, Dr Phillip Smith, for guiding me through this project with great enthusiasm and confidence in my abilities, despite a seemingly overwhelming project at first glance. Second, I would like to thank my parents for their continued support from start to finish both in this project and during my life. Third, I would like to thank Google for giving me a free Google Assistant developers T-shirt upon releasing Mood Magic Therapy onto the Actions-on-Google console. Finally, I would like to thank Amber Simpson for providing great cooking, warmth and her own version of the Mood Magic Therapy ethos unto me throughout this project and throughout this year.

Contents

Table of Contents

CONTENTS **3**

1. BACKGROUND **6**

1.1 Overview 6
1.2 The Problem 6
1.2 Background 7
1.2.1 Sentiment Tone Analysis *7*
1.2.2 Music Mood Analysis and the Role of Lyrics *8*
1.2.3 Where this Project Fits In *9*
1.3 Project Significance 10
1.3.1 From a Business Perspective *10*
1.3.1 From a Mental Health Perspective 10
1.4 Technical Details of the Services Used 11
1.4.1 IBM's Watson Tone Analyser Service *11*
1.4.2 Google Assistant and the 'Actions-on-Google Console' *12*

2. PROJECT SPECIFICATION **12**

2.1 Overview 12
2.2 Functional Requirements 13
2.3 Non Functional Requirements 14
2.4 A Typical Use Case Scenario 15

3. PLAN **17**

3.1 Overview 17
3.2 Objectives and Milestones 17
3.3 Project Gantt Chart 18

4. SYSTEM DESIGN **19**

4.1 Overview 19
4.2 The Database 19
4.3 The Google Assistant Application 19
4.4 The Spotify API to Google Cloud Webhook Connection 20

5. IMPLEMENTATION **25**

5.1 Overview 26
5.2 Creating The Database 26
5.3 Creating the Google Assistant Application 33
5.3.1 Creating the Conversational App *34*
5.3.1.1 Launching the App 34
5.3.1.2 Intents 34
5.3.1.3 Entities 35
5.3.1.4 The Challenge of User Input Solved Using the 'Checker System' 37
5.3.1.5 Events and Groups Entities 38
5.3.2 Creating the Fulfilment Webhook behind the Conversational App *40*
5.3.2.1 Intent Mapping: 'playSadSong' Function 41
5.3.2.2 'recommendASong' Function 44
5.3.2.3 'giveAudioAnalysis' Function 44
5.3.2.4 'playMusic' Function 46
5.3.3 Conversational Loops and Ending of Conversation *47*

6. TESTING 49

6.1 Overview 49
6.2 Unit Testing Results 49
6.3 integration Testing Results 50
6.3.1 Methodology 50
6.3.2 Testing the correct response and intent matching throughout the Path 50
6.3.3 Testing the Appropriateness of the Response (NFR4.2) and Closeness to being a Human Response (NFR4.3) 51
6.3.4 Evaluating whether the Song matched the user's Emotion (FR6.4) 51
6.4 Testing the accuracy of the Pre-Analysed Song Database (FR2.0-2.3) 51

7. EVALUATION AND FUTURE DEVELOPMENT 52

7.1 Overview 52
7.2 Testing Limitations 52
7.3 Limitations of the System 53
7.4 Future Development 56

8. CONCLUSION 57

9. BIBLIOGRAPHY 61

10. APPENDICES 65

Appendix A – Glossary of terms 65
Appendix B - Node.js Libraries and Dependencies Used 66
Appendix C 67
i) Angry Unit Tests 67
ii) Sad Unit Tests 70
iii) Happy Unit Tests 74
iv) Confidence Unit Tests 77
v) 'playMusic' Function Testing Results for each emotion 81
Appendix D 82
i) Comparing the Console Log scores of 'Dance' by 'Abba' with the Output SQL Table 82
ii) Output SQL Table showing Abba's 'Dance' Sentiment Tone Analysis Scores 87
ii) Output SQL Table Showing the Same Values After 15 Repetitions for Elis Presley's 'All Shook Up' 87
Appendix E 88
i) Copy of the Email Containing the questionnaire and the instructions on how to Complete It 88
ii) Averages of the Results from all the responses 91
iii) Total of the Results from all Responses 92
Appendix F 92
i) The lyrics of Celine Dion's Cover of 'All By Myself' 92
ii) SentiWordNet Analysis of Celine Dion's cover of 'All By Myself' 93
Appendix G 97
i) Searching for the application on the Google Assistant Actions-on-Google Console 97
ii) The application's page on Google Assistant showing the list of Supported Devices 98
Appendix H 98
Setup 98
Repository 98

Introduction

Smart home technology and music streaming are both highly competitive industries which are growing at a phenomenal rate (Kinsella, 2019). In order to provide a competitive edge, a user-centred approach must be taken. Spotify, one of the most subscribed-to music services, provides user-specific and highly customisable playlists that can be played by smart home devices such as the Google Home Hub speaker. Although users can ask to hear songs of a desired mood, music streaming services such as Spotify fail to take into account the emotions conveyed by the lyrics of the songs that are selected. This can lead to user frustration when receiving songs which have lyrics that do not fit the desired mood or it can lead to ineffective 'music therapy' experiences – one of the key reasons for listening to music. Based on the benefits that 'music therapy' – the practice of using music to improve one's mental wellbeing - can provide, the problem which this project addresses is the lack of song lyrics analysis within song suggestions for users searching for a song in a specific mood. The lack of either any music streaming company or any smart home appliance in offering this will also be tackled.

This project uses 'Sentiment Tone Analysis' to provide users with a system that matches users' moods with the emotions conveyed by artists' song lyrics. It does so through a conversational 'Google Assistant' application which provides support for users, particularly those suffering mental illnesses such as anxiety and depression, by having an outlet to express one's feelings in addition to regulating moods through 'music therapy'. The application also aims to give users information about the song recommendations in addition to the freedom with which to listen to the suggested song, choose another song or end the conversation altogether.

The system features three main parts: 1. A Google Cloud hosted 'Firebase database' of pre-tone-analysed songs created using IBM's 'Sentiment Tone Analyser', 2. A conversational application made using Google's 'Actions-on-Google' developer studio in addition to Google's 'Dialogflow', and 3. A Google Cloud hosted JavaScript 'Webhook' which connects the 'Firebase database', the 'Google Assistant' application and the Spotify API for providing melodic analysis and for playing music recommendations through the connected device's speaker.

The following report will outline the process in which this system was built. The first section, *Background*, will outline the motivations behind the project, the relevant research and the project significance. The second section, *Project Specification*, shall outline the requirements necessary to fulfil the aims of the project. The third section, *Plan*, will show the preparation process and what the project milestones were. The fourth section, *System Design*, shall outline the design choices and justifications for making them. The fifth section, *Implementation*, shall detail exactly how each section of the system was implemented. The sixth section, *Testing*, shall outline the testing process and how successful the different elements of the system were in providing fulfilment for the original project specification. The seventh section, *Evaluation and Future Development*, shall discuss how the project's successes were evaluated against the project limitations and failures, in addition to discussing how the system could be improved. Finally, the eighth section, *Conclusion*, shall weigh up

the project's key successes and failures, passing judgement on its ability to tackle the original problem.

The ninth section, *Bibliography*, contains the references used in writing this project and the tenth section, *Appendices*, will detail any further information used including test results and explanatory diagrams, and provides a link to the project code itself via GitHub.

1. Background

1.1 Overview

This section discusses the motivations for the project, alongside the technical aspects of the services used. First, the problem which the project aims to address is discussed. Second, the relevant background research and scientific grounding for the project is presented. Third, the project's significance is outlined. Finally, fourth, the technical details of the services which the project uses are outlined.

1.2 The Problem

'Music Therapy' is the practice of using music to improve one's emotional state (Davis, Gfeller and Thaut, 2008). Although music therapy has been clinically proven to provide numerous mental health benefits (Bailey, 1986; Bunt, 1994), automated playlists on music streaming sites such as 'Spotify' fail to take into account the sentiments within the music lyrics. This is problematic for users since they may choose a song from a particular mood which has lyrics that fail to meet the emotion they desire. This can result in a poorly matched song which fails to provide any music therapy in the best case scenario, or one which is so badly matched that it becomes detrimental to the user's mental health in the worst case scenario; This potential negative effect of music therapy is consistent with findings which show that psychotherapy can indeed pose the risk of deteriorations in the mental health of patients, notably, from inappropriate types of therapy (Lambert and Vermeersch, 2002). It is also worth noting, however, that this risk may be less significant when providing music therapy in comparison to psychotherapy since psychotherapy is typically more intrusive and aimed at patients with high grade mental health issues. Despite this possible disagreement, it remains the case that music therapy can be more beneficial to users if song selections are more attuned to what the user desires. This is especially important for users with low and medium grade mental health issues, as shown by Perez et al's (2010) discovery that music therapy is more effective than psychotherapy in these cases.

Moreover, users experiencing extreme emotional events, such as relationship difficulties or the passing of a family member, may also be more likely to examine the lyrics in songs to help provide music therapy than those who are not. This is explained by the psychological theory of 'Confirmation Bias' wherein people typically use new input data to fit, or confirm to, their own attitudes and beliefs (Klayman, 1995). Users facing these scenarios may therefore have even more of a need for songs with lyrics that match their emotional state so

that the information they receive via the lyrics is useful and applicable to their situation. Furthermore, combining the correct artist's emotional lyrics to the correct use case scenario will also finally bridge the gap between how the artist felt when they wrote the song and how the user feels when listening to it.

The aim of this project is therefore to tackle the problem of mismatching song lyric semantics to the song's allotted mood category. The project will therefore place song lyric semantics at the forefront of song mood categorisation but will also utilise a multimodal analysis based on the audio features and audio analysis of the songs to provide further information to the user. This will help to provide users with music that more closely matches their mood.

1.2 Background

1.2.1 Sentiment Tone Analysis

Semantic Tone analysis stems from the fields of 'Psychoanalysis' and 'Psycholinguistics' in Psychology. Freud (1940) posited that perceptions can be garnered from the perceptual information which we sub-consciously make based on our choice of language. This was developed through much of the 20th Century with psychiatrists using language for psychoanalysis of patients; Where patients' real emotions were presented, not in what they were saying, but, in the *tone* they used when speaking (Thomas and Ogden, 1997). It was in the early 21st Century, where computers became readily available to the general public and opinion data was dispersed across the internet via websites such as 'Twitter', 'Trip Advisor' and 'Facebook' (Gou *et al.*, 2014), which finally melded the two disciplines of Psychoanalysis and Machine Learning under the field of Sentiment Tone Analysis (Hirst and Liu, 2012).

Sentiment Tone Analysis is a field of 'Natural Language Processing' which aims to analyse emotions, attitudes and opinions from written language (Hirst and Liu, 2012). Extraction is the first stage whereby, given a piece of text, words or phrases are chosen which are deemed important. There are various techniques used to help identify the 'important' words or phrases. These include: Part Of Speech (POS) analysis, which tags each occurrence of a word with its POS identifier – for example a noun or verb – which can be useful in identifying each word's role and polarity in the wider context of the text, and Synset Identification, which involves identifying synonyms of the words used and placing them into distinct unordered sets, known as 'synsets' (Miller, 1995; Fellbaum, 1998). A rudimentary version of the latter technique is used in this project to assess the user's emotion by grouping word synonyms and phrases into 'Entity' sets, and both techniques are employed within the 'SentiWordNet' analysis used to analyse the song lyric excerpt from the recommended song to the user; This will be discussed further in the System Design and Implementation sections (sections 4 and 5 respectively).

After extraction, each word must be classified into its respective emotion. As a simplified example, words from the same synset or from inter-synset-relations are likely to be classified into the same emotion. Sentiment classification is a supervised machine learning technique which can be fulfilled through the 'Naïve Bayes' algorithm or by 'Logistic Regression'. In Naïve Bayes, for example, Bayesian classifiers make simplifying assumptions on which class an object is based on its inputs, and the frequency of its inputs, which are *assumed* to be independent of one another. Naïve Bayes would examine the text as a 'Bag Of Words' –

meaning the words are viewed without taking their position into account - rather than evaluating word position in text by POS analysis. As an example, the following text, "I hate to say it, but I'm badly in love.", would be examined based on the frequency of words as follows: 'I' = 1, 'hate' = 1, 'to' = 1, 'say' = 1, 'it' = 1, 'but' = 1, 'I'm' = 1, 'badly' = 1, 'in' = 1, 'love' = 1. If the objective is to determine whether the text should be classified as positive or negative, the negative words used 'hate' and 'badly' minus the only positive word 'love' would cause it to have an overall classification of slightly negative. This would, of course, be more challenging had the original sentence been "I hate to say I love him!" since there would appear to be an even number of positive and negative words. Despite these problems, Naïve Bayes remains a popular algorithm in text classification as a result of its speed and ease of implementation and, with some minor alterations, the negative effects of any incorrect assumptions can be minimised (Rennie *et al.*, 2003). The 'SentiWordNet' algorithm similarly aims to categorise text based on its positivity/negativity value but instead of treating the text as a 'Bag of Words', each word is scored based on factors such as its position within the text and the synset it belongs to along with the synset relations it has to other words (Miller, 1995; Fellbaum, 1998). The existing word classification techniques and algorithms all have varying degrees of success based on which text it is used upon; The predominant Sentiment Tone Analyser – IBM's Watson Tone Analyser – will be discussed later in this section and evaluation of all the semantic tone analysis techniques and services used in this project will be discussed in the Testing section (section 6).

1.2.2 Music Mood Analysis and the Role of Lyrics

There has been an enormous surge in attempts to classify musical moods starting from the work by companies such as MoodLogic, founded in 1997 - which created a user generated database of music mood categorisation (Wikipedia, 2005), and the annual Music Information Retrieval Evaluation eXchange (MIREX)'s inclusion of a mood classification challenge in 2007 (Feng, Zhuang and Pan, 2003) - whereby audio analysis aligned with human validation was used to classify songs into five emotion categories.

Although important for the development of mood classification systems, many of the categorisation choices were found to be highly dependent on which algorithm was used. Many choices still required subjective human validation and the lyrics were being ignored (Hu *et al.*, 2008). Within the same couple of years, a journal article in the 'Psychology of Music' found that the melodies of music played a larger and more dominant part in the emotional responses conjured up by the listener (Ali and Peynircioglu, 2006). This aligned with Logan, Kositsky, and Moreno's (2004) study where song lyrics were first analysed to determine artist and genre but they were found to be less useful in categorisation than audio analysis. Despite this, they suggested that perhaps song lyric Sentiment Tone Analysis could be used together with Audio Analysis; This was furthered by Neumayer and Rauber (2007) who stated that textual data from song lyrics may provide a useful additional layer for the categorisation of music. Multimodal song analysis with the inclusion of Natural Language Processing of song lyrics was then examined with studies showing a 21% relative improvement in classification accuracy over using only audio analysis alone (Yang *et al.*, 2008).

This multimodal approach to attaining sentiment from music aligned with studies in Cognitive Neuroscience. For instance Peretz and Coltheart (2003) claimed that, upon hearing a sound, the lyrics of the song are processed in the musical lexicon module in parallel to the pitch content and temporal content modules of the brain. Furthermore, it also aligned with

Omar and Peynircioglu's (2006) paper which concluded that, although song lyrics lessened the emotional responses from happy and calm music, they actually intensified the responses to music with negative emotions. The surrounding research seemed to suggest that, even if lexical information has a relatively small effect on the sentiment garnered through music, it certainly makes an impact on the sentiment that the music releases nonetheless. This was explored further from a Music Therapy perspective where a notable study by O'callaghan, O'brien, Magill, and Ballinger (2009) exemplified the importance of lyrics in songs for children whose parents previously left them orphaned due to contracting cancer. Therefore, it seems an important feature to take into account when conducting the overall sentiment used in music (Laurier, Grivolla and Herrera, 2008).

Whilst musical tone analysis can be provided by machines through measuring chord progression and key changes (Katayose, Imai and Inokuchi, 1988), this project will combine Audio Analysis with Semantic Tone Analysis thus providing a fully multimodal mood analyser.

1.2.3 Where this Project Fits In

In recent studies, the direction has transitioned from proving *why* lyric analysis is useful in music sentiment categorisation to demonstrating *how* to do it with large data sets. Choi, Song and Kim (2018) developed a method using 'Billboard Hot 100' crawlers to generate music data and lyrics crawlers from the 'Genius' library to find the lyrics and match them with the Billboard chart information. This was followed by Natural Language Processing and lemmatisation, a Semantic Tone Analysis technique whereby the inflected forms of a word are grouped together to form one single 'lemma' group; For example 'be', 'being' and 'been' would all fit under the lemma of 'be'. Once complete, the lyrics were used to give each song a sentiment score as to which emotion it best fits, given Ekman and Friesen's (1971) 'six senses': anger, disgust, fear, joy, sadness and surprise. Finally, when being posed with a new song, the 'K-Nearest Neighbour' algorithm could be used to match the new song's sentiment analysis to the pre-categorised song set which it most closely resembles based on the 'Euclidean Distance' between them. Whilst the objective of Choi, Song and Kim (2018) was to build a system that would recommend the most closely matching sentiment to the song currently being listened to, this paper aims to build a system that allows users to ask for a song which fits under an emotion of their desire from a pre-tone-analysed database. Moreover, the emotions chosen for this project are also based off Ekman and Friesen's six senses. Instead of using all six however, since three in particular were deemed unsuitable for music therapy, 'disgust' and 'fear' are replaced by 'confidence' and 'surprise' is not featured in this project.

Most recently, Saluja, Jain and Yadav (2019) developed an algorithm for mining 55,000 song lyrics from www.genius.com and used APIs from www.developer.musicmatch.com to match song lyrics with the correct song titles and artists after being processed. The outputted data from their project, found here www.kaggle.com/moousehead/songlyrics#songdata.csv, was used as the foundation for this project since it contained all the necessary information to begin creating the project's database on 'MySQL'; This will be discussed in further detail in the System Design and Implementation sections (sections 4 and 5 respectively).

1.3 Project Significance

1.3.1 From a Business Perspective

From a business perspective, music streaming companies, such as 'Spotify', are facing extreme amounts of competition. It was rumoured in 2019 that 'Apple Music' has overtaken 'Spotify' in annual US subscription numbers (Li and Nellis, 2019). This is somewhat disputed, however, and it has since been claimed that Spotify has reclaimed the top spot for US subscriptions after rolling out several package deals and offers, one of which being with 'Google' offering a free 'Google Home hub' to family account members (Solsman, 2019). Despite this lack of clarity over which company has more customers, it can be concluded with certainty that music streaming competition has never been more intense. For this reason, music streaming companies have been eager to implement distinguishing features which may encourage more subscribers. From zodiac star-sign-inspired playlists that change every month based on the user's birth month, to using algorithms that aim to create user-specific automated playlists based on audio analysis on the scores of traits such as 'positive valance' and 'danceability', Spotify's driving success has been largely as a result of the high degree of user personalisation (Sun, 2019).

Although Spotify currently has playlists using 'moods' such as 'happy' and 'sad', there is no option to select a series of songs which analyse the tone of the lyrics themselves. Likewise, there are no options to select a song with lyrics of a certain emotion via the Google Home hub, which also faces extreme competition from Amazon and Baidu in the smart speaker market, having fallen to third place in worldwide sales in 2019 (Perez, 2019).

Through creating the functionality which allows users to ask for songs which have lyrics in a certain emotion in addition to being able to provide users with song excerpts and additional Audio and Feature analysis, both Spotify and Google may provide for themselves a competitive edge within these markets. This could be furthered with future iterations to allow users to ask for songs which contain specific subject matter, for example "can you play me some songs which mention 'cats'", or perhaps in the far future when smart speakers become synonymous with 'Neurological Links' – an ultra-high bandwidth human brain-machine interface which connects humans and computers (Neuralink, 2019) - users may be able to think "play me a song to match my mood" and based off the user's physical and mental health data, they can listen to suitable emotion-matching songs whilst the room lighting is changed to a suitable emotion-matching colour; This will be discussed in more detail in the Evaluation and Future Development section (section 7).

1.3.1 From a Mental Health Perspective

From a mental health perspective, Mood Magic Therapy - the application which this project develops - may provide an invaluable resource in bridging the gap between how user's feel and how artists felt when writing their songs, offering 'closeness' to the artist which is otherwise unavailable through audio analysis alone. In the majority of cases, appropriate music selection for use in music therapy should begin with matching the mood of the user with the emotion conveyed played in the music (Bailey, 1986). This project will be able to provide the only multimodal music sentiment analysis system available via a smart speaker or

on any music streaming service which will result in more effective user-emotion–song-emotion matching and, therefore, more effective music therapy.

Mood Magic Therapy shall also provide users with more information and freedom than any current application for smart speakers or offered by music streaming services. In addition to using music therapy through emotion-*matching* songs, it is also used to control, change and regulate one's mood through choosing songs which convey a *desired* emotion (Schäfer et al., 2013). This system will also allow this as users can utter, for example, "I would like a happy song please!", and they will be recommended a 'happy' song. Users will be provided with information, first, with a focus on the lyrics; The system will explain the emotional tone scores and provide a brief reading of the lyrics used along with further analysis to explain whether it is positive or negative. Second, the user will be provided with audio and feature analysis of the song in addition to the emotional meanings that the key and mode typically convey. In addition to this further information, users will be provided with the freedom to ask for a different song, play the song or decline a song of the selected emotion before or after hearing the audio analysis. This extensive amount of song information which focusses on emotional aspects of music is not available on any music recommendation system available to users. This information combined with the freedom offered to the users allows them to make informed decisions based on the music they listen to. This will enable them to take more control of their listening experience in addition to being able to select the perfect song for them in the moment. This, once again, should result in more effective music therapy and a better listening experience for the user.

As mentioned by Schäfer et al. (2013), the song emotion selected for users undertaking music therapy match the emotion as closely as possible to begin with and then gradually progress into a more positive song emotion. Based on this, perhaps future iterations could include a feature which creates a playlist of songs to play sequentially to the user; beginning with songs that match the emotion of the user, followed by increasingly positive songs. This will be discussed further in the Evaluation and Future Development section (section 7).

1.4 Technical Details of the Services Used

1.4.1 IBM's Watson Tone Analyser Service

The Sentiment Tone Analyser used to create the database of pre-tone-analysed songs in this project is the 'Watson Tone Analyzer'. Given a piece of text, the service uses linguistic analysis techniques to define the certainty that the text fits within a certain emotion at both a document level and a sentence level – '0' being entirely uncertain, 0.5+ being more certain than not, 0.75+ being extremely certain and '1' being entirely certain of the text fitting that specific emotion.

The service utilises the following four 'emotional' tone categories: 'anger', 'fear', 'joy' and 'sadness' and the following 'language' tone categories: 'analytical', 'confident' and 'tentative' (IBM, 2017). Only four emotional categories ('anger', 'joy', 'confidence' and 'sadness') are used in this project, however, since the remaining categories outputted by the Watson service were deemed unnecessary or irrelevant to providing music therapy.

Specifically, the Watson Tone Analyser uses a stacked generalisation-based ensemble framework, meaning it combines multiple lower level analyses together with higher level analysis to provide a more thorough word and word-context analysis. The Watson Tone

Analyser analyses based on punctuation, swear words, greetings, sentiment polarity and 'N-grams' (IBM, 2019). Similar to how POS analysis used in 'SentiWordNet' takes into account word position in categorising sentiment, 'N-grams' are any number of words grouped together as a single 'gram', or phrase, which alter the phrase's meaning. For instance, "I am not happy", would have the 'Bigram' of 'not happy'. The 'not' in the Bigram would change the meaning of the utterance's subject from being 'happy' to being the opposite; Thus the sentence categorisation would be 'sadness' in this instance. The reasons for choosing IBM's 'Watson Tone Analyser' and 'SentiWordNet' will be discussed further in the System Design section.(section 4)

1.4.2 Google Assistant and the 'Actions-on-Google Console'

Google Assistant is Google's cloud-based voice assistant service, available on any internet-enabled device which has a microphone and a speaker, which uses its own machine learning API service 'Speech-to-Text' to convert users' speech commands into written text (XAPPmedia, 2018; Google, 2019a). The text strings are then matched with actions, represented by 'Intents', which are the functions that manage the specific responses that occur based on a specific string of text (Google, 2019f). For instance, a user can utter to the Google Assistant "Hello Google", this will be recorded by the device microphone, converted to text and matched with Google Assistant's default 'greeting intent'. The 'greeting intent' will have a stored response string which is sent through the 'Text-to-Speech' API and the Google 'agent', the spoken voice of the Google Assistant device, will be played out of the device speaker: "Hi there, [user's name], how can I help?". For more complex user requests, such as asking what the weather is for the day, the matched intent will have the function of calling out to the weather service API with the user's location data. The response of the API call is then combined with specific and applicable pre-set speech depending on the type of data received. This, again, is sent back through the Text-to-Speech API and outputted through the device speaker: "In [user location] today, it will be [weather data, for example, mostly cloudy with a 20% chance of rain]." This project uses both intents which simply reply to users via pre-written responses, and intents which make an API call to the 'Webhook' code - Google cloud hosted code - which fulfils intent functions that require external API calls such as to the 'Firebase' database, the 'SentiWordNet' API or the 'Spotify' API endpoints.

The service used in this project to design and write 'Intents' is Google-owned 'Dialogflow' (formerly, API.ai). This controls the conversational aspect of the application and is a part of the 'Actions-On-Google Console', a developer studio for creating Google assistant applications. The specific details of how these services were used within this project are detailed in the System Design and Implementation sections of this project. Before designing the system, however, it is first essential to outline the key specifications which the project aims to fulfil.

2. Project Specification

2.1 Overview

This section shall outline the system requirements as set out from the beginning of the project. They take the form of Functional Requirements (FRs), which are defined as the features of the system – or, what the system *can do*, and Non Functional Requirements (NFRs), which are the system constraints that impact the system as a whole and show *how well* the system can perform the FRs (Rashwan, 2012). Only the Non Functional Requirements which are relevant to the system are included. Since 'Use Cases' help to convey System Requirements (Sindhgatta and Thonse, 2005), this section finishes with a typical 'Use Case' which forms one of the possible 'Use Case' scenarios used in the Testing section (section 6).

2.2 Functional Requirements

1.0 Pre-Sentiment-Tone-Analysed Database		
ID	**Requirement**	**Priority**
FR1.1	The system must use IBM's Watson Tone Analyser to create a database of pre-sentiment-tone-analysed songs.	H
FR1.2	The database of songs must have the artist and song names, in addition to the emotion that was found to match the song with the highest certainty, the sentence which matched the emotion with the highest score the most along with its certainty score for matching it.	H
FR1.3	The songs in the database must be positioned with their correct emotion and score.	H

2.0 User Authentication		
ID	**Requirement**	**Priority**
FR2.1	The system must allow the user to connect their Google and Spotify accounts.	H
FR2.2	The system must remain connected to the Spotify premium account for the duration of the use of the application.	H

3.0 The User Interactions with the Google Assistant Application		
ID	**Requirement**	**Priority**
FR3.1	The system must be able to 'listen' to the user via the microphone.	H
FR3.2	The system must be able to differentiate between which of the four emotions to designate to the user based off their speech.	H
FR3.3	Once designated an emotion, the system must ask the user to confirm that it is the correct emotion. If it is incorrect or if the user would like a song in a different emotion, the system must allow the user to change their designated emotion before recommending a song.	H
FR3.4	The system must allow the user to ask for another song after being recommended it and after hearing audio analysis of it.	H
FR3.5	The system must have a fall back intent in place for when the user utters a phrase which it is not trained to deal with.	H
FR3.6	The system must allow for the user to end the conversation during their interaction with it.	H

4.0 Providing Song Recommendations		
ID	**Requirement**	**Priority**
FR4.1	The system must be able to retrieve and recommend a random song with the same emotion as the user, from the pre-semantic-tone-analysed song database.	H
FR4.2	The system must be able to read out an excerpt from the song which scored the highest certainty of being from its maximum emotion.	H
FR4.3	The system may be able to use 'SentiWordNet' analysis on the song lyric excerpt to determine if it is a positive or negative statement and tell the user the result.	M
FR4.4	The system must then offer to provide Audio and Audio Feature Analysis to the user, but will give them the option to decline it.	H

5.0 Providing Audio Analysis and Audio Feature Analysis		
ID	**Requirement**	**Priority**
FR5.1	Upon the user's confirmation that they want further analysis of the song, the system must provide Audio Analysis and Audio Feature analysis of the song's tempo, key, mode, how suitable it is for dancing to, its valence and the emotions that the key and mode typically convey.	H
FR5.2	The system must give the option to listen to the song, end the conversation or to recommend another song of the same emotion.	H
FR5.3	The analysis provided must match the analysis provided by Spotify for that specific song.	H

6.0 Playing the Song Through The User's Connected Premium Spotify Account		
ID	**Requirement**	**Priority**
FR6.1	After the user confirms that they would like to hear the recommended song, (before or after audio analysis), the system must be able to play the song via the user's connected Premium Spotify account.	H
FR6.2	The system will end conversation once the user has confirmed that they want to listen to the recommended song (before or after audio analysis).	H
FR6.3	The song which is played must match the recommended song from earlier in the conversation.	H
FR6.4	The song may fit the intended mood as per the user's own subjective criterion.	M

2.3 Non Functional Requirements

1.0 Availability		
ID	**Requirement**	**Priority**
NFR1.1	The system must be available for 99.4% of operating time.	H
NFR1.2	The system should have a maximum of one crash for every 2 hours of usage.	H

NFR1.3	Once down, the system should reboot within 60 seconds.	H
NFR1.4	Upon launching the application via the 'invocation phrase', ("Ok Google, Talk to Mood Magic Therapy"), the application must start up and be ready to use within 5ms.	H
NFR1.5	The system application must be available to use on the Google Home hub and other Google Assistant compatible devices such as phones, laptops and tablets.	H

2.0 Safety and Security		
ID	**Requirement**	**Priority**
NFR2.1	The system must not allow users to edit/overwrite/delete any of the code or database entries.	H
NFR2.2	The system must not be used by users who do not have a Spotify premium account.	H

3.0 Efficiency		
ID	**Requirement**	**Priority**
NFR3.1	The system must respond to the user within 5 seconds of recording and processing their speech.	H

4.0 Accuracy		
ID	**Requirement**	**Priority**
NFR4.1	The system must be able to detect input speech and process it as text with an 85%+ accuracy.	H
NFR2.2	The system may provide an agent response which is at least 70% appropriate to the user's speech.	M
NFR2.3	The system may provide an agent response which has 70% closeness to a human response.	M

5.0 Regulatory		
ID	**Requirement**	**Priority**
NFR5.1	The system must protect users' data in accordance with the General Data Protection Regulation (GDPR) 2018.	H

6.0 Usability		
ID	**Requirement**	**Priority**
NFR6.1	The system should be easy to use without any training.	H
NFR6.2	The system should intuitively guide the user through the conversation process.	H

2.4 A Typical Use Case Scenario

The following Use Case models the scenario in which the user asks for the first recommended song to be played after hearing its audio analysis. It begins after the user has already connected the Google Assistant application to their Premium Spotify account, and thus assumes that they indeed have a Premium Spotify account. The example showcases the 'sad' emotion; however the use case can be mapped onto any of the four emotions.

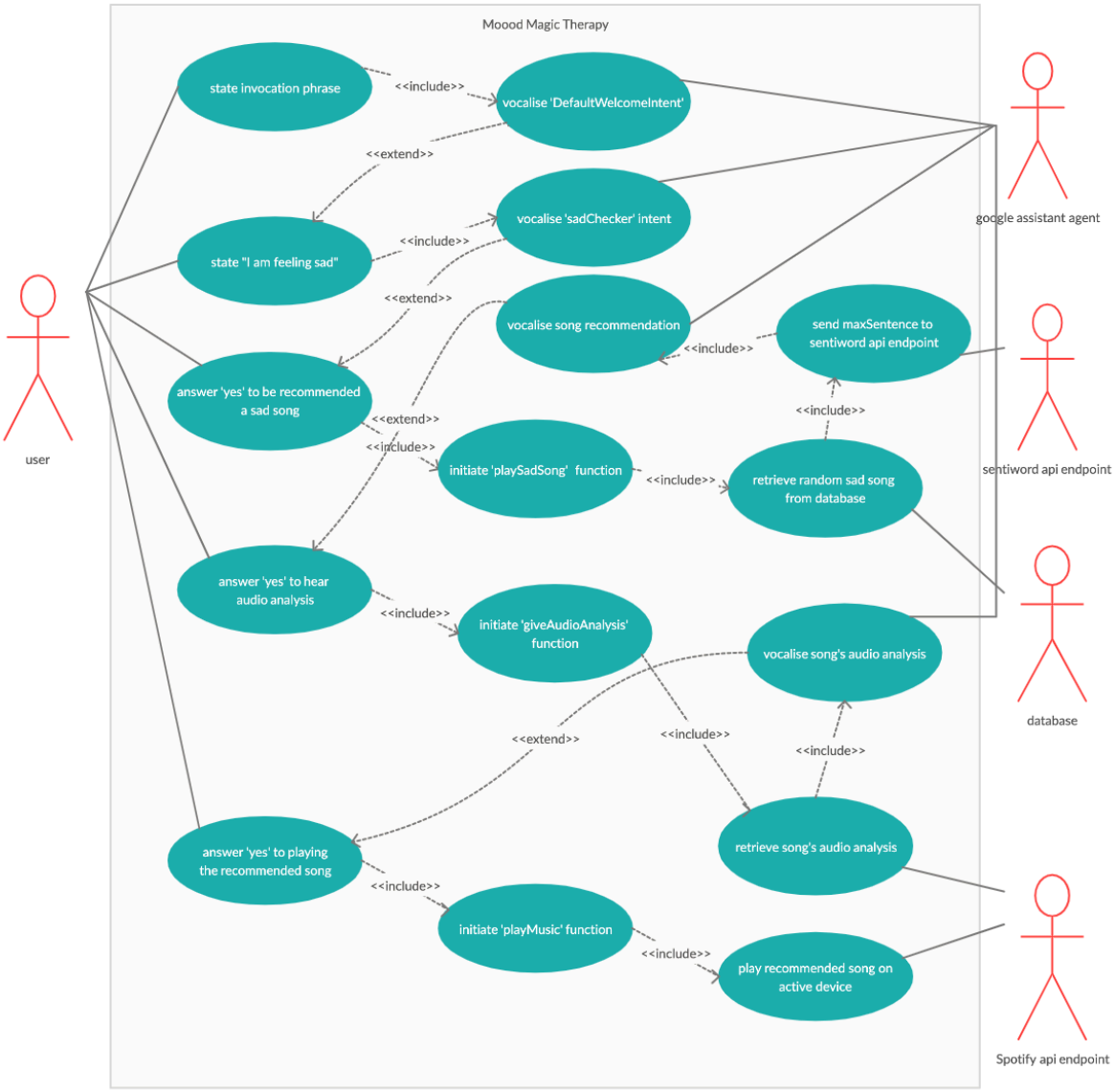


Figure 1: Use Case diagram for being recommended a sad song, hearing its audio analysis and finally listening to it – UCID 1 (See Appendix EI for all the Use Cases used in Testing)

3. Plan

3.1 Overview

This section outlines the organisational approach taken to tackling the project. It includes the objectives and milestones for the project which will be shown through the use of a Gantt Chart.

3.2 Objectives and Milestones

It was clear from the beginning that this project would consist of three main parts: 1. Creating the database of pre-tone-analysed songs, 2. Creating the Google Assistant application which would be able to converse with the user and recommend suitable songs from the database and 3. Connecting the application to the Spotify API endpoint so that audio analysis can be retrieved and songs can be played. This manifested itself as fourteen distinct milestones:

The Database

1. Create the SQL database containing songs with their artist name and lyrics.
2. Write the code, in 'JavaScript' using a locally hosted 'Node.js' library, which scans the SQL database row by row. For each row, the code must send a request to the IBM Watson Tone Analyser with the song's lyrics. The code must take the response of the analysis and output it into a new SQL table.
3. A cleaned copy of the table must then be imported into the Firebase database.

The Google Assistant Application

4. The conversational style Google Assistant application must be created using the Actions-on-Google Console.
5. The application must be linked with the Firebase database so that the system can retrieve songs from the database to recommend to the user.

The Spotify API Endpoint

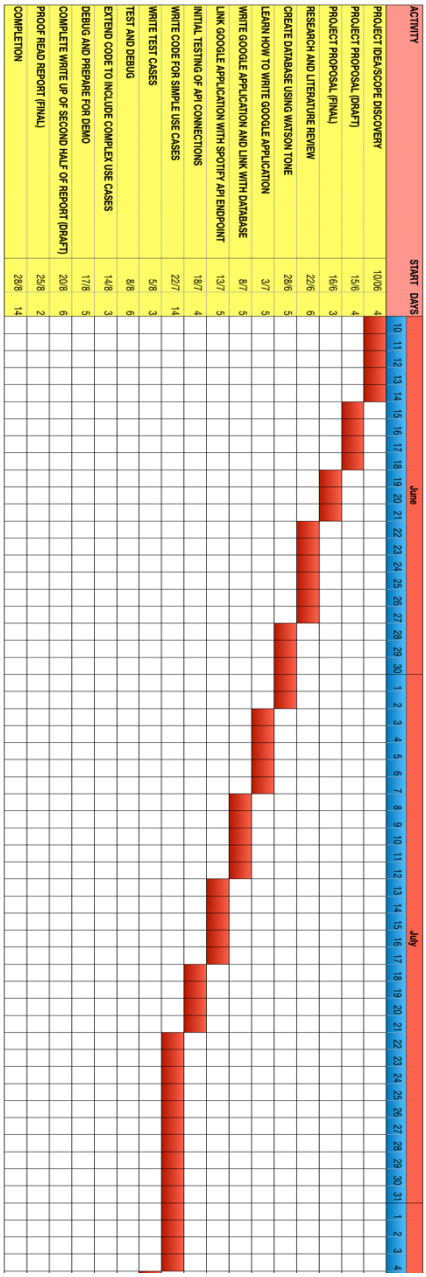
6. Connect the Spotify API to the Google API so all parts of the application are connected.
7. Use the Spotify API to give the user audio analysis of the song and play the song via a Spotify Premium account.

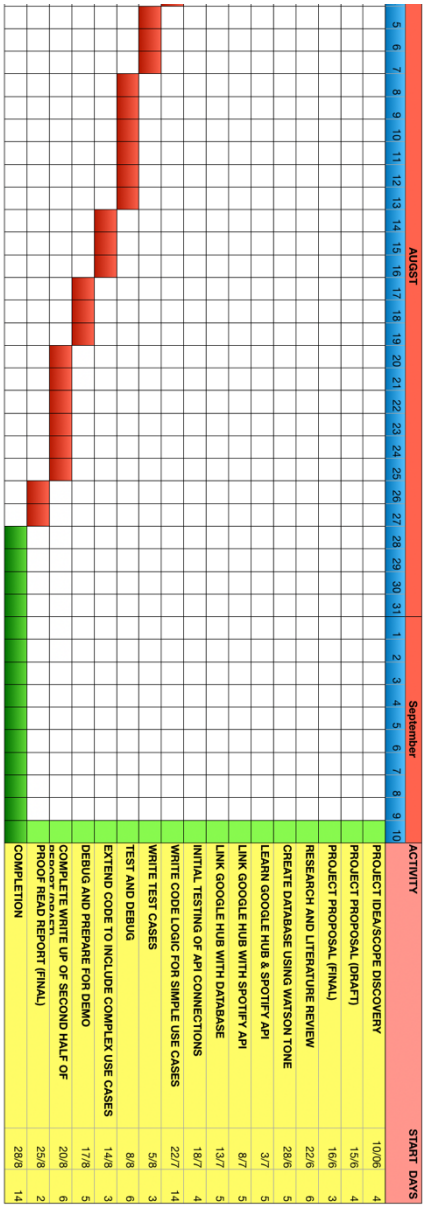
Testing, Debugging and Finalising the Application

8. Test the API connections between the database, Spotify API and Google hub API
9. Write simple test cases which ensure that the application works for simple use cases e.g. them saying "I feel happy".
10. Debug the code upon reviewing the results.
11. Add 'SentiwordNet' analysis using a Node.js library
12. Add more conversational features, such as events, to the Google Assistant Application.
13. Write more tests for more complex use cases e.g. users saying "I just broke up with my girlfriend!"
14. Test and debug the code to a satisfactory standard.

3.3 Project Gantt Chart

The following Gantt Chart was produced at the start of the project. Although the lengths of each task deviated from the displayed times to some degree, the overall project organisation and order of task completion did not.





4. System Design

4.1 Overview

This section will discuss the architecture and services used, and how they are used, within the entire system. To create the database of pre-tone-analysed songs, 'SQL', IBM's 'Watson Tone Analyser' and Google's, 'NoSQL' database, 'Firebase', was used. To create the user-to-app conversational experience, 'Google Assistant' entwined with 'Dialogflow' (formerly, 'API.ai'), was utilised. Finally, to allow for audio analysis and the playing of music via the application, 'Spotify's' API service was used. These were all connected together via a single 'Webhook' – code that acts as an API endpoint, which in this project, communicates first with the Dialogflow web API service – written in 'JavaScript' with the use of 'Node.js libraries'. These libraries were helpful throughout the project as they provided a simplistic way of making API requests to external APIs, for example, in allowing for the 'SentiwordNet' analysis of the sentence that scored the highest for the highest-scoring emotion used in the song and by providing the ability to make API requests using the 'request' library when certain library methods failed to execute.

4.2 The Database

'MySQL' was chosen as the service for generating a hosted database. This was required in order to make API request-and-response calls to and from the IBM cloud-hosted 'Watson Tone Analyser' service. Various database services were trialled, however, it was decided relatively early on that MySQL, and its 'MySQL Workbench' application, were among one of the simplest and user-friendly options. Likewise, IBM Watson's Tone Analyser service was extremely user friendly, highly rated among the top sentiment tone analysers (Doerrfeld, 2015; Winchurch, 2018) and accurate; It has been benchmarked against standard data sets with results showing that it is statistically more accurate than even the best reported accuracy of the other most accurate models (IBM, 2019). After sending the lyrics of each song to the analyser service, the outputted tone-analysed songs were exported from their SQL table into JSON format allowing it to then be imported into Google's 'Firebase' database. The 'NoSQL' database was chosen due to its ease of connectivity with the Google Cloud hosted Webhook code and the Google Assistant's API service, 'Dialogflow'. Since there were no tabular relations to tend to, due to the simplicity of my data only requiring one table, the 'NoSQL' formatting complemented the system architecture well and provided efficiency in addition to scalability of the system. Once these connections were made, no further searching for alternatives took place.

4.3 The Google Assistant Application

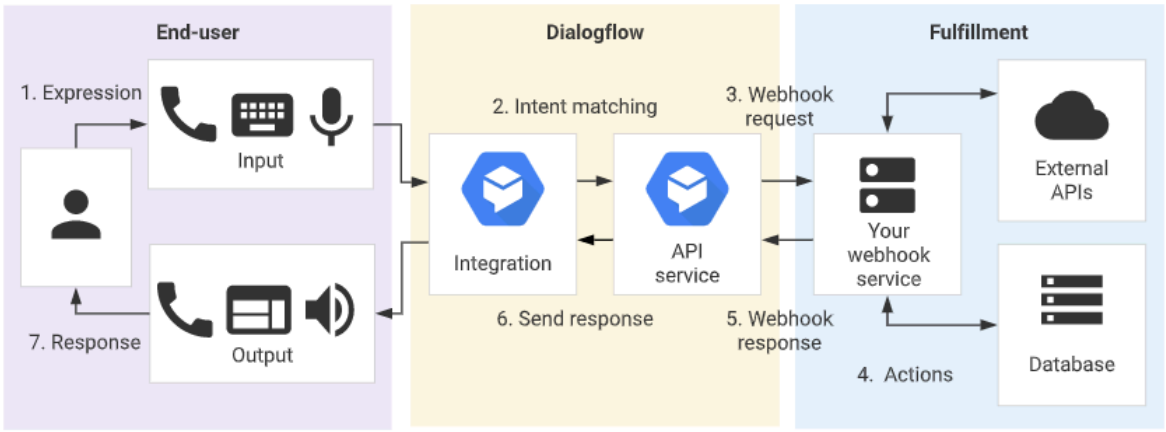


Figure 2: Conversation data flow through the System

The application was always destined for use on a Google Home Hub device. The architecture choices naturally formed around that decision. For instance, in order to create a Google Assistant app, one must follow the 'Actions On Google' developer console. The application needed to have conversational abilities which used a hosted Webhook service to connect to the database and, eventually, the Spotify API. To connect the Google application and to set up the conversational actions, known as 'intents', Google's own 'Dialogflow' was used. This allowed for the setting up of the conversational aspects of the system. As a simple use case, a user could utter "I am feeling sad". This is recorded by the Google Home Hub device's microphone and converted into text Strings, using the built in Google Cloud Speech-to-Text machine learning API, and Dialogflow would then match the emotion 'sad' to the intent for being 'sad'. Once intents have been matched, the Google assistant agent will either respond to the user with a pre-defined response from within the Dialogflow architecture, or it will initiate a request to the Webhook service wherein the Webhook code will match the intent with a specific action – in this case, retrieving a 'sad' song from the database and sending the song recommendation text back through to the Dialogflow API service to be spoken by the agent which is outputted via the Google Home Hub device's speakers.

4.4 The Spotify API to Google Cloud Webhook Connection

The third, and most challenging, stage in the system design was to connect the Google Cloud hosted Webhook to Spotify's API. Spotify was chosen as the music streaming endpoint (over Apple Music, Google Music and Amazon Music) due to its extensive music library, with over 35 million songs (Casey, 2019), in addition to providing a vast amount of audio analysis and song data on every song in that library; This audio analysis was integral in providing a fully multimodal solution, i.e. lyrics analysis plus audio analysis, to the problem of inaccurate music mood classification.

Spotify ensure that any application that connects to it does so from a 'Premium' account, meaning a paid subscription account to the Spotify service. Due to a series of problems in doing this, connecting the Mood Magic Therapy app to Spotify's API service was one of the most challenging parts of this project.

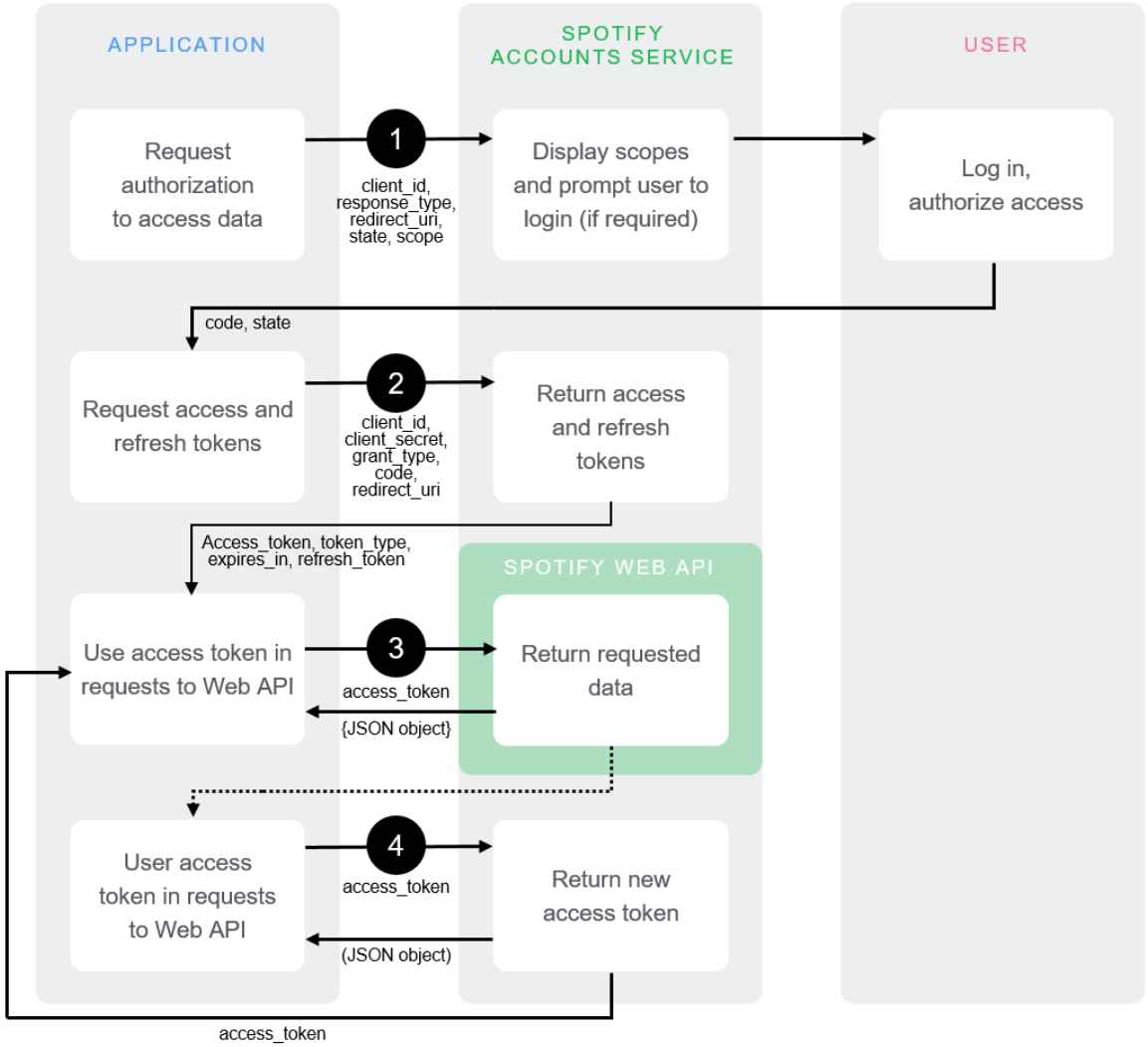


Figure 3: Authorisation Code Flow (source: (Spotify, 2019a)*)*

In order to connect the application to a Premium Spotify account, the project chose to follow the Authorisation Code Flow. Of the entire Authorisation options, this code flow required minimal user activity and allowed for continuous authorisation as long as a new Access token was used as the OAuth code before it expired. The first step in following the authorisation code flow was to create a 'Spotify app' on the Spotify developer console which would act as the Spotify API endpoint and provide the Client Id and Client Secret codes. Although this was created and used successfully, Spotify's inability to allow new app submissions to the public amounted to one of the largest limitations of the project; This is discussed in the Evaluation and Future Development section (Section 7). The second step was to host a webpage via the Firebase web-hosting service which enforces that the user follows a one-off sign in via the Spotify interface. For this, an online template was used (Garnier, 2016); This template held the Spotify 'html' page for redirecting the user to along with a couple of functions for use in the Google Cloud which requested Access Tokens and Request Tokens to be used in the Webhook. This template in addition to the 'spotify-web-api' Node.js wrapper was used to simplify the connectivity process but it may have made it more complex than it would have been to write the requests; Due to difficulties, many of the requests *were* written.

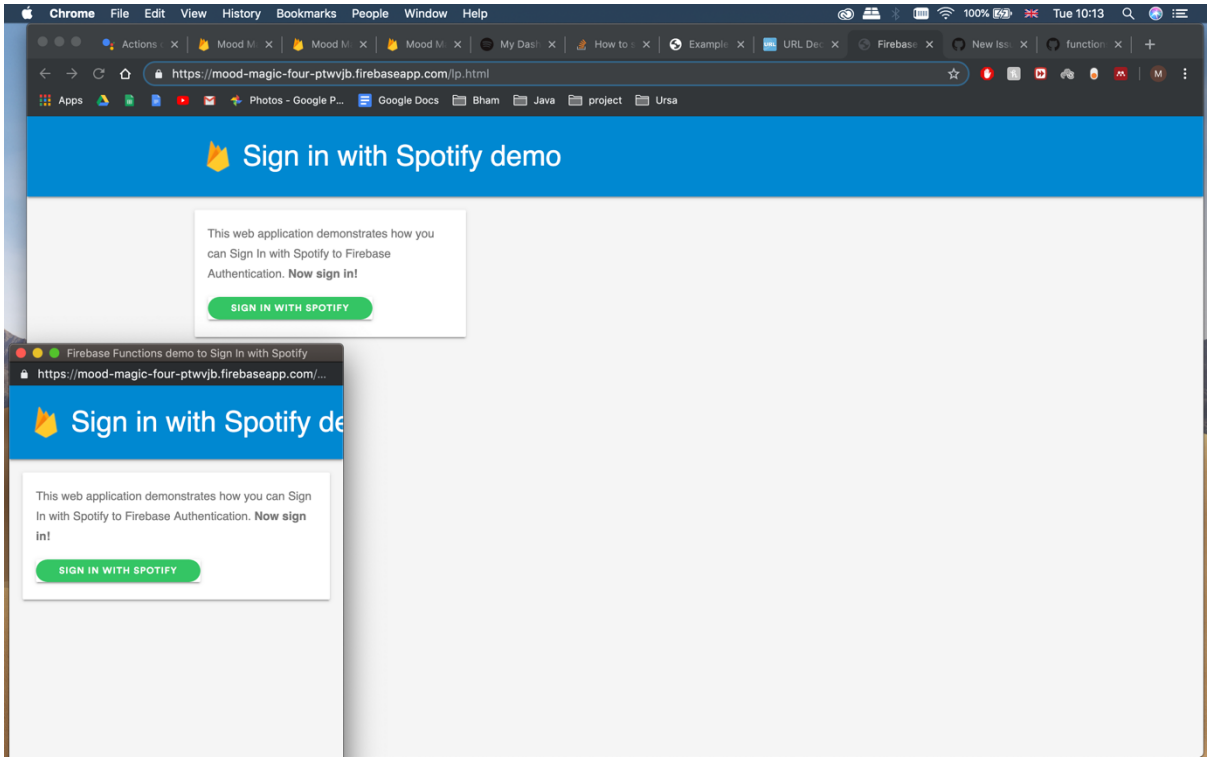


Figure 4: Spotify Sign In Page Redirect Uri issues – redirects back to page

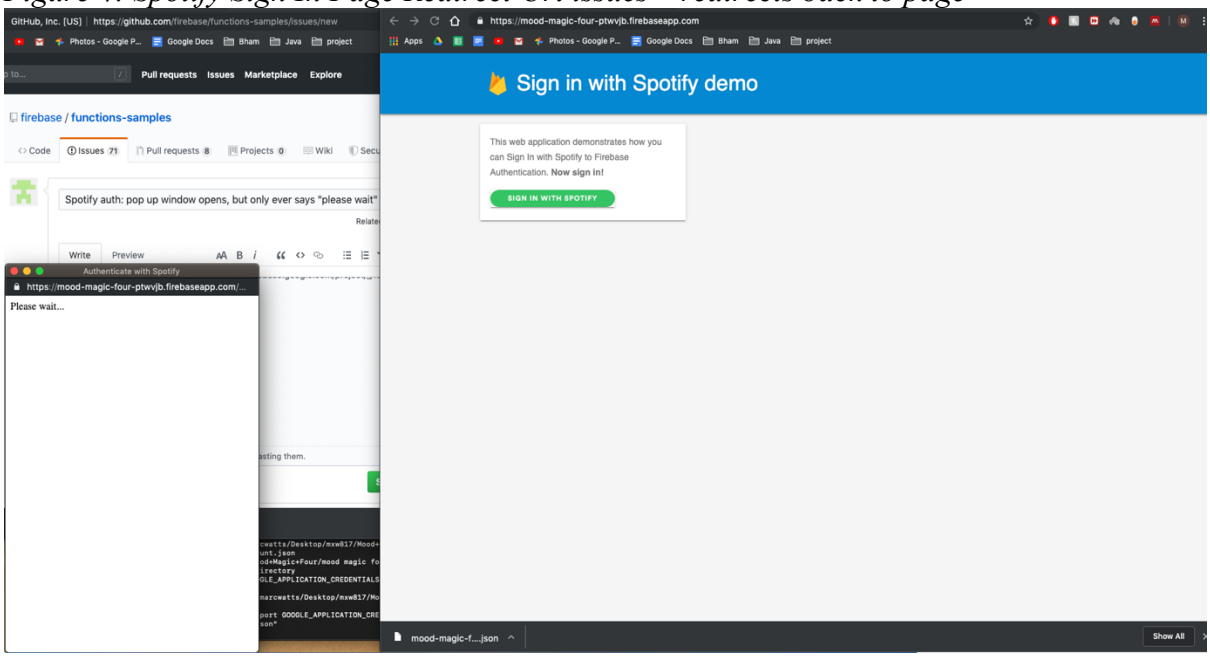


Figure 5: Spotify Sign In Page Redirect Uri issues- endless please wait page

Unfortunately, both the template and the 'Spotify-web-api' methods failed to function correctly throughout the project. The greatest challenge came from the 'spotify.authorizationCodeGrant' method; This method was supposed to send the authorisation code to the 'https://accounts.spotify.com/api/token' endpoint, which it did correctly; The endpoint would then respond with the Access and Refresh tokens. Depending on which redirect URI was used, however, the user would be taken back to the original page with no Spotify sign in whatsoever, or pass the sign in stage, only to be stuck in an endless 'please wait' page. In the latter scenario, the authorisation code was found to be 'invalid' which created a '400 Bad Request Error'. This was mysterious since the exact authorisation

code which was logged to the console was found to be correct as shown by numerous tests to the Spotify Token API endpoint; This was shown by using the HTML request structuring and testing facility, 'Postman'. Without securing the Access and Refresh tokens, the application could not progress since these were needed to utilise the Spotify API functions. In some but not all cases, however, I found by checking the Firebase Console Logs that the functions for finding the Access token and Refresh tokens were in fact retrieving the correct codes despite this error with the page not loading. The Access token, provided by the failing code, was tested in its ability to grant access to the Spotify API endpoints using cURL commands on the Mac Terminal. The exact commands were replicated to the correct structure necessary (Spotify, 2019a) and it was discovered that they did indeed allow for the retrieving of audio analysis and playing music through Spotify. Although this proved that the application was fulfilling the correct requeest procedures, for usability issues and to allow users to sign in with ease, it was essential that the never-ending 'please wait' page be fixed.

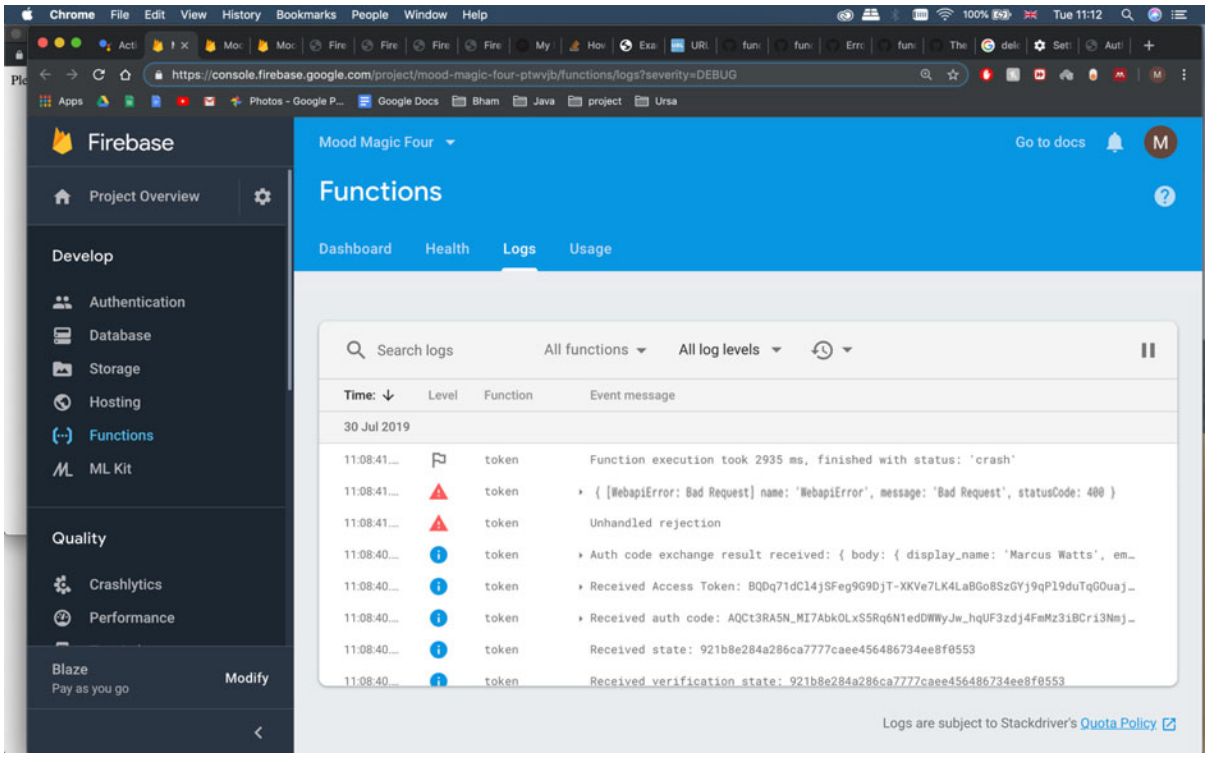


Figure 6: Firebase Console logs – Despite Crashing, Access and Refresh Tokens received

To solve the problem with the page not loading, after trialling every possible fix and reading multiple accounts of other developers facing similar issues with the tempermental API endpoint (Mdooxey06, 2017), it was concluded that the best way to tackle the problem was to start the project code again from the beginning so that the issue could be isolated and fixed. Since the project code at this point was already relatively complex, this would allow for a more accurate view of exactly what was not functioning. A new project was created with the template at the basis. Once the template and API endpoints were functioning correctly, the rest of the project was successfully added in installments.

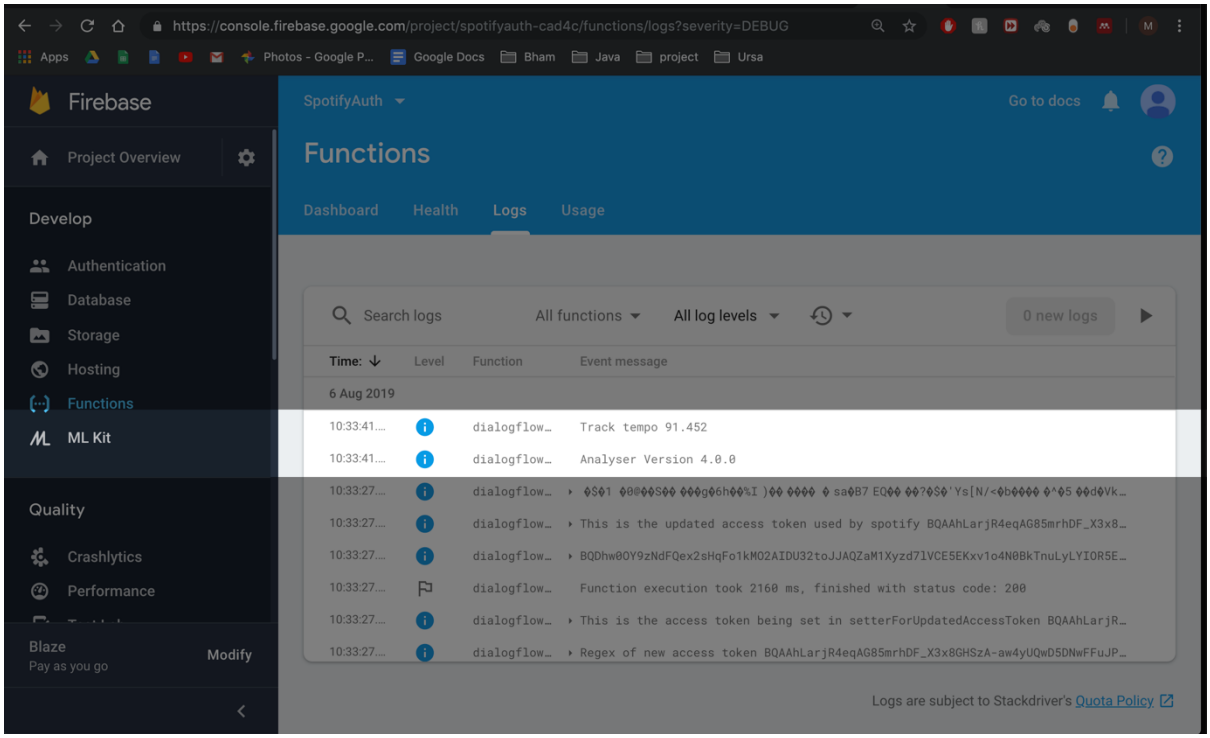


Figure 7 : Firebase logs showing successful retrieval of audio analysis

At this stage, with the successful Spotify app linking and correct API requests, the application was able to use the audio analysis from Spotify's API and log the findings to the Firebase console. The template used, however, failed to request new Access tokens before the allotted expiry time, which was around 60 minutes. In order to do this without suffering the same issues from the previously used 'spotify.authorizationCodeGrant' method, POST requests to the token endpoint were made following Spotify API guidelines (Spotify, 2019a); These will be discussed in more detail in the Implementation section. Regarding the system design, this requirement led to the design decision that, for every time an intent is triggered which maps to the action of reccommending a song (such as in 7a in Figure: 34), the application will send a POST request using the original Refresh token in return for a new Access token. This would allow the user to always hold the updated Access token necessary to get the Spotify service they required without running the risk of it expiring before performing the action – whether that means playing a song recommendation, or retrieving the song's audio analysis or feature analysis.

The last remaining objective, which also provided a challenge in the design of the system, was to actually make the agent 'speak' using the received audio analysis. Up to this point, the retrieved analysis was only shown on the log of the Firebase console despite the Webhook code directing the agent to 'say' it. This occurred due to Dialogflow's policy of timing out after 5 seconds of inactivity in its fulfilment. Originally, the intent which recommended a song and provided its audio analysis was situated within the same fulfilment intent. To allow for this 5 second timeout rule, it was necessary to make the API request to Spotify in the original intent, but instead of the agent 'saying' it, the resulting data analysis would be stored in class variables. This could then be retrieved from the class variables later down the conversation flow in the form of another intent. Thus, from the design perspective, the conversation in Dialogflow was lengthened by asking the user whether they would like to hear the audio analysis. Before they answer, the API request would already have been made

and will be stored within the class variables. A ‘yes’ response would trigger the code for the intent which retrieves this information from the class variables and the agent would ‘say’ the result to the user; This is shown in the Use Case diagram (see Figure 1). The following Implementation section will detail how these design choices manifested themselves from within the Webhook code.

5. Implementation

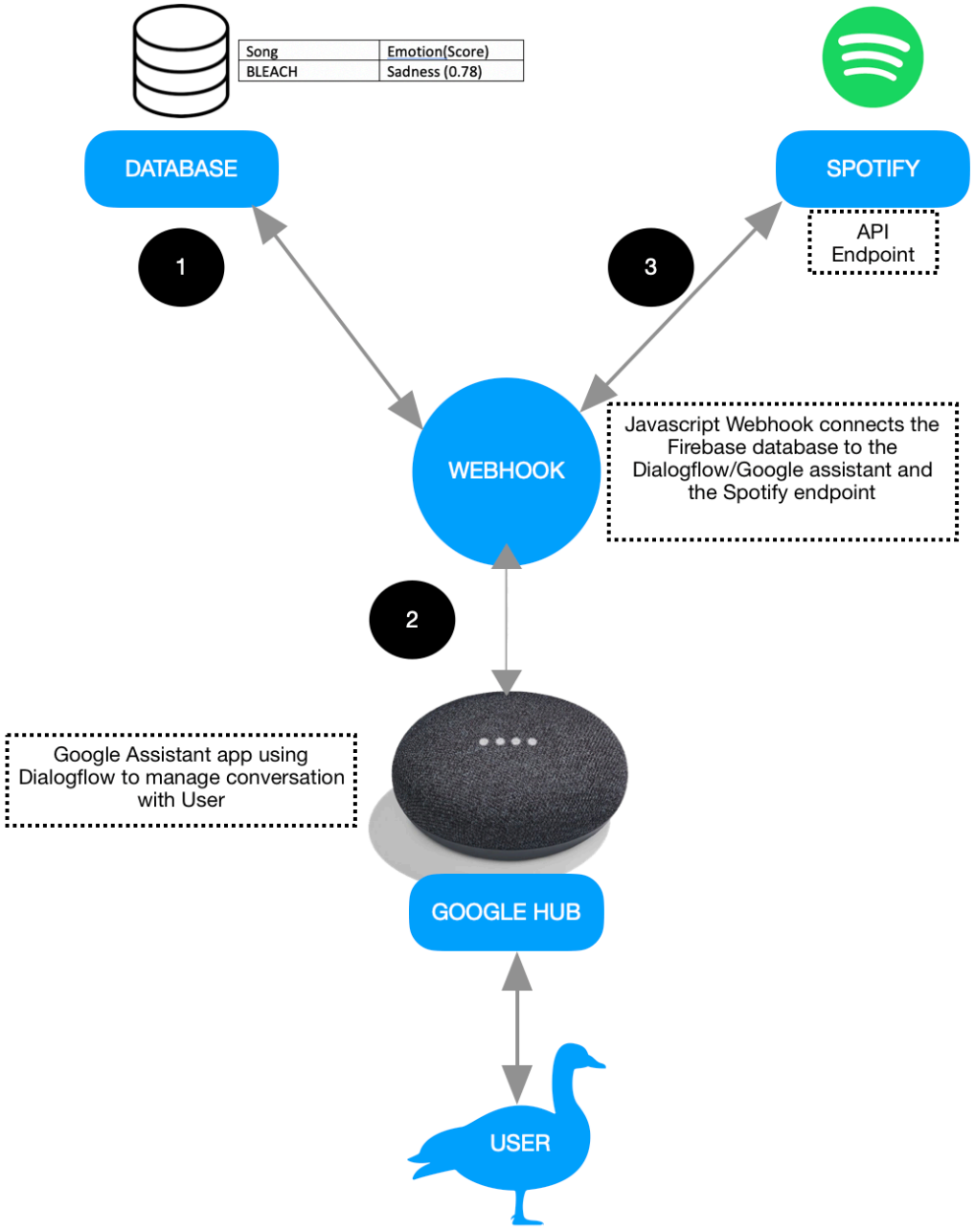


Figure 8: A High Level Diagram of the System Architecture showing the Order of Completed Stages

5.1 Overview

The implementation of the system took three distinct stages (see Figure 8); First, the database of sentiment tone analysed songs had to be created. Second, the Google Assistant application needed to be created to allow the user to interact with the system. Third, the 'Webhook', which connected the Google assistant app to the database, had to establish an API connection with the Spotify web app; This would allow the system to actually play the music recommendations in addition to providing further audio analysis. This section shall detail the implementation process of these three distinct stages.

5.2 Creating The Database

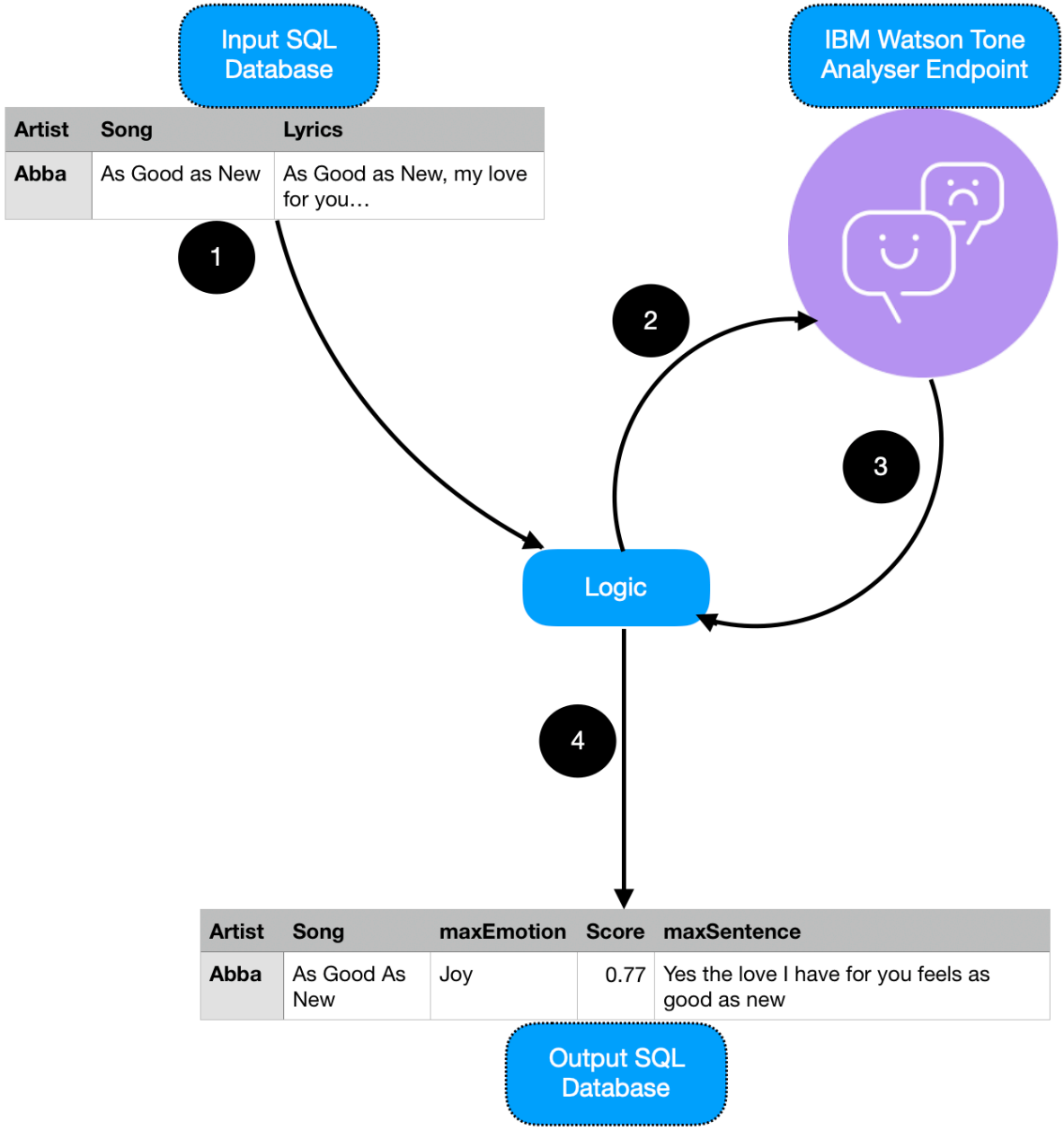


Figure 9: A Lower Level Diagram showing the Steps in Creating the Pre-Tone-Analysed Database using Watson

This project started by taking the 55,000 web scraped songs, with song titles, artist names and song lyrics from the 2019 project by Saluja, Jain, and Yadav (https://www.kaggle.com/mousehead/songlyrics#songdata.csv). This decision was taken over producing a new web scraped database since this '.csv' file contained all the data necessary for the project; It seemed unnecessary to replicate it with a new database for this reason. The first objective in creating the database was to use this data to construct an equivalent SQL database. The data required cleaning since there were commas within the lyrics field which meant that importing the data from CSV to SQL was not possible; Since the import command scanned the text for commas which designated the end of a field. There were also newlines within lyrics that needed to be removed since the import execution scanned for the newline keyboard key '\n' to designate a new row in the table. In addition to this, there were numerous spelling mistakes and randomly placed symbols which exacerbated the already hostile importing execution command for such a large database. This made this first objective reasonably challenging especially in the cases where there were clearly import errors due to the table not receiving all of the data yet there were no command line errors, since nothing was flagged up to the MySQL or Terminal logs.

A further problem with this stage was the not particularly user-friendly options for importing this data. After following the MySQL Workbench guides on using the import command, the error: 'ERROR 1148: THE USED COMMAND IS NOT ALLOWED WITH THIS MYSQL VERSION' would present itself. This was due to a default setting in Mac users' machines and many of the online solutions to it, which required a relatively high degree of Operating System prowess, failed to solve it. After a lengthy time in searching for an alternative that worked, the answer came from within a www.youtube.com video in French (MySQL Fix ERROR 1148: The used command is not allowed with this MySQL version - YouTube, 2018). This video held the necessary command syntax for changing the 'local_infile' value from 'OFF' to 'ON' in addition to the syntax for logging back into MySQL with this setting set as the default. With this error solved, there just remained some minor cleaning issues in addition to the need for placing full stops in the lyrics at suitable points to allow for the Watson Tone Analyser to analyse the text at both a sentence level and at the entire text level. This presents a limitation of this project's ability to analyse the tone used in song lyrics since it is uncertain exactly where in the lyrics the artist originally desired a full stop; This will be discussed further in the Evaluation and Future Development section (Section 7). All of the data cleaning was completed from within the 'Excel', 'Number' and 'TextEdit' applications via the 'Find and Replace' options.

```
const mysql = require('mysql');

const con = mysql.createConnection({
   host: 'localhost',
   user: 'root',
   password: '<mySQLpassword>',
   database: 'songdata'
});

con.connect((err) => {
   if (err) throw err;
   console.log('Connected!');
});

for (let index = 13; index < 50000; index+= 500 ) {
   con.query(`select artist, song, text, id FROM midAugustTableWithSentences WHERE
id = ${index}`, (err, lyrics) => {
```

```
      if (err) {
         console.log(err);
         return;
      }
      let justLyrics = lyrics[0].text;
      let rowLyrics = JSON.stringify(lyrics[0].text);
      let id = JSON.stringify(lyrics[0].id);
      let artist = JSON.stringify(lyrics[0].artist);
      let song = String(JSON.stringify(lyrics[0].song));
```

Figure 10: SQL connection and query code

The project followed the Node.js library, 'mysql', for connecting to the database; This entailed hosting on the localhost and connecting to it via a 'con.connect' (w3schools, 2019). The code was then written which queried the table, row by row using a 'for loop', storing the results of the queries in variables (see Figure 10).

```
const ToneAnalyzerV3 = require('watson-developer-cloud/tone-analyzer/v3');

const toneAnalyzer = new ToneAnalyzerV3({
   version: '2017-09-21',
   iam_apikey: 'GMmCSwVGk3hafuqlqE_BqosWt5fXzwhcrgBWqoLTpNpD',
   url: 'https://gateway-lon.watsonplatform.net/tone-analyzer/api'
});
const toneParams = {
      tone_input: { text: rowLyrics },
      content_type: 'text/plain',
      sentences: true
   };
   toneAnalyzer
      .tone(toneParams)
      .then((toneAnalysis) => {
         console.log(JSON.stringify(toneAnalysis, null, 2));
```

Figure 11 : Connecting to the IBM Watson API and sending the lyrics variable

The variable containing the row's lyrics is then sent to the Watson Tone Analyser endpoint (see Figure 11). The greatest challenge during this stage of the project was ensuring that the endpoint connection was indeed made. After trialling numerous walkthroughs of other projects which used the IBM Watson Tone Analyser, including those by Opesanya (2018) and Perrin (2018), it was found that IBM had recently changed their endpoint addresses and structures. The only way to solve this, since there were no other more recent tutorials, was to follow the API documentation on the IBM website itself. Once the endpoint service was established on the IBM cloud and the code was structured exactly as directed by the IBM website (IBM, 2017), the connection could be made between the locally hosted Node.js Javascript code, the MySQL database and IBM's Watson Tone Analyser.

```
         /**
          * Find the highest score for Document Tone and set that to max along with
the associated emotion:
          */
         if (JSON.stringify(toneAnalysis.document_tone.tones[0] !== undefined)) {
            var max = JSON.stringify(toneAnalysis.document_tone.tones[0].score);
            var maxEmotion =
JSON.stringify(toneAnalysis.document_tone.tones[0].tone_id);

            for (let i = 0; i < toneAnalysis.document_tone.tones.length; i++) {

               if (max < JSON.stringify(toneAnalysis.document_tone.tones[i].score))
{
```

```
                max = JSON.stringify(toneAnalysis.document_tone.tones[i].score);
                maxEmotion =
JSON.stringify(toneAnalysis.document_tone.tones[i].tone_id);
              }
            }
```

Figure 12 : The highest recorded scoring emotion was found in the song

To find the highest scoring emotion in the song, it was necessary to loop over all the possible emotions and their respective scores found in the resulting JSON response from the Watson Tone Analyser. To do this, a 'for loop' was implemented which looped over the number of all the resulting tone scores. The maximum recorded score was updated at every loop iteration in the result of the emotion score being higher than the previous score. This was stored inside the 'max' variable, along with its respective score in the 'maxEmotion' variable (see Figure 12).

```
            /**
             * Find the highest scoring sentence in the song of the same emotion
(as maxEmotion):
             * @type {string}
             */
            if (JSON.stringify(toneAnalysis.sentences_tone[0] !== undefined)) {
               var maxSentenceScoreForCurrentEmotion = 0;
               var maxSentenceSentenceForCurrentEmotion = '';


               // Loop through every separate sentence
               for (let j = 0; j < toneAnalysis.sentences_tone.length; j++) {
                  //Loop through every sentence's scores
                   console.log('There are ' +
toneAnalysis.sentences_tone[j].tones.length + ' scores for this sentence.' );

                  if (JSON.stringify(toneAnalysis.sentences_tone[j].tones.length) >
0) {
                      for (let k = 0; k <
toneAnalysis.sentences_tone[j].tones.length; k++) {

                        console.log('These are the individual scores for each
sentence: ' + JSON.stringify(toneAnalysis.sentences_tone[j].tones[k].score));
                       if
(JSON.stringify(toneAnalysis.sentences_tone[j].tones[k].tone_id) === maxEmotion &&
toneAnalysis.sentences_tone[j].tones[k].score !== undefined){
                          if
(maxSentenceScoreForCurrentEmotion<toneAnalysis.sentences_tone[j].tones[k].score) {
                             maxSentenceScoreForCurrentEmotion =
toneAnalysis.sentences_tone[j].tones[k].score;
                             maxSentenceSentenceForCurrentEmotion =
toneAnalysis.sentences_tone[j].text;
                          }
                       }


                  console.log('This is the max score and corresponding text for the
current emotion:');
                  console.log(maxSentenceScoreForCurrentEmotion);
                  console.log(maxSentenceSentenceForCurrentEmotion);
               console.log(maxEmotion);

               if (maxSentenceScoreForCurrentEmotion>0) {

                  var sql = `INSERT INTO midAugustAnalysedSongsOutput (artist,
song, maxEmotion, score, maxSentence, maxSentenceScore) VALUES ( ${artist},
```

```
${song}, ${maxEmotion}, ${max},
${JSON.stringify(maxSentenceSentenceForCurrentEmotion)},
${maxSentenceScoreForCurrentEmotion})`;
                  con.query(sql, (err, result) => {
                     if (err) throw err;
                     console.log(`1 record ( ${max} for ${maxEmotion})inserted for
${song} by ${artist}`);
                  });
               }
               }
            }

      })
      .catch((err) => {
         console.log('error:', err);
      });
});
```

Figure 13: Finding the highest scoring sentence of the same highest scoring emotion for the song

The final stage in utilising the Watson Tone analysis was to find the sentence, of the same emotion as that in 'maxEmotion', which scored the highest in the entire song. This involved using a 'double for loop' – one for the list of all the highest scoring sentences and another for retrieving the individual scores for those sentences. The structure of this code followed the same 'find maximum' pattern from the method to find the highest scoring emotion above. The resulting JSON is shown below for the song, 'Disillusion' by 'ABBA', as an example of this code being implemented:

Lyrics sent to Watson:
Changing moving in a circle I can see your face in all of my dreams Smiling laughing from the shadows When I hear your voice I know what it means I know it doesn't matter just how hard I try You're all the reason for my life . Disillusion disillusion's all you left for me How can I forget you when my world is breaking down You're all I had you're all I want Disillusion disillusions now that's all I have . Wishing hoping chasing shadows Did I see your face somewhere in the crowd Thinking wondering what you're doing I can't stop myself from crying out loud They say my wound will heal and only leave a scar But then they never shared our love . Disillusion disillusion's all you left for me How can I forget you when my world is breaking down You're all I had you're all I want Disillusion disillusions now that's all I have . Disillusion disillusions now that's all I have.


```
{
 "document_tone": {
  "tones": [
   {
    "score": 0.628372,
    "tone_id": "sadness",
    "tone_name": "Sadness"
   },
   {
    "score": 0.9215,
    "tone_id": "confident",
    "tone_name": "Confident"
   }
  ]
 },
```

```
  "sentences_tone": [
    {
      "sentence_id": 0,
      "text": “Changing  moving in a circle I can see your face in all of my dreams Smiling laughing from the shadows When I hear your voice  I know what it means I know it doesn't matter just how hard I try You're all the reason for my life  .",
      "tones": [
        {
          "score": 0.601682,
          "tone_id": "joy",
          "tone_name": "Joy"
        }
      ]
    },
    {
      "sentence_id": 1,
      "text": "Disillusion  disillusion's all you left for me How can I forget you when my world is breaking down You're all I had  you're all I want Disillusion  disillusions  now that's all I have  .",
      "tones": [
        {
          "score": 0.688539,
          "tone_id": "sadness",
          "tone_name": "Sadness"
        },
        {
          "score": 0.892415,
          "tone_id": "confident",
          "tone_name": "Confident"
        }
      ]
    },
    {
      "sentence_id": 2,
      "text": "Wishing  hoping  chasing shadows Did I see your face somewhere in the crowd Thinking  wondering what you're doing I can't stop myself from crying out loud They say my wound will heal and only leave a scar But then  they never shared our love  .",
      "tones": [
        {
          "score": 0.678322,
          "tone_id": "sadness",
          "tone_name": "Sadness"
        }
      ]
    },
    {
      "sentence_id": 3,
      "text": "Disillusion  disillusion's all you left for me How can I forget you when my world is breaking down You're all I had  you're all I want Disillusion  disillusions  now that's all I have  .",
```

```
    "tones": [
      {
       "score": 0.688539,
       "tone_id": "sadness",
       "tone_name": "Sadness"
      },
      {
       "score": 0.892415,
       "tone_id": "confident",
       "tone_name": "Confident"
      }
    ]
   },
   {
    "sentence_id": 4,
    "text": "Disillusion  disillusions  now that's all I have.",
    "tones": [
      {
       "score": 0.92125,
       "tone_id": "confident",
       "tone_name": "Confident"
      }
    ]
   }
  ]
}
```


This is the max score and corresponding text for the current emotion:
0.92125
Disillusion disillusions now that's all I have.
"confident"
1 record (0.9215 for "confident")inserted for "Disillusion" by " ABBA"
Figure : Example of response JSON from the Watson Tone Analyser

The fourth and final part of implementing the database for this project, was to output the necessary results of the Watson Tone Analyser response into a new SQL database. This required creating the table and inserting the artist, song, the highest scoring general emotion in the song with its score and the highest scoring sentence of that particular emotion along with its score. The 'mysql' library was used in a similar way to the query for the input table, only this time, the results were outputted into a new table within the same database. The input and outputted tables are shown below.

	artist	song	text	id
▶	ABBA	She's My Kind Of Girl	Look at her face it's a wonderful face And it means something special to me Look at the way that…	1
	ABBA	Andante Andante	Take it easy with me please Touch me gently like a summer evening breeze Take your time mak…	2
	ABBA	As Good As New	I'll never know why I had to go Why I had to put up such a lousy rotten show Boy I was tough pa…	3
	ABBA	Bang	Making somebody happy is a question of give and take You can learn how to show it so come on…	4
	ABBA	Bang-A-Boomerang	Making somebody happy is a question of give and take You can learn how to show it so come on…	5
	ABBA	Burning My Bridges	Well you hoot and you holler and you make me mad And I've always been under your heel Holy…	6
	ABBA	Cassandra	Down in the street they're all singing and shouting Staying alive though the city is dead Hiding the…	7
	ABBA	Chiquitita	Chiquitita tell me what's wrong You're enchained by your own sorrow In your eyes there is no ho…	8
	ABBA	Crazy World	I was out with the morning sun Couldn't sleep so I thought I'd take a walk I was thinking of you a…	9
	ABBA	Crying Over You	I'm waitin' for you baby I'm sitting all alone I feel so cold without you It chills me to the bone I nev…	10

Figure 14: Input table of songs pre-Watson-Tone-Analysis

artist	song	maxEmotion	score	maxSentence	maxSentenceScore
Who	The Quiet One	sadness	0.606601	Still waters run deep so be careful I don't drown you You've got nothing to he…	0.62098
Ingrid Michaelson	Girls Chase Boys	confident	0.911386	All the broken hearts in the world still beat Lets not make it harder than it has…	0.717802
Alice Cooper	I Just Wanna Be God	tentative	0.882904	I'm just trying to be God I only want to be God I just want to be God Why can…	0.849846
ABBA	She's My Kind Of Girl	tentative	0.987022	She's just my kind of girl she makes me feel fine Who could ever believe th…	0.984352
Zeromancer	Famous Last Words	sadness	0.669059	Tar for feathers Blood for honey Milk for money Isn't it funny how it hurts Ho…	0.75937
Ugly Kid Joe	Busy Bee	tentative	0.815134	Some people they just ain't satisfied Watch the world go by .	0.957828
Deep Purple	Paint It Black	joy	0.594117	If I look hard enough into the setting sun My love will laugh with me before t…	0.864903
Judds	In My Dreams	joy	0.64506	In my dreams My soul is wide awake Craving the love we make In my dream…	0.792499
Cheap Trick	Who D'king	tentative	0.939202	Don't try to please me You just give me idle conversation Doesn't give me an…	0.953512
Elvis Presley	A Big Hunk O' Love	tentative	0.851683	I got a wishbone in my pocket I got a rabbit foot around my wrist I'd have all…	0.763337

Figure 15 : Output table of songs post-Watson-Tone-Analysis

Due to the restrictions on the number of api calls which can be made to Watson's Tone Analyser for the standard 'free' account, only a select number around 150 songs were used in the final database. Once the outputted table had been cleaned further – for instance, changing 'Who' to 'The Who' in the tables above, and the Spotify track URI's were added, the table was converted to 'CSV' format. It was then converted from 'CSV' to 'JSON' using https://codebeautify.org (codebeautify, 2016) and validated using https://jsonlint.com/ (JSONLint, 2019). Once validated and encoded in UTF-8 (as per the Firebase requirements), the table was imported into the Firebase Realtime Database.

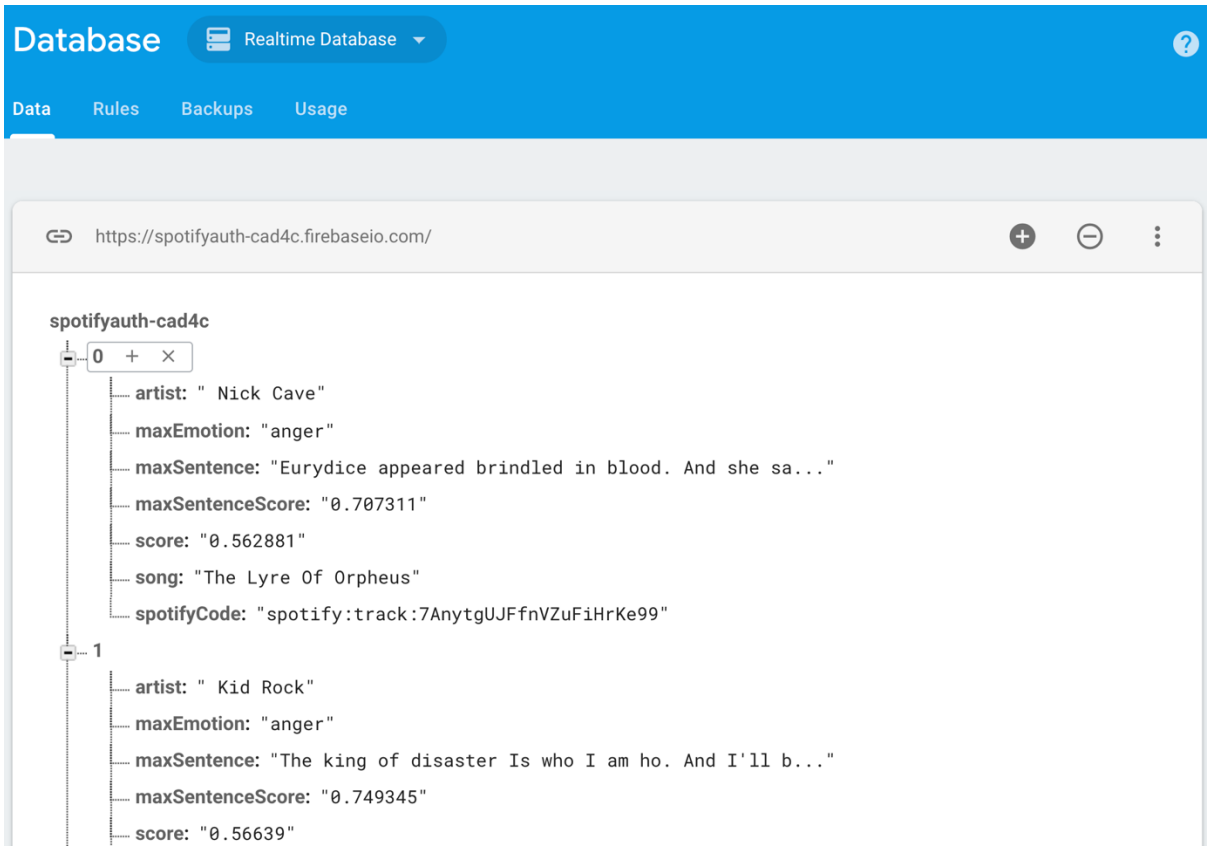


Figure 16 : Firebase Realtime Database after the tone analysed songs table import

5.3 Creating the Google Assistant Application

The Google Assistant application implementation process can be divided into two stages: 1. Create the conversational application using the Actions on Google Console and Dialogflow. 2. Write the fulfilment code using the Google cloud hosted Webhook.

5.3.1 Creating the Conversational App

5.3.1.1 Launching the App

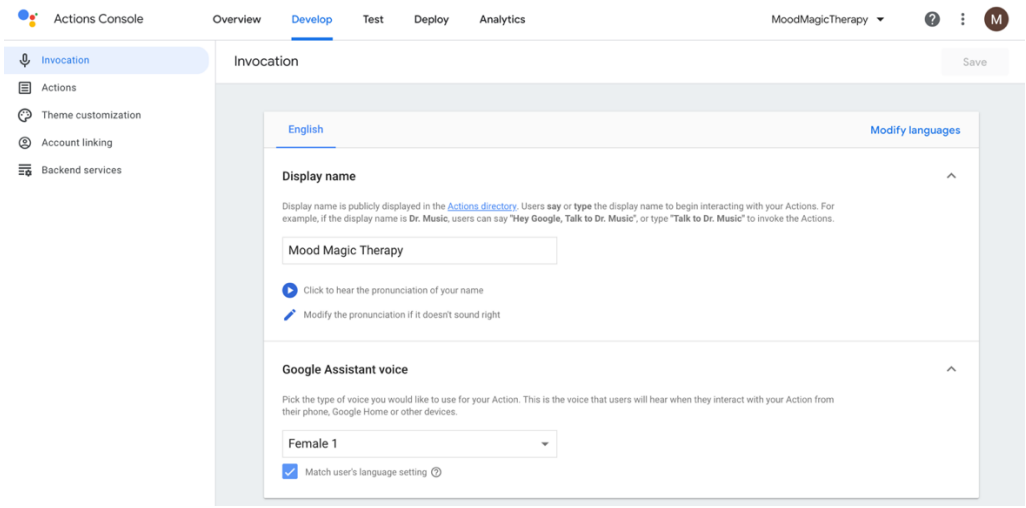


Figure 17 : Setting the invocation phrase to start the application

The first step was to create the Google Assistant application. This involved navigating through the Actions on Google console to set up the application's invocation phrase. To open the application, users must clearly state: "Talk to Mood Magic Therapy". Once open, the application will execute the 'action'.

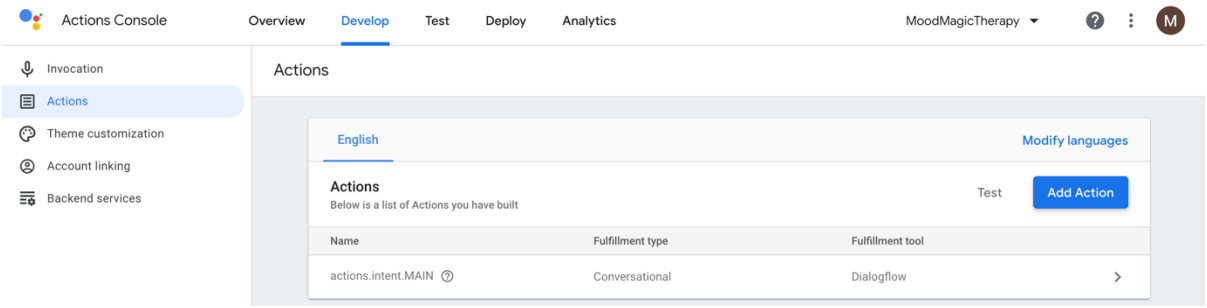


Figure 18: The conversational 'action' for the application

The action was set to 'conversational' style and, as such, was handled by Google-owned Dialogflow. Dialogflow is integrated with Google 'actions' and provides the natural language and artificial intelligence capabilities to be able to extract the useful parameters from spoken text and use them to generate particular response (Google, 2019c). 'Intents' are what handle these particular responses.

5.3.1.2 Intents

Figure 19: Dialogflow Intents

When the user utters the invocation phrase, "Ok Google, talk to Mood Magic Therapy", Dialogflow automatically executes the 'DefaultWelcomeIntent'. It is inside this intent that the Google agent will vocalise a greeting to the user along with a question asking how they are feeling. For example, the Google agent will say: "Hello! How are you feeling?". The user will then respond, and Dialogflow will examine the user's response, extracting any and all necessary parameters. Since this project focuses on four emotions: 'happy', 'sad', 'confident' and 'angry', four intents of the same name were integrated into the Dialogflow structure. The parameters that Dialogflow will be checking for are any words which closely match these four emotions.

5.3.1.3 Entities

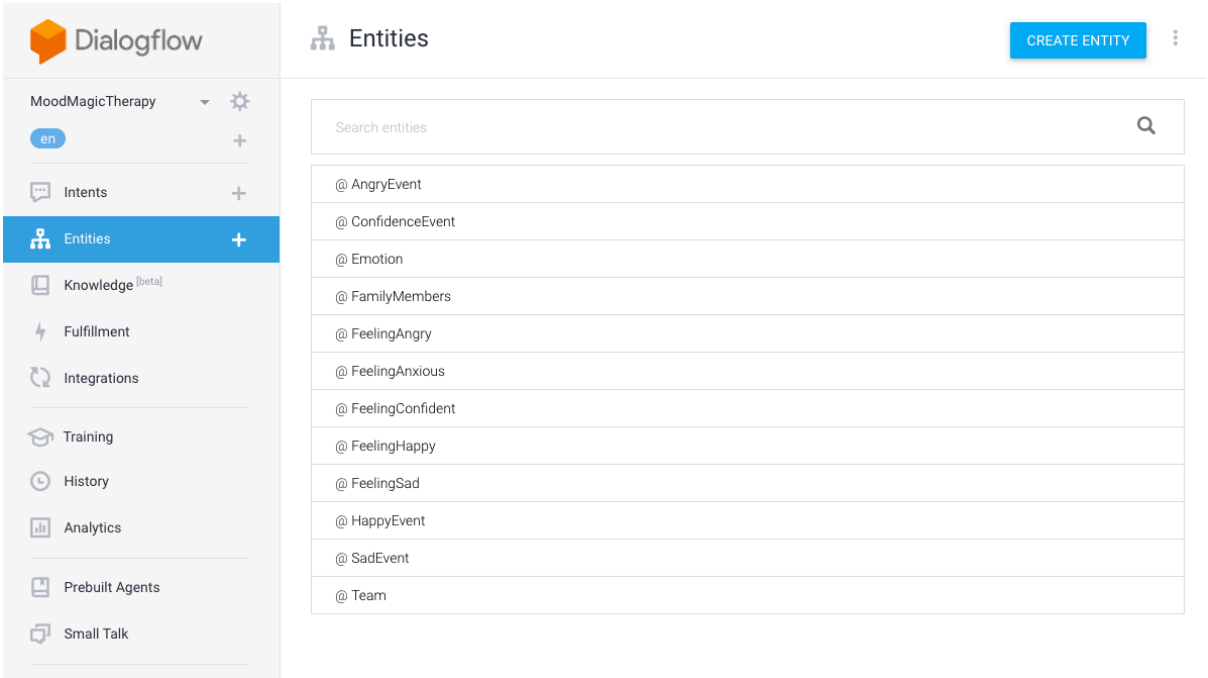


Figure 20: Dialogflow Entities

Figure 21: FeelingAngry Emotion

To match the parameters to the correct intent, 'Entities' are used. Entities constitute the 'type' of data that the parameter falls under (Google, 2019d). There are 'System Entities', such as 'dates' - for instance, a user saying "tomorrow" will fall under the 'date' 'System Entity', and 'Custom Entities'. In this project, 'Custom Entities' have been created to account for the four emotions. Inside each emotion entity resides a multitude of different phrases spoken when feeling that particular emotion (see Figure21).

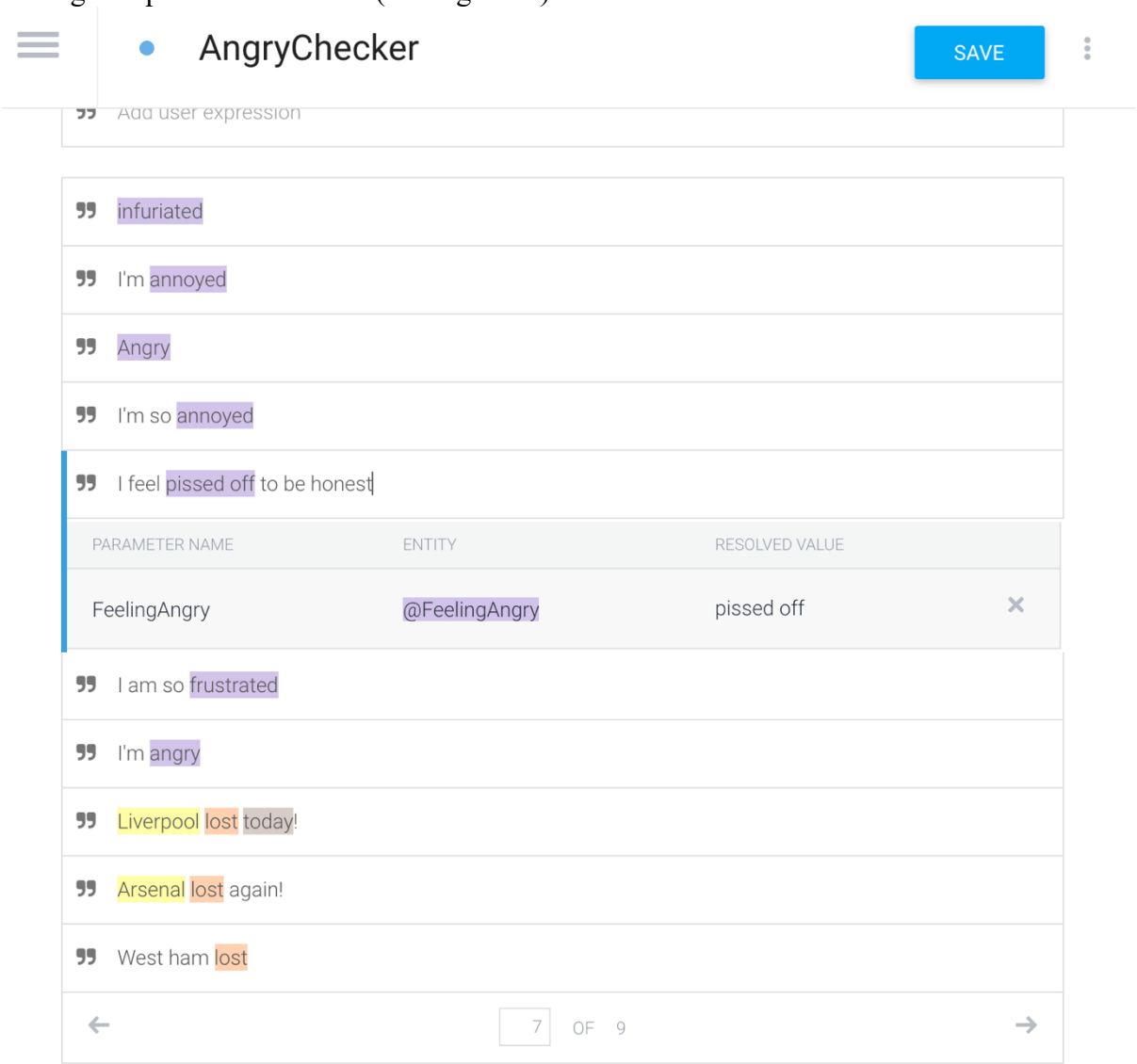


Figure 22: AngryChecker Intent

If the user, therefore, responds to the 'DefaultWelcomeIntent' with "I feel pissed off to be honest", the phrase 'pissed off' is mapped to the 'FeelingAngry' entity which is a necessary parameter in the 'AngryChecker' intent. Likewise, phrases constituting other emotion entities will direct the user towards that particular emotion checker. It is also worth mentioning that the system design chosen for invoking a confidence emotion included both the user speaking about being confident in addition to feeling anxious since it was concluded that in both scenarios, a confidence song would be most appropriate; This led to the designing of both 'feelingConfident' and 'feelingAnxious' entities (see yellow highlights in Figure 24 and 25).

5.3.1.4 The Challenge of User Input Solved Using the 'Checker System'

One of the challenges with this section of the project was that the user may say a phrase that could be construed as being one of several emotions. Or, the user could say something which is not found among any of the entities. Following testing, it was found that the system could not account for all possible user responses in all scenarios. It was therefore concluded that the 'checker system' must be implemented to ensure user-system agreeableness. Before being designated to a specific emotion intent path, the user would be asked something like, "I'm detecting that you are feeling angry, is this correct?". The user could then either confirm or deny this. Upon confirming, the parameters they used in answer to the 'DefaultWelcomeIntent' are used in the response to the confirmation intent – '-AngryIntent'. This allows for a seamless conversation as the system will 'remember' what the user said previously through the use of 'Contexts'. For every intent, there are input and output 'contexts'; Input contexts are derived from the parent intent and output contexts are those that are sent to the input of the intent's child. This ensures that the conversational flow moves in the correct way, for example, the 'DefaultWelcomeIntent' has the output context 'DefaultWelcomeIntent-followup-2' – this is the input context of all the checker intents ensuring that the answer the user gives will execute only one of these four intents and no other intent. Parameters stored in the answer to an intent, for instance the word 'angry' in response to the 'DefaultWelcomeIntent', will exist further along the conversation for any intent which has the outputted context from the original intent as its input context (Google, 2019b). Should the user deny the checker by saying that it is *not* the correct emotion, they are taken to the intent which asks them to state which emotion out of the four options they would like a song to be in: "I'm sorry. I'm still learning. Which of the following four emotions would you like me to find a song in? Angry; Happy; Sad; or Confident?". The user must then reply with one of those four words and they will then be taken to the appropriate emotional intent.

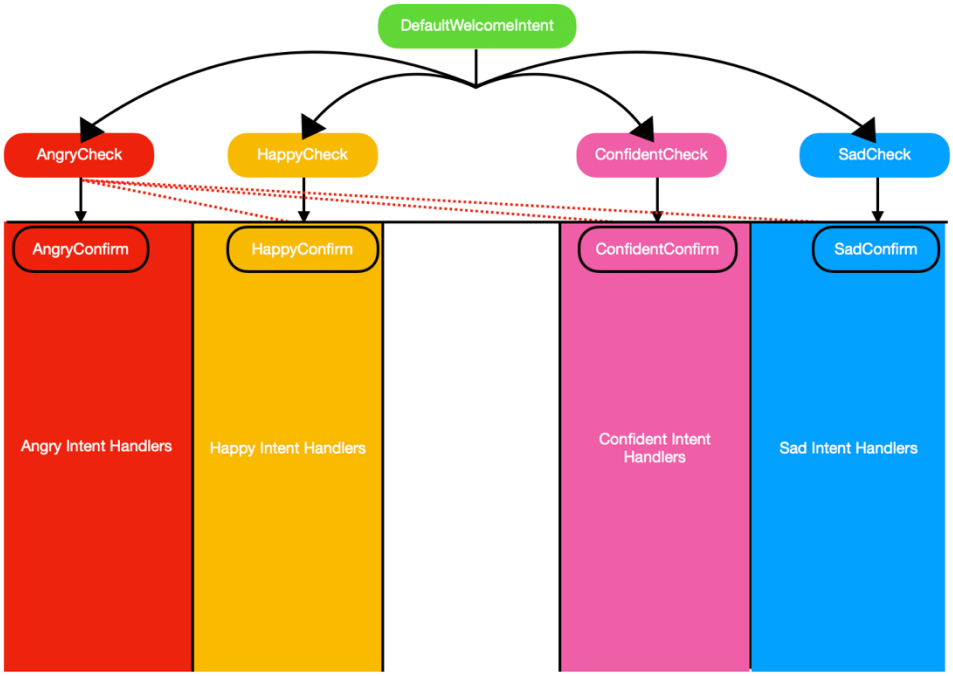


Figure 23: The Checker System

The conversational flow therefore directs the user to a specific checker intent which asks them to confirm or deny the emotion that is detected. They are then able to change the flow by jumping to another emotion to the one that is chosen (see the AngryCheck dotted lines in Figure 23 to represent changing to another emotional intent). Once they confirm this new emotion, or confirm the original chosen emotion, they are locked in to one of the four emotions and are directed through the conversation with that specific emotion. The only way to have a song with another emotion from that point is to end the conversation with the Google agent and restart the application. The checker system should ensure that the user actually wants a song in the chosen emotion, however, so this should not be an issue.

5.3.1.5 Events and Groups Entities

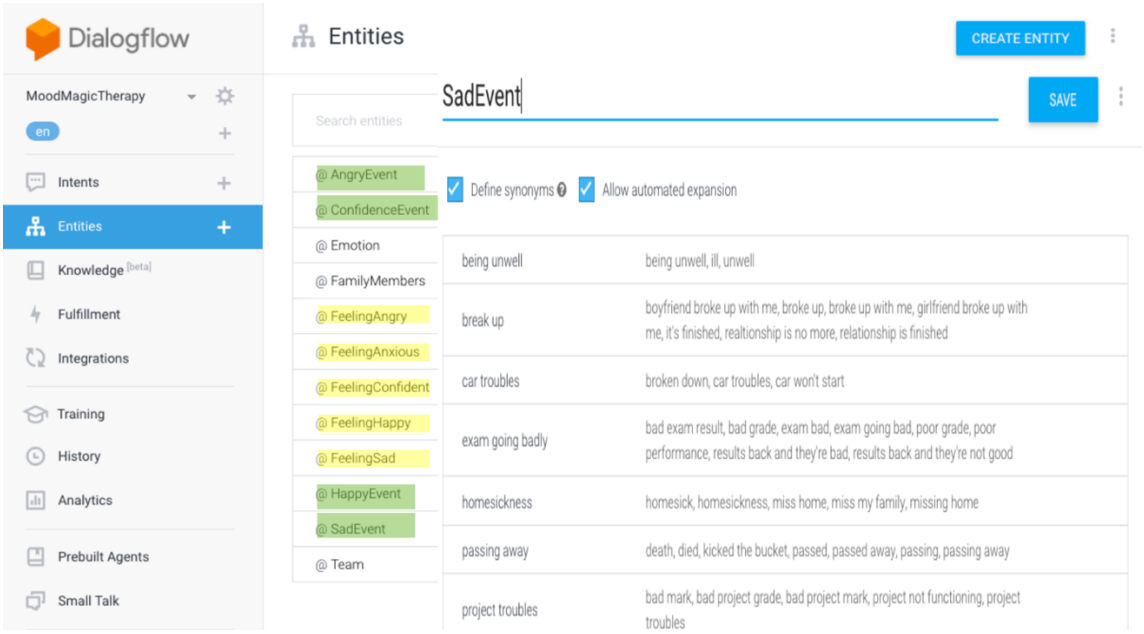


Figure 24: Event Entities (green)

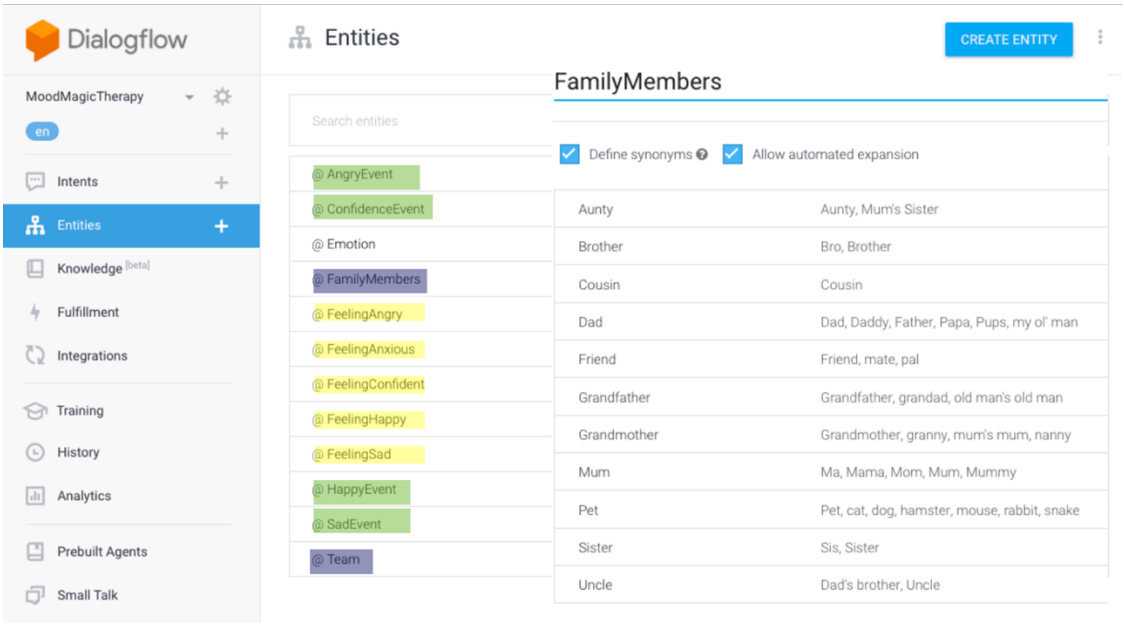


Figure 25: Group entities (purple)

Another feature used in the Dialogflow section of this project is being able to describe 'events' to the agent in reply to "How is your day going?". For instance, the user can say, "My cat just passed away". The phrase 'passed away' is registered with the 'passing away' type of 'SadEvent' and the word 'cat' is found to be a 'Pet' under the 'FamilyMembers' entity; Both of these parameters are then stored. After confirming the 'SadChecker' intent, the user will be brought to the '-Sad' intent and will receive the reply, "I'm sorry to hear about your Pet passing away. It might help if we listen to some sad music together - would you like that?". These event entities were developed in order to find situations where a user is likely to feel a certain way after an actual event in their life. They were included after initial testing found that users may want to state what is happening in their life rather than actually saying how they feel. Moreover, the system offers unique responses based upon which emotion is used. Since sadness was shown to be the most common emotion faced by users during testing, responses were inspired by an article by the Mental Health charity, HeadsTogether, which offered advice on the best responses for those suffering mental health issues (HeadsTogether, 2019). Likewise, user utterances were based on initial testing in addition to a HuffPost article which noted typical 'encoded' speech suggesting a person is facing depression (Quinn, 2018). These design choices aid the application's ability to provide a 'human' response; One of the requirements of the system.

```
const puppeteer = require('puppeteer');

(async () => {
    const extractPartners = async (url) => {
        const page = await browser.newPage();
        await page.goto(url);
        console.log('scraping:' + url);

        const partners = await page.evaluate(() =>
            document.querySelector(
```

```
                '#mw-content-text > div > table:nth-child(11) > tbody > tr:nth-
child(2) > td:nth-child(1) > a'
            )
        );
        await page.close();
        return partners;
    };
    const browser = await puppeteer.launch();

    const url = 'https://en.wikipedia.org/wiki/List_of_football_clubs_in_England';
    const artists = await extractPartners(url);
    console.log(artists);

    await browser.close();
})();
```

Figure 26: Web Scraping Wikipedia's list of Football Clubs in England

A common event which triggered mostly happy or angry emotions in users during initial testing was through their football team winning or losing. In order to accommodate a large variety of teams, the project used a 'Node.js' library, 'Puppeteer', entwined with the headless-browser version of Google Chrome to web scrape the names of around 1,000 football teams; This tutorial was followed in order to do this: https://www.youtube.com/watch?v=pixfH6yyqZk (DevTips, 2018).
The resulting teams were then imported as a JSON file into the entity 'Team' under the 'Football Team' type. This allows users to state their football team, for example 'Liverpool' and whether they 'won' or 'lost'. The term 'lost' would be denoted as an 'AngryEvent' and the team 'Liverpool' would be used as a 'Football Team' resulting in the reply: "I'm sorry to hear that your Football Team lost, would you like to listen to some angry music together to blow off some steam?". Likewise, if their team "won" this would be mapped to the 'HappyEvent', and therefore the '-Happy' intent, instead. Similar events were made for the other emotions including an event for having an important exam the following day or having a date with someone in the evening; Both of which would be used as confidence events as the user may benefit from a confidence boosting song.

The design therefore aims to replicate a real world music therapy session. It is impossible to know exactly how the user is feeling, however, even with these events occurring. The user may feel extremely upset after a relationship ending, or they may be all the more happy that the relationship is finally over. This, once again, provides another reason why the checker system is used since the system cannot be expected to 'know' exactly how the user is feeling based on words or events alone; This is discussed further in the Limitations and Future Development section. Moreover, by allowing the user to change the emotion before being locked in to a certain emotional path, the system allows for users to listen to music which is not of the same emotion that they are in. This fulfils the Use Case where users may want to transform their mood by listening to songs of a contrasting emotion (Schäfer *et al.*, 2013).

5.3.2 Creating the Fulfilment Webhook behind the Conversational App

As previously mentioned, users become fixed within one emotional intent path upon confirming the checker intents. At this point, they are asked if they would like to hear a song of that chosen emotion. A 'no' answer will end the conversation, but a 'yes' answer will

trigger the system to find and retrieve a song of the desired emotion. This is all handled via the Webhook, which is the back-end Google Cloud hosted code. As per the diagram at the beginning of the System Design section (Figure 2), Dialogflow will make a request to the Webhook service and the name of the intent is matched with the desired function.

5.3.2.1 Intent Mapping: 'playSadSong' Function

```
/**
 * intentMap maps the google actions to the various functions required for
fulfilment.
 * @type {Map<any, any>}
 */
let intentMap = new Map();
intentMap.set('-Sad - yes', playSadSong);
intentMap.set('-Sad - yes - no - no-chooseAgain', playSadSong);
intentMap.set('-Sad - yes - no - yes', playMusic);
intentMap.set('-Sad - yes - yes', giveAudioAnalysis);
intentMap.set('-Sad - yes - yes - yes', playMusic);
intentMap.set('-Sad - yes - yes - no - yes', playSadSong);
agent.handleRequest(intentMap);
```

Figure 27 : Intent Mapping

A user who wants a sad song, for instance, will create a Webhook request from answering 'yes' to the checker intent which invokes the '-Sad – yes' intent. This is matched with the 'playSadSong' function (see Figure 27). Inside this function, a random number will be generated within the bounds of the sad songs in the database. This number will be used to select a sad song and the song data will be set in the class variables.

```
admin.database().ref(`${SongInfo.randomNumber}`).once('value').then(async
(snapshot) => {
        // Get the song, artist and spotify uri (with and without the preceding
characters) from the Firebase Realtime Database
        let song = snapshot.child('song').val();
        let artist = snapshot.child('artist').val();
        let spotify_uri = snapshot.child('spotifyCode').val();
        let maxEmotion = snapshot.child('maxEmotion').val();
        let score = snapshot.child('score').val();
        let maxSentence = snapshot.child('maxSentence').val();
        let maxSentenceScore = snapshot.child('maxSentenceScore').val();
        setSong(song);
        setArtist(artist);
        setMaxEmotion(maxEmotion);
        setScore(score);
        setMaxSentence(maxSentence);
        setMaxSentenceScore(maxSentenceScore);

        setSpotify_uri(spotify_uri);
        let regex = /(?<=spotify:track:)[^"]+/gm;

        let m;
        let just_uri = '';

        while ((m = regex.exec(spotify_uri)) !== null) {
            // This is necessary to avoid infinite loops with zero-width matches
            if (m.index === regex.lastIndex) {
                regex.lastIndex++;
            }

            // The result can be accessed through the `m`-variable.
            m.forEach((match, groupIndex) => {
                just_uri = match;
```

```
                setJustUri(just_uri);
                console.log(`Found match, group ${groupIndex}: ${match}`);

            });
        }


        recommendASong(agent);



    });
}
```

Figure 28: Getting the song data from the database

The necessary variables are set including the song name, artist name, Spotify uri – both the URI code on its own through the use of a 'regex' in addition to the entire URI, max emotion, score, max sentence and max sentence score for later use in the code. Once this occurs, the 'recommendASong' function is called which forms the agent's reply to the user.

```
function playSadSong(agent) {
    /**
     * POST request to the Spotify token api for using the refresh token (which
stays the same since sign in) to generate
     * a new access token.
     * @type {request.RequestAPI<request.Request, request.CoreOptions,
request.RequiredUriUrl> | request}
     */

    var request = require('request');

    var options = {
        method: 'POST',
        url: 'https://accounts.spotify.com/api/token',
        headers: {
            'cache-control': 'no-cache',
            Connection: 'keep-alive',
            'Content-Length': '173',
            'Accept-Encoding': 'true',
            'Content-Type': 'application/x-www-form-urlencoded',
            Cookie:
                'remember=1; _ga=GA1.2.1841930364.1564302868;
sp_dc=AQBRfigyfwzPDuKDr3TD7B9t8jz8ibWyPCNqlI_l69t4ag83KWgVhEad-Q_XQaJW-rJTsuWF-
T01GCO5OmJYMW4F1lMHzDa4rDRzJrUWVQ; sp_key=33e53da7-1842-4597-8782-1542e79bb5b2;
csrf_token=AQDLW8zfPN3hKQxA-qZ-pHXVc98Im7n7lRY5cqKmRl3TQgYFshchxZzg_iz-
iV532Sos6frSI9nvjfw;
sp_ac=AQCioZsuj_tT96FmVqPkVd392uDDEtqzGG01J2sB_sxXJzU8abs9RD8ImjMCn7zDdBCZ26cnnAKIR
98GYatUGaQ45J56b7YNBIPVgEazCURdWQ-xR4x0yiwmcM-19LRhaEVQleaaM4cc_dkm6PaqFwMDT7yBG-
yYoV3uJ-iVV0PSzv-pJJT_NWnNxb9kl9KSBQ-WMw3BXbNUw92fpRqVLn_3ivbwg06kWPs',
            Host: 'accounts.spotify.com',
            'Cache-Control': 'no-cache',
            Accept: '*/*',
            'User-Agent': 'PostmanRuntime/7.15.2',
            Authorization:
                'Basic
MzcyZDFiYTU0Y2VlNDIxNjgxZGM0NmNmYzZhMmJkMTU6ZDU1MDAwYTRjNGQ1NDVhMTg1NjExYTRhOWNiMjg
3ZmQ='
        },
        form: {
            grant_type: 'refresh_token',
            refresh_token:
`AQCkGuy2h2pV8SCNOJjYP2FpUgQcjV4gkRxf_uW7LvJ6GyNFGPThxURArnsorh75QHqxa3CMUcREIV1pvb
_jh2IFXi2APj6gRKcPC0jifUumrd4fA2ZgOZ8RMzUWZ9U0Pt6F_Q`
        }
```

```
    };

    /**
     * Treating the response of the request: Store the returned body of the
response and then use a redex to extract only
     * the new access token.
     */
    request(options, function (error, response, body) {
        if (error) throw new Error(error);
        var returnedString = body;

        console.log('Whole response of new access token', returnedString);
        const regex = /(?<="access_token":")[^"]+/gim;

        let temporary;
        // While temporary, after asssuming the regex-returned string, does not
equal null
        while ((temporary = regex.exec(returnedString)) !== null) {
            // This is necessary to avoid infinite loops with zero-width matches
            if (temporary.index === regex.lastIndex) {
                regex.lastIndex++;
            }

            // The result can be accessed through the `temporary`-variable.
            temporary.forEach((match, groupIndex) => {
                returnedString = match;
                console.log(`Found match, group ${groupIndex}: ${match}`);
            });
        }
        console.log('Regex of new access token', returnedString);
        setterForUpdatedAccessToken(returnedString);
    });
```

Figure 29: playSadSong function with its Post Request for a new Access Token. NB: this POST request is inserted within this function to enhance readability here; It has its own function and is used by all four emotion intents in the actual code (See Appendix H)

Before this happens, however, the POST request for the updated Access Token must be made. As mentioned in the System Design section, it is imperative that an up-to-date Access token – used as an OAuth token – is used for any Spotify API calls. As such, every recommendation begins with such a POST request (see Figure 29). The updated Access token is then set to the class variable for use in other functions. The Node.js 'request' library is used to make this POST request. Due to the exacting nature of the request, which needed to be formed in a specific and precise way (Spotify, 2019a), the 'Postman' application was used to test the request. Gradually over a period of two or three days, the request was finally successful and the resulting code was translated over into the Webhook. The issue of an expired Access token was no longer causing the system to crash and the user could continually use the application despite token expiry times. Despite this, the Google agent was not initially 'saying' the result of the song retrieval.

It was found that, although the results were being sent to the application as shown by the Firebase console logs, the Google agent timed out before the results could be vocalised to the user. This, in addition to the Access token expiry problem, led to the largest design decisions of the system. The system needed to be designed in such a way as to allow for Dialogflow's standard '5 second timeout' rule (Google, 2019e). Instead of providing the user with Audio analysis simultaneously with the song name and artist, API calls are made sequentially with separate intent handlers through child intents. Meanwhile, the parent intent will store the necessary information to be used in the child intents. For instance, in the 'playSadSong' function, the updated Access token and all other necessary song data is set for use in either

the ‘giveAudioAnalysis’ or ‘playMusic’ functions. This staggering of API calls within intent handlers, in addition to use the ‘await’ keyword inside ‘async’ functions - meaning that the parent function must wait for the child function to finish executing before continuing the execution of the rest of the code, solved the issue of timing out before the allotted ‘5 second’ timeframe; It also gave the conversational flow its structure. Once the song information and Access token is set, the ‘playSadSong’ function calls the ‘recommendASong’ function.

5.3.2.2 ‘recommendASong’ Function

```
function recommendASong(agent){
    var sw = require('sentiword');

    let POSAnalysis = sw(SongInfo.maxSentence);
    POSAnalysis = POSAnalysisConverter(POSAnalysis.sentiment);

    // Agent vocalises the retrieved song to the user
    agent.add(`I recommend ${SongInfo.song} by ${SongInfo.artist}. According to my
lyrics analysis, those used in ${SongInfo.song} are mainly
${grammarify(SongInfo.maxEmotion)} with ${watsonScoreAnalyser(SongInfo.score)} high
degree of certainty. The following extract, ${POSAnalysis}, was found to be the
most ${grammarify(SongInfo.maxEmotion)} lyric in the entire song. I'll read it to
you now, Ahem... ${SongInfo.maxSentence}... `);

    agent.add(`Would you like to hear audio analysis of ${SongInfo.song}'s
melody?`)

}
```

Figure 30 : recommendASong function stores the Access token so it is ready for use in further API calls

Within the ‘recommendASong’ function, the Google agent will use the data stored within the ‘SongInfo’ class variables to recommend a specific song to the user. The user is provided with the name of the song and the artist in addition to the maximum emotion with how much certainty exists for placing this emotion untoward this song – either ‘relatively high’ (over 0.5) or ‘extremely high’ (over 0.75). Similarly, trivial functions exist for making the sentence read more grammatically correct. The agent will then read the top scoring excerpt from the song for that particular emotion but before then, as an added feature, the agent will state whether the excerpt has been received as mostly ‘positive’ or ‘negative’ as according to the ‘SentiWordNet’ analysis of it. To do this, the ‘SentiWord’ Node.js wrapper (https://github.com/DanielMurdoch/SentiWord) is used (Murdoch, 2014) with a simple function ‘POSAnalysisConverter’ to control whether the outputted word is positive or negative based on its overall score. The function ends with the agent asking if the user wants “to hear audio analysis” of the song.

5.3.2.3 ‘giveAudioAnalysis’ Function

```
async  function giveAudioAnalysis(agent) {
    /**
     * Callout to the Spotify api using the spotify-web-api node package
     * Agent vocalises the analysis extracted on the track.
     */
    await Spotify.getAudioAnalysisForTrack(`${SongInfo.just_uri}`).then(
        function (data) {
            let temp = Math.round(data.body.track.tempo);
            let key = data.body.track.key;
            let mode = data.body.track.mode;
            setTempo(temp);
```

```
            setKey(key);
            setMode(mode);

        },
        function (err) {
            console.error(err);
        }
    );

    await Spotify.getAudioFeaturesForTrack(`${SongInfo.just_uri}`)
        .then(function(data) {
            console.log(data.body);
            let danceability = data.body.danceability;
            let valence = data.body.valence;
            setDancability(danceability);
            setValence(valence);
        }, function(err) {
            console.log(err);
        });

    let str = convertNumberToKey(SongInfo.key);
    let keyPlusMode = str.concat(convertNumberToMode(SongInfo.mode));
    agent.add(`The song has a ${lowMediumHigh(SongInfo.tempo)} tempo with
${SongInfo.tempo} beats per minute and is
${suitabilityForDancing(SongInfo.danceability)}. It is in the key of
${keyPlusMode}... Songs that are written in the key of ${keyPlusMode} often evoke
${convertKeyToEmotionalMeaning(keyPlusMode)} emotions...
${despiteOrFurthermore(SongInfo.mode, SongInfo.valence)} Would you still like to
listen to this song?`);
}
```

Figure 31: 'giveAudioAnalysis' Function

A 'yes' reply to the 'recommendASong' function's question, in a 'sad' emotion Use Case, will instigate the '-Sad - yes - yes' intent's fulfilment; This is mapped to the 'giveAudioAnalysis' function which is then executed. This function once again utilises a Node.js wrapper. The Spotify Node.js wrapper was used as a simple way of making the API call to receive Spotify's 'Audio Analysis' and 'Audio Features' data (Thelin, 2014). The project uses the song's tempo, key and mode from the former and the song's danceability and valence from the latter. In addition to more trivial functions in displaying this data to the user, this project converts both the key and the mode into the emotional meaning which they often convey in music.

```
function convertNumberToKey(keyNumber){
    let keyNumbers = new Map();
    keyNumbers.set(0, 'C');
    keyNumbers.set(1, 'C-sharp or D-flat');
    keyNumbers.set(2, 'D');
    keyNumbers.set(3, 'D-sharp or E-flat');
    keyNumbers.set(4, 'E');
    keyNumbers.set(5, 'F');
    keyNumbers.set(6, 'F-sharp or G-flat');
    keyNumbers.set(7, 'G');
    keyNumbers.set(8, 'G-sharp or A-flat');
    keyNumbers.set(9, 'A');
    keyNumbers.set(10, 'A-sharp or B-flat');
    keyNumbers.set(11, 'B');
    return keyNumbers.get(keyNumber);

}
function convertNumberToMode(modeNumber){
    let modeNumbers = new Map();
    modeNumbers.set(0, ' minor');
```

```
    modeNumbers.set(1, ' major');
    modeNumbers.set(-1, '');
    return modeNumbers.get(modeNumber);
}


function convertKeyToEmotionalMeaning(key){
    let keyEmotions = new Map();
    keyEmotions.set('C major', 'happy and uplifting');
    keyEmotions.set('C minor', 'sad and love sick');
    keyEmotions.set('C-sharp or D-flat minor', 'despairing and sorrowful');
    keyEmotions.set('C-sharp or D-flat major', 'grieving and depressive');
    keyEmotions.set('D major', 'triumphant and victorious');
    keyEmotions.set('D minor', 'negative and melancholic');
    keyEmotions.set('D-sharp or E-flat minor', 'distressing and angsty');
    keyEmotions.set('D-sharp or E-flat major', 'cruel and hardened yet intimate');
    keyEmotions.set('F major', 'angry and regretful');
    keyEmotions.set('F minor', 'depressive and harrowing');
    keyEmotions.set('E major', 'boisterous, quarrelsome but also joyous');
    keyEmotions.set('E minor', 'relamorous, restless and grief-carrying');
    keyEmotions.set('F-sharp or G-flat minor', 'gloomy and resentful');
    keyEmotions.set('F-sharp or G-flat major', 'relief and clarity-giving');
    keyEmotions.set('G major', 'calm, idyllic and fanciful');
    keyEmotions.set('G minor', 'discontentful and uneasy');
    keyEmotions.set('G-sharp or A-flat major', 'haunting and lingering');
    keyEmotions.set('G-sharp or A-flat minor', 'resentful, life-loathing and
negative');
    keyEmotions.set('A major', 'loving and joyful');
    keyEmotions.set('A minor', 'tender, plaintive and pious');
    keyEmotions.set('A-sharp or B-flat major', 'optimistic and hope-filled');
    keyEmotions.set('A-sharp or B-flat minor', 'pessimistic and dark');
    keyEmotions.set('B major', 'angry, jealous, desparing and burdened');
    keyEmotions.set('B minor', 'solitary, melancholic and patient');
    return keyEmotions.get(key);
}
```

Figure 32: Mapping key and mode to their emotional significance functions

The key number and mode numbers received from the Spotify API request are first converted into its respective key, through the twelve pitch classes (Morris, 2007), and mode (major or minor) and are then mapped to their respective emotional meaning in the 'convertKeyToEmotionalMeaning' function. This is done using the key characteristics as presented by Purwins (2005) and reinforced by the musical magazine LedgerNote (2019).

5.3.2.4 'playMusic' Function

```
function playMusic(agent){
    agent.add('Enjoy!');
    let request = require("request");

    let options = { method: 'PUT',
        url: 'https://api.spotify.com/v1/me/player/play',
        headers:
            { 'cache-control': 'no-cache,no-cache',
                'Content-Type': 'application/json',
                Authorization: `Bearer ${Access.accessToken}`,
                Accept: 'application/json' },
        body: { uris: [ `${SongInfo.Spotify_uri}` ] },
        json: true };

    request(options, function (error, response, body) {
        if (error) throw new Error(error);

        console.log('This is the body of the play music request ' + body);
```

```
    });
}
```
Figure 33: 'playMusic' Function

A 'no' reply to the agent's question of whether the user would like to "hear audio analysis" of the recommended song followed by a 'yes' reply to whether the user would still like the agent to play the song will initiate the 'playMusic' function. Since no music playing functions exist within the Spotify Node.js library, this function sends out an API 'PUT' request as per the Spotify documentation (Spotify, 2019b). The successful request uses the Access token and the song's Spotify URI, both of which are set within the 'playSadMusic' function. If there is an 'active device', a device which has the Spotify application running, the recommended song will start playing upon execution of this function.

5.3.3 Conversational Loops and Ending of Conversation

As shown in Figure 27, some functions are matched with multiple intents. This is to ensure circularity within the application conversation and allows the user more freedom with which to choose their music. They are able to:

1. …without hearing audio analysis…
    a. listen to the song,
    b. choose another song,
    c. or end the conversation without hearing the song or choosing another
2. …in addition to hearing the song's audio analysis…
    a. listen to the song,
    b. choose another song,
    c. or end the conversation without hearing the song or choosing another.

A full control flow diagram of the conversational flow including all the different branches can be seen below (Figure 34); It was used in determining comprehensive testing as discussed in the next section.

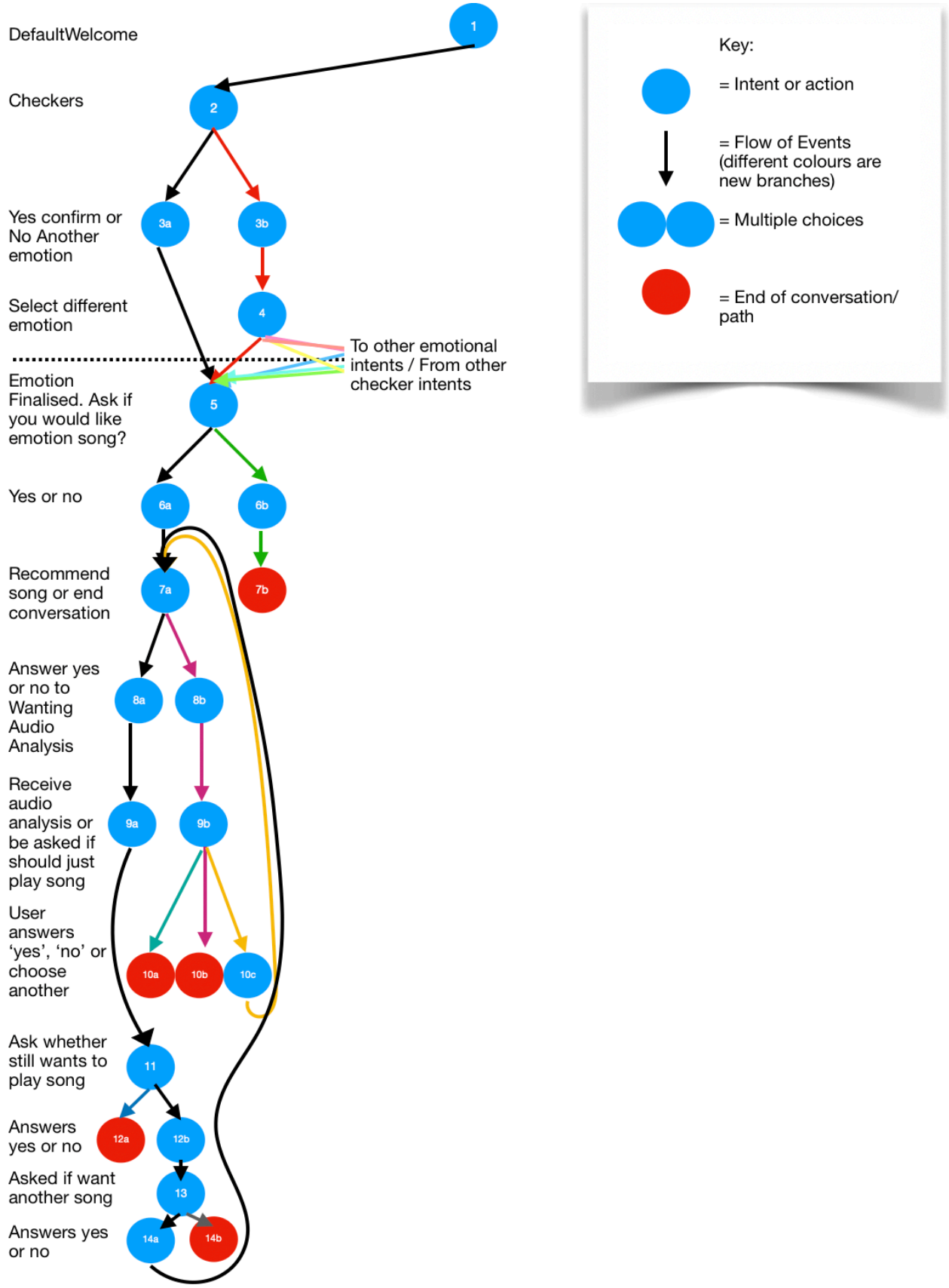


Figure 34: Control Flow Diagram of Conversational Flow for One Emotional Intent

6. Testing

6.1 Overview

This section will outline the testing and evaluation procedures for the project. The Functional and Non Functional requirements will be used, in addition to Use Case scenarios, to evaluate the extent to which the system fulfils the project specification.

Due to the complexity of the project, it was decided early on that both 'Unit Testing', testing a single feature or function, and 'Integration Testing', testing multiple connecting features or functions, was necessary. Since data flow specifications entwined with requirement specifications allow for a comprehensive testing environment (Chen, Liu and Nagoya, 2005), these were used in tandem with the 'Conversational Flow' Control Flow diagram (Figure 34). The diagram shows all the possible branches which one emotional intent consists of – these are depicted by the number of varying colours of arrows. The number of possible paths is a much greater number, however, and would require an automated testing facility; It was therefore out of the scope of this project to test all the different paths for every intent. Due to the large amount of possible paths, limited Path Coverage was implemented through several typical and atypical Use Cases. This could enable Integration Testing since it demonstrated the functioning of a varying number of cooperating functions and intents. One such scenario is found in the Use Case diagram (see Figure 1 and UCID 1 in Appendix Ei).

Functional requirements for the sentiment tone analysed song database (FR2.0-2.3), whether the matched song was a beneficial song to hear in the user's current emotional state (FR6.4) and the Non Functional Requirements surrounding the closeness and appropriateness of the agent's response (NFR4.2 and NFR4.3) required their own testing and will be discussed towards the end of this section.

6.2 Unit Testing Results

'Unit Testing' was achieved through Branch Coverage in which all of the possible branches of the system were tested. The testing for this was conducted by the developer of the system by speaking to the system with a wide range of utterances used unless the intent only required a simple 'yes' or 'no'. The user speech, expected response and actual response are recorded and can be found in Appendix C.

The final Unit Testing results were successful with a 100% rate of being directed to the correct intent – this is shown by the user's speech invoking the correct intent, and therefore the correct agent response. These tests confirm the fulfilment of all functional requirements between FR3.0 and FR6.3, inclusively; This includes the 'playMusic' function (see Appendix Cv). The 100% success rate in addition to the architectural design of the system, which must respond within 5 seconds, also confirms the fulfilling of the Availability and Efficiency Non Functional requirements. The Security requirements are also met by the inability to manipulate other user's data. The Accuracy requirement, NFR4.1 ("The system must be able to detect input speech and process it as text with an 85%+ accuracy."), is confirmed since

each emotional intent managed over 85% for recording the user speech in full, however, for NFR4.2 ("The system may provide an agent response which is at least 70% appropriate to the user's speech.") and NFR4.3 ("The system may provide an agent response which has 70% closeness to a human response."), another method of testing was deemed appropriate. Since it required additional users, this is discussed in the Integration Testing Results section.

6.3 integration Testing Results

6.3.1 Methodology

Integration Testing was completed via 'Path Coverage'. 'Path Coverage' ensures that every combination of branches are explored through every single path. This was important since DialogFlow 'Intents' require an input and output context, the lifespan of which can create unexpected results; In these cases 'Branch Coverage' alone may fail to detect these errors.

In order to conduct thorough 'Path Coverage' testing without being able to go through all possible combinations – since all the possible paths through the app, due to looping and the ability to swap emotional intent part way through, presides in the extremely large number range – several 'Use Cases' were devised which explore a wide range of different paths. Following the release of the Beta version of the application, Google allows developers to invite up to 20 people for testing purposes. Respondents' email addresses were included in the application's page on the 'Actions-on-Google' console and, once they followed the instructions to launch the application, were asked to fulfil each 'Use Case' on every emotion in addition to filling out a simple questionnaire (see Appendix Ei). The results can be found in Appendix Eii and Eiii.

It is worth noting that it was not possible to submit the Spotify app to the Spotify developer submissions page due to the submission form on the website not responding accordingly. This was trialled across multiple browsers, machines and internet connections and upon emailing the Spotify team about this, it was concluded that users would be unable to link their Spotify accounts to the application due to the Spotify app remaining in the 'sandbox' environment. The Spotify app contains the necessary credentials for using the Spotify API and normally provides the connection between Spotify user accounts and the Spotify app, allowing for connectivity between the Spotify user account and the Google Assistant application; Other users are not able to connect to the Spotify app in sandbox mode since it is only seen by the app's creator itself. This meant that, for testing purposes, respondents were provided with the system's developer Spotify account details. It was therefore not possible to test the first functional requirement on other user's accounts. Despite this, one Spotify account was successfully tested across several devices; The results of these tests are outlined below.

6.3.2 Testing the correct response and intent matching throughout the Path

Similar to the unit testing conducted, the first test which respondents were required to answer was whether all the agent's responses were correct for the given path. The success rate of 100% was updated from 95% of testing in the previous round; This was due to some input contexts and output contexts not being correctly matched with the intents. Once all the correct input and output contexts were shared by the necessary intents, the high success rate was achieved.

6.3.3 Testing the Appropriateness of the Response (NFR4.2) and Closeness to being a Human Response (NFR4.3)

Although it is difficult to judge the agent's reply, since it would require a degree of subjectivity in judging its 'closeness' to a human response or 'appropriateness' of a response to the user statement, the respondents were asked to give the entire path's responses a rating of (0-10) for these two factors. The average between the respondents for 'closeness to being a human response' was 8.1 and 'appropriateness of response' was 8.9. This fulfilled NFR4.2 and NFR4.3 but it showed that the application certainly has room for improvement, especially in other real world scenarios which might not already be included as entities within the system.

6.3.4 Evaluating whether the Song matched the user's Emotion (FR6.4)

Due to the subjective nature of FR6.4, ("The song may fit the intended mood as per the user's own subjective criterion."), the respondents were asked to assess the extent (as a percentage out of 100) to which the first recommended song fits their designated mood in their own opinion. The results showed an average of 72.6% for the recommended songs to be deemed as matching the emotion asked for. This result indicates that the system, more often than not, matches a correctly fitting song. Despite this, the test is limited by the user having to make predictions on which song they think they would like to hear given the emotion of the Use Case, an emotion which they most likely were not feeling during testing. This could have resulted in incorrect evaluations of the chosen song in addition to a degree of guesswork by the respondent predicting what effect the song would have on them should they be feeling a particular emotion. Although this is an important requirement, around which this project is focussed, it also does not account for the user's ability to choose another song before it is played. Further limitations and evaluations are discussed in the Evaluations and Future Development section (Section 7).

6.4 Testing the accuracy of the Pre-Analysed Song Database (FR2.0-2.3)

FR2.1 ("The system must use IBM's Watson Tone Analyser to create a database of pre-sentiment-tone-analysed songs."), is fulfilled since all of the songs in the database used the Watson Tone Analyser endpoint to be analysed. Likewise, FR2.2, ("The database of songs must have the artist and song names, in addition to the emotion that was found to match the song with the highest certainty, the sentence which matched the emotion with the highest score the most along with its certainty score for matching it."), is fulfilled since the Firebase database contains all the necessary columns with no 'NULL' values. Finally, for testing the correctness of the data being recorded, 1. The outputted scores had to align with the correct song, 2. In the case that more than one score was received in the JSON response, only the highest score and the corresponding emotion of that score would be chosen and 3. The data recorded has to be consistent for repeat requests.

To ensure this, Junit tests were deemed insufficient since although the function within the code could have functioned perfectly well, the wrong data could have been sent from the Watson Tone Analyser endpoint or to the output SQL table. For this reason, it seemed more

appropriate to compare the console log data from within the function to the recorded data in the output SQL table and finally cross examine this with the online demo version of the IBM Watson Tone Analyser.

Multiple ‘console.logs' were used throughout the code to record the data for a particular song – this showed all the possible tone scores at a document and at a sentence level. The maximum of each was compared with the outputted data onto the SQL table to ensure that the correct data being recorded was indeed the data which resided on the output table (see Appendix Di and Dii). This was performed using 100 songs with a success rate of 100%. Finally, to test the consistency of the song lyric analysis, a while loop was used to assess 10 random songs at least 15 times in a row (see Appendix Diii); Consistent data was recorded across all tests. Evaluation of this test and the others are discussed in the following section.

7. Evaluation and Future Development

7.1 Overview

This section shall outline the evaluation of the project through measuring its overall success. The limitations with both the Unit Testing and Integration testing are discussed first followed by the extent to which the system fulfils the original specification second, and recommendations for future developments are discussed third.

7.2 Testing Limitations

The testing in this project produced 100% accuracy in the application’s ability to bring users to the correct intent, however, these tests certainly do not account for the wide range of user utterances which could be used as the input strings to the application. In a real world environment with multiple users using the system, the range of input strings spoken by the users would be significantly greater than the finite number of inputs tested in this project. These inputs would include emotional ‘Events’ which might not be accounted for within the Dialogflow entities, in addition to regional dialects and even other languages. Testing which covers this would only be possible on a much larger scale which goes beyond the scope of this project; Although, providing machine learning facilities within the application to ‘learn’ any new user inputs and store them for future use is an improvement worth noting and one which is discussed in Future Development later in this section.

To test for the success rate in how close the system’s song recommendation matches the desired emotion, the questionnaire (see Appendix Ei) was developed whilst taking into account typical limitations in self-report style questionnaires. For instance, Brewer-Venaik’s criticism of the GLOBE self-report style questionnaire which did not allow for marginal preference choices between responses (Taras, Steel and Kirkman, 2010) inspired the 0-100 scale in measuring a recommended song’s closeness to the desired emotion; This would allow respondents the opportunity to marginally prefer one song recommendation over another. A similar approach was chosen for testing the closeness to a human response and appropriateness of the response (see appendix E) using the 0-10 scale. The limitations of the questionnaire style testing come with the typical limitations associated when using humans in

testing including respondent confusion over the meaning behind ratings and how to conduct the tests. For this reason test subjects were provided with brief explanations of each rating on the scale, however, it is unlikely that all of the test subjects were entirely consistent in their judgements throughout the testing and cognitive failures would have been present (Reason, 1990). Moreover, the questionnaire was not completed under control conditions meaning that there would have been a multitude of variables between participants. For instance, some could have been testing the application in a relatively calm state after finishing dinner but others may have conducted the testing in a rush before work; This would have affected the time allowed for conducting the tests, for example listening to the entire recommended song rather than just a segment of it, in addition to affecting the resting mood of the user. Both of these limitations could be mitigated against to an extent by conducting supplementary, controlled interviews (Reason, 1990); This would still leave room for human error in addition to being beyond the scope of this project.

In conclusion, the testing used by this project fails to offer full Path Coverage. Furthermore, the more subjective requirements testing will have elements of human error. Despite this, complete Branch Coverage testing was conducted which enabled the accurate testing of whether user responses matched with the correct Intents. More thorough testing using real world scenarios would be necessary to offer a comprehensive testing of the system, however, the select Use Cases and questionnaire findings should at least serve as a useful guide from which to conduct future testing.

7.3 Limitations of the System

The most obvious limitation of the system in its current state is the inability to allow for other users to connect to it. It is hoped that Spotify will allow new app submissions via its developer's page so any Spotify user is able to use Mood Magic Therapy. Taking this as a given, upon using the application, the requirement for users to visit the cloud hosted page and link their Google and Spotify Premium accounts limits the system to a certain extent. Since Mood Magic Therapy is intended primarily as a voice application, this detracts from the typical Use Case where first-time users may want to state the invocation phrase and start using the application immediately. Unfortunately, due to the restrictions imposed by Spotify on ensuring that third party applications are used by paying subscribers, this is an inescapable limitation for the project. It is hoped that, in future developments, Google will improve the accessibility of linking accounts and applications with Spotify.

Another limitation of the project derives from the cases where the 'Sentiment Tone Analyser' and 'SentiWordNet' analyser fails to find the correct tone in the song lyrics. The most problematic case found during the testing of the closeness of the song to the user's desired emotion was Céline Dion's cover of 'All By Myself'. Nearly all of the participants flagged this recommendation for the emotion, 'Confident', as being inaccurate. The lyrics (see appendix F) seem undoubtedly sad, rather than confident.

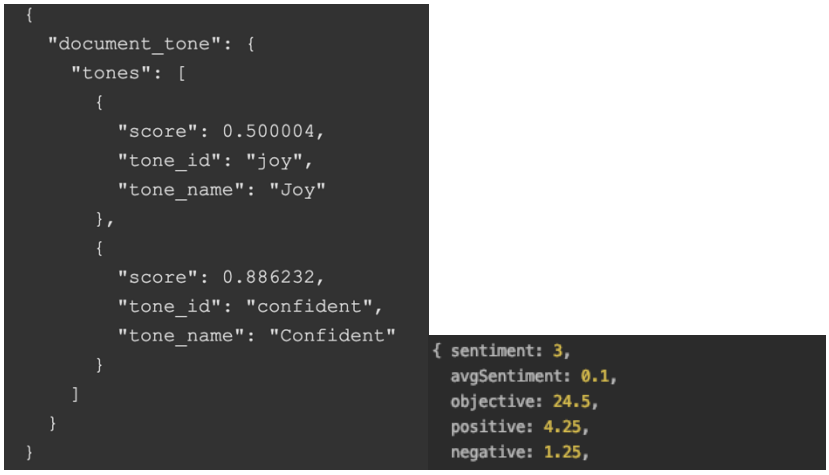


Figure 35: IBM Watson Tone Analyser's analysis of 'All By Myself' (left) and SentiWordNet's analysis (right)

Despite this, both sentiment tone analysers in this project failed to find the sadness within the song (see Figure 35). The certainty of 'sadness' was below the 0.5 threshold on Watson's Tone analysis despite recording the 'joy' within the song. Although SentiWordNet analysis used in this project was focussed on analysing the excerpt read by the Google agent, the complete lyrics of 'All By Myself' were sent to the SentiWordNet API. It, too, found the song to be mostly positive with an average sentiment score of 0.1, meaning even the average sentiment was positive. After examining the SentiWordNet N-Grams and synsets recorded in this example, it seems that the majority of the words used in the song do seem positive if viewed individually or as bigrams or even trigrams but perhaps in this case a larger N-gram number is required. For instance, all of the following words, 'Don't' 'want' 'to' 'be', need to be analysed with respect to the entire next line of the song, 'all' 'by' 'myself'; The negative contraction, 'Don't', in the former line should at the very least be attached to the latter line's 'All by myself' so that it is clear the singer does NOT want to be alone. It seems that 'don't' fails to be registered in this way throughout SentiWordNet's analysis whilst positive words such as 'love', 'young' and 'fun' are well-represented synset terms. Despite not being able to receive specific analysis such as this from the output of the Watson Tone Analyser, it is probable that its analysis of 'All By Myself' falls victim to a similar fate. This specific example, therefore, demonstrates a limitation in the accuracy of sentiment analysers themselves; Which is something that sentiment analysers of the future may be able to improve upon.

Another limitation, which relates to the previous one, would be that SentiWordNet analysis is only provided on the song lyrics excerpt with the maximum score for the desired emotion. A possible improvement for the database of pre-analysed songs would be to conduct both IBM Watson Analysis and SentiWordNet analysis on the entirety of the song lyrics. The outputted database could then include both scores individually along with an amalgamated average score between the two for added analysis. Although this may have improved the accuracy of the sentiment analysis in some cases, Watson's Tone Analyzer was found to have the highest accuracy compared with other analysers (IBM, 2019); Whereas there was no data on the accuracy of using both analysers together. It was therefore decided that the SentiWordNet analysis could be better suited for providing the user only with additional information on the song lyric excerpt and it could be added to future iterations of the system once more testing could be conducted to show whether it would be beneficial to sentiment tone analysis of song lyrics.

A further database issue would be the lack of full punctuation received from lyrics scraping. Almost all song lyrics are written in poetry style with stanzas and commas. It was necessary to take out the commas for importing to the SQL database. Furthermore, the lyrics required full stops in order to be able to provide sentence level analysis. Initially, every line in the songs' lyrics ended with a full stop however this led to many extremely short sentences, some only being two or three words long. It was therefore decided that longer song excerpts would be more engaging for the user to hear and provide the user with more indication as to the tone of the lyrics used in the song. The full stops were therefore spaced out over one to four lines of the song. Although this improves the user experience by providing a lengthier song excerpt, it might not necessarily be where the artist originally intended the sentences to end. These newly placed full stops and missing commas may therefore change the sentiment expressed slightly since the Watson Tone Analyser takes punctuation account (IBM, 2019). Since it was impossible to find lyrics with full punctuation as directed by the artist themselves for a large database, this decision to provide full stops was concluded to be the appropriate decision. However, it is worth noting that the lack of punctuation limits the ability to connect the song lyrics to an emotion to a small degree.

As a potential limitation on the application's ability to provide multimodal analysis, the database of pre-tone analysed songs could have also accounted for melodic analysis; This would negate the need to provide the user with melodic analysis via speech. This would have required sending the lyrics to the Watson Tone Analyzer and sending the song URI to Spotify simultaneously and outputting an average of the results based on the tone analysis of the lyrics, the key of the melody, the mode and the song's valence and danceability parameters. Once again, since there is no thorough testing nor conclusive studies which use all of these parameters and the song lyrics together to produce the song's emotion – for instance, there is no pre-defined protocol for defining the overall emotion if a song has a high valence, low danceability, D# key, minor mode and relatively certain score of 'confidence' for its lyrics – it was decided that it would be more useful to provide this audio analysis to the user and let them choose whether to listen to the song or not. This has the added benefit of making the user more informed in their choosing along with the freedom to say 'no' should the song not sound like the one which they would like to hear. In an ideal world, however, such testing would have taken place so that mood categorisation can employ all multimodal analyses.

Finally, an important limitation this project has is the need for the database of *pre*-tone analysed songs. This means that users are restricted to the songs in the database rather than their particular library of songs. Limited by the number of API calls that can be made to the Watson Tone Analyser under the 'basic' (free) account, the project had to take a random sample of the 55,000 songs from the original database. Even this would not have been enough for most users, however, since they are still restricted even with more songs. An improvement to this application would be if Spotify implemented its own sentiment tone analyser service and used it to supply lyrics data on all of its songs. In this case, there would be no need for the pre-tone analysed database since all of the songs on the Spotify library, some 30+million songs (DigitalTrends, 2019), would have lyrics analysis included and accessible via the Spotify API; The Google Assistant would only need to retrieve this information and the user can have full access to a library which would be likely to satisfy their needs. It would seem that this would be within the reach of Spotify to achieve in the not too distant future, however, it remains a limitation for this project.

7.4 Future Development

In addition to addressing the limitations of the project, mentioned in section 7.3, the future development of this application could be segregated into the near-future and far-future developments.

In the near future, the application could be improved by enabling 'learned' words and phrases from user's input speech. Currently, under the guise of the Checker System, the application will respond to a user's denial of being directed to the correct emotion, "I'm sorry, I'm still learning. What of the following four emotions would you like to hear a song in…". The user then responds with the emotion that they wanted in the first instance. To enable the application to 'learn', the system should store the phrase that the user used in the Firebase database and then the emotion that the user asks for in reply to the checker intent could be stored with it. Over time, the system would keep track of all these phrases along with the desired emotion. The intents must then be modified to scan the database for the phrase used by the user. This would be relatively simple to modify, however, it would bring with it the disadvantages of users using the same phrases but with differing emotional aims or even users purposely attempting to trick the system into containing incorrectly emotion-defined phrases. It would therefore be necessary to include moderation within the application so that any harmful users could be banned from the application entirely.

Another near-future development would be to integrate other home tech devices with the application. Wifi enabled lightbulbs, smart TVs and even smart kettles could be introduced. As an example, a user who is found to be wanting a sad song would start hearing the sad song whilst the lights in the room turn a soothing blue colour, the TV begins a slideshow of melancholic paintings, and the kettle begins boiling with the Google agent saying "Everything is going to be OK. I'm here for you [user name]. Let's get through this sad spell together with some sad music, some fine art from artists going through similar problems and a nice warm cup of tea!". The agent could then provide brief notes on the artwork creating a 'music and the arts' therapy application, which has been shown to be beneficial for mental health therapy (Chancellor, Duncan and Chatterjee, 2014). Although the Wifi kettle is currently unavailable for integrating with the Google assistant, in the US applications can be integrated with Google Home's 'Nest' TV application. Using this application, users can cast picture slideshows automatically from their Google Photos library. With four album folders containing different artworks which fit under the four emotion categories, the Google Assistant would therefore only need to initiate the casting of pictures from the desired emotion album via the Nest API upon an emotion intent being matched.

In the far future, it is estimated that humans will be entwined with technology via a human-brain-machine link such as Elon Musk's 'Neuralink'. In this case, such an application would not need to rely on user speech to detect emotion. Instead, mood would be assessed via methods such as heart beats per minute and brain scans, and the fully integrated smart home would respond accordingly. Until this distant future becomes a reality, however, perhaps the application would benefit from using IBM's Watson Tone Analyser for analysing user speech instead of using Dialogflow custom entities. This would allow for the system to offer more precise emotion measurements of the user's emotion, for instance, being able to tell the difference between a moderately sad user and an extremely sad user. The database could also be structured accordingly so that the songs are listed in order of their scores for each emotion. This would also suit gradually increasing in positivity in music selection which Schäfer et al. (2013) found to be beneficial when providing Music Therapy since the user's exact emotion

is matched to a song with the same emotional score first followed by songs that gradually increase in positivity scores. To make this relatively simple alteration, there would only need to be one checker intent which checks for all emotions after the DefaultWelcomeIntent. The checker intent would include in its fulfilment an API request containing the user's speech to the Watson Tone Analyser and the result of which would be mapped to specific functions based on the emotion and score. These functions would have the aim of calling the Spotify create a playlist API with a list of suitable songs in gradually increasing positivity of emotion. The user would then need to say 'yes' to begin the Music Therapy, using the stored Watson Tone Analysis results as class variables for directing to the correct function, to allow time for the 5 second maximum response time of the application. Their 'yes' response would then be mapped to a 'playPlaylist' function and the list of songs would start playing. Due to the restrictions on API calls to the Watson Tone Analyser, however, a paid account would be needed for the API calls to IBM in order to allow the application to be scalable. Furthermore, continuing to give the user song information would be made difficult in this version of the application since the Dialogflow agent must end the conversation in order to hear the music through the speaker. This closes the Mood Magic Therapy app meaning it cannot execute any more functions. Thus, this update would only be able to play the music without giving the user freedom to change songs unless there are changes in the future wherein the Google Assistant allows this. Presently, however, it remains necessary to provide a conclusion on what this project has achieved, its limitations and potential improvements.

8. Conclusion

This section shall, first, outline what the main challenges of the project were and how they were met, second, describe what the most important services were and how they were added to, third, evaluate the successes and failures of the project, and fourth, discuss the design choices and their effect on the project. Finally, a conclusion will be provided taking account of these aspects.

The most challenging part of this project was creating seamless communication between the Google Assistant application and the Spotify API service. This section of the project had several stages: 1. Authorise the user's Spotify account, 2. Retain the account throughout the application's use and, 3. Enable the Google Assistant application to request and use the requested data within the application.

To tackle the first stage, a Google-Spotify authorisation template in addition to a Spotify API library were used. Neither of these functioned properly, however, resulting in further challenges. The first of these involved making the request to the Spotify API service containing the correct authorisation code. After testing that the authorisation code was in fact correct despite the console errors by examining it in Postman and applying a 'Regex' to remove any whitespaces, it was decided that the best way to solve the problem was to delete everything and start again from scratch; This technique paid dividends since on the second time around, the authorisation methods suddenly worked.

The next challenge, retaining the account throughout the application, was achieved by meticulously following the Spotify API documentation for creating POST requests to the API service using the Refresh token. This was an extremely arduous process comprising many

parameters which could be formatted incorrectly. The Postman application aided in forming the requests necessary for the project but the challenge was ultimately met by an eye for detail combined with persistence.

Finally, the unexpected challenge of enabling the Google application to use the data from the Spotify API within the allotted timeframe created a challenge which could not be overcome. After reading the API documentation for Dialogflow, it was concluded that there would be no way of providing users with all the received data in a single agent response. This problem determined the most profound and influential design choice of the project; creating an extended conversation flow which allowed users to confirm or deny their wanting of further audio analysis whilst important data from previous API calls in the webhook would be stored for later use. Though there were multiple challenges during this project, these three challenges, which were all successfully met, were those which required the most energy and creative thinking.

This project used several services, the most integral being IBM's Watson Tone Analyzer, Google's Dialogflow and the Node.js Request library. First, the entire database of pre-tone-analysed songs was made using IBM's Watson Tone Analyser. The resulting JSON data was manipulated to find the maximum emotion conveyed by the song and its' corresponding score in addition to the song's top scoring sentence for that particular emotion. This manipulation consisted of devising appropriate looping algorithms which could examine, assess and compare the appropriate data from the appropriate sections of the JSON response. Second, since a custom conversational application was chosen for Dialogflow, only the 'DefaultWelcomeIntent' and 'DefaultFallbackIntent' existed. In order to create a Google Assistant application which suited this project, Custom Entities were added which provided visibility to the four types of emotions being investigated in the user's speech. These were all mapped to the correct follow-up intents and fulfilment methods where appropriate once the Webhook code had been set up and linked with Dialogflow. Finally, the last important service, Request, is a simple Node.js library which aids with making HTTP requests and sending them to the desired API endpoints. This library was influential for this project since it enabled the possibility of not requiring non-functioning or non-existent Spotify methods such as requesting an updated Access token or playing the music through the Spotify application.

In order to evaluate this project, the extent to which the requirement specifications were fulfilled, the results from testing, and the extent to which the application solved the problem which this project aimed to address, were examined. First, the entirety of the high priority requirements have been fulfilled and, based on the somewhat limited testing results, the system performs relatively well in meeting the more subjective requirements. For Future Development, more testing would be required with a particular focus on this area along with multimodal field research of unexamined Use Cases. Before this can happen, it remains the highest priority for Spotify to allow submission of the application on their website so that other users can use the application from their own Spotify accounts. Despite Spotify's relatively hostile development facility, Google's contrastingly accommodating developer's console has allowed the app to be released both as a Beta version for testing and as a full production release; They even provided the developer of this project with a free Google T-shirt for making a production-approved application! Since the full production release is available to use by any Google Assistant user, further rounds of testing will be easy to conduct once Spotify begin allowing new app submissions again. The completion of the

application to its production release should certainly be viewed among the successes of this project.

The main problem which this project aimed to address was that there was no system on any smart speaker device or music streaming service which accounted for song lyrics in categorising songs into certain moods or emotions. This application successfully does this with the added benefit of providing further audio analysis in addition to an analysed excerpt of the song lyrics in the song. The user is given the information with which to understand decisions and the freedom to make these informed decisions. The secondary aim of the project was to provide a Music Therapy application which is more beneficial to users than the existing mood categorisation systems. The system provides a 'person' to speak to who provides soothing words for any of the four emotions along with song recommendations; Judging whether the Music Therapy provided by this application is more beneficial than other mood-finding services or advice-giving services requires more testing, however the ability to provide both within the same system is an achievement in and of itself. Detracting from this to an extent, however, is the accuracy of the sentiment tone analysers used. Although the tests were largely positive for the closeness of the matching of the recommended song to the desired emotion, there were anomalies. The song, 'All By Myself' for instance, was arguably falsely classified as 'confident' instead of 'sad' for instance. This inaccuracy constitutes one of the main limitations of this project and perhaps shows a need for an improvement in sentiment tone analysis technologies used. Further, another limitation from the Watson Tone Analyser stems from the limited nature of API calls within the free IBM Cloud account; Without this limitation, the database used in this project could have been much larger, improving the selection of songs that users have access to. Despite this, the high average rate of accuracy should remain as an achievement and the anomalies in addition to the limitations of the free IBM Cloud account should be used as a guide on where to focus for future developments.

The design choices for the application were formed partly through extensive research and partly through pragmatism. The initial design choices were outlined during the planning process at the beginning of the project. For the most part, these design choices remained unchanged with some exceptions. It was also here where the order of task completion was outlined, starting with the database, followed by the Google Assistant application and finished by connecting the Webhook to Spotify. First, IBM's Watson Tone Analyser was found to be the most accurate sentiment tone analyser through research and MySQL was reportedly the most user friendly way to develop a hosted database. Originally, MySQL was to be used as the main database for connecting to the Webhook but, through pragmatically searching for the most accessible, efficient and scalable database, it was decided that Google's Firebase was more appropriate for this project. The requirement for the Google Assistant Application was known from the start but the actual design of the application itself changed over the course of development based off the 5 second timeout for intents and the results from initial testing. Lastly, Spotify was chosen as the music streaming service endpoint due to its extensive audio analysis and large library. All of these design choices came with various benefits and limitations.

One such limitation as a result of the architecture and services used stems from the lack of full account connectivity between the Spotify and Google APIs. The application requires, as a result of the need for full authorisation, that first time users visit a cloud hosted webpage in order to connect their two accounts. This provides accessibility issues where first time users might not have the option to access a website, for example if they are just using the Google

home hub via voice chat alone without a nearby computer or mobile device. Despite this, the project should be commended on its high availability following successful account syncing; A full list of available devices can be found in the application's download page on the Google Assistant website (Appendix G). Furthermore, the server less cloud-based architectures used throughout this project offer enhanced scalability. Within this architecture, the emphasis is not on the developers themselves to maintain the hosting of the application; Instead, the Google cloud handles all ingoing and outgoing API calls and function executions meaning the application can be used by as many users as Google's architecture can cater for. The architectures used also offer cost and security benefits for the same reason. This amounts to an application which can be used by an extremely large user base without any costs incurred by the developers themselves in addition to a broad number of scalability opportunities which cloud hosted applications allow.

To conclude, the project has fulfilled all of the system specifications and requirements, with varying degrees of fulfilment in the more subjective areas. It falls victim to a small number of anomalies when analysing the tones used in songs, however, its' relatively high success rate amounts to an overall application success with a focus on the areas to improve in future developments. Moreover, rigorous testing is necessary to assess the system's use as a Music Therapy app since this was not possible within the scopes of this project but it should perhaps be commended on achieving the specifications of providing a means for users to finally receive songs based on lyrics analysis in addition to allowing users to express how they are feeling; Something which no other service offers. Finally, due to the architectures used, the system is fast, highly available and scalable, providing an important stepping stone on which future developments can be made.

9. Bibliography

Ali, S. O. and Peynircioglu, A. . (2006) 'Songs and emotions: are lyrics and melodies equal partners?', *Psychology of Music*, 34(4), pp. 511–534. doi: 10.1177/0305735606067168.

Bailey, L. M. (1986) 'Music therapy in pain management', *Journal of Pain and Symptom Management*, 1(1), pp. 25–28. doi: 10.1016/S0885-3924(86)80024-0.

Bunt, L. (1994) 'Music Therapy: An Art Beyond Words', *BMJ*, 308(6937), p. 1175. doi: 10.1136/bmj.308.6937.1175a.

Casey, H. (2019) *Apple Music vs Spotify: Which Is the Best Music Service? | Tom's Guide*. Available at: https://www.tomsguide.com/us/apple-music-vs-spotify,news-21069.html (Accessed: 22 August 2019).

Castillo-Pérez, S. *et al.* (2010) 'Effects of music therapy on depression compared with psychotherapyNo Title', *The Arts in Psychotherapy*, 37(5), pp. 387–390.

Chancellor, B., Duncan, A. and Chatterjee, A. (2014) 'Art Therapy for Alzheimer's Disease and Other Dementias', *Journal of Alzheimer's Disease*. IOS Press, 39(1), pp. 1–11. doi: 10.3233/JAD-131295.

Chen, Y., Liu, S. and Nagoya, F. (2005) 'An Approach to Integration Testing Based on Data Flow Specifications. In: Liu Z., Araki K. (eds) Theoretical Aspects of Computing - ICTAC 2004. ICTAC 2004. Lecture Notes in Computer Science', in. Springer. Available at: https://link.springer.com/content/pdf/10.1007%2Fb107116.pdf (Accessed: 28 August 2019).

Choi, J., Song, J. H. and Kim, Y. (2018) 'An analysis of music lyrics by measuring the distance of emotion and sentiment', in *Proceedings - 2018 IEEE/ACIS 19th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD 2018*. Institute of Electrical and Electronics Engineers Inc., pp. 176–181. doi: 10.1109/SNPD.2018.8441085.

codebeautify (2016) *Best online CSV to JSON and CSV to XML Converter/Transformer tool*. Available at: https://codebeautify.org/csv-to-xml-json# (Accessed: 26 August 2019).

Davis, W. B., Gfeller, K. E. and Thaut, M. H. (2008) *An Introduction to Music Therapy: Theory and Practice*. American Music Therapy Association.

DevTips (2018) *Web Scraping with Node.js & Puppeteer (⛵ rants included, no extra charge) - YouTube*. Available at: https://www.youtube.com/watch?v=pixfH6yyqZk (Accessed: 27 August 2019).

DigitalTrends (2019) *Apple Music vs. Spotify | Which Service is the Streaming King? | Digital Trends*. Available at: https://www.digitaltrends.com/music/apple-music-vs-spotify/ (Accessed: 2 September 2019).

Doerrfeld, B. (2015) *20+ Emotion Recognition APIs That Will Leave You Impressed, and Concerned | Nordic APIs |*. Available at: https://nordicapis.com/20-emotion-recognition-apis-that-will-leave-you-impressed-and-concerned/ (Accessed: 21 August 2019).

Ekman, P. and Friesen, W. V. (1971) 'Constants Across Cultures in The Face and Emotion', *Journal of Personality and Social Psychology*, 17(2), pp. 124–129.

Fellbaum, C. (1998) *WordNet : an electronic lexical database*. MIT Press. Available at: https://books.google.co.uk/books?hl=en&lr=&id=Rehu8OOzMIMC&oi=fnd&pg=PR11&dq=Christiane+Fellbaum+(1998,+ed.)+WordNet:+An+Electronic+Lexical+Database.+Cambridge,+MA:+MIT+Press.&ots=IrmeLhRRf8&sig=ZTI3wBT99vReo-O85Ab0M9zdK5o#v=onepage&q=Christiane Fellbaum (1998%2C ed.) WordNet%3A An Electronic Lexical Database. Cambridge%2C MA%3A MIT Press.&f=false (Accessed: 1 September 2019).

Feng, Y., Zhuang, Y. and Pan, Y. (2003) 'Popular music retrieval by detecting mood', in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval - SIGIR '03*. New York, New York, USA: ACM Press, p. 375. doi: 10.1145/860435.860508.
Freud, S. (1940) 'An Outline of Psycho-Analysis', *The International Journal of Psychoanalysis*, 21, pp. 27–85. Available at: http://icpla.edu/wp-content/uploads/2012/10/Freud-S.-An-Outline-of-Psychoanalysis-Int.-JPA.pdf (Accessed: 8 September 2019).
Garnier, N. (2016) *Contributors to firebase/functions-samples*. Available at: https://github.com/firebase/functions-samples/graphs/contributors (Accessed: 22 August 2019).
Google (2019a) *Cloud Speech-to-Text - Speech Recognition | Cloud Speech-to-Text | Google Cloud*. Available at: https://cloud.google.com/speech-to-text/ (Accessed: 2 September 2019).
Google (2019b) *Contexts | Dialogflow Documentation | Google Cloud*. Available at: https://cloud.google.com/dialogflow/docs/contexts-overview (Accessed: 27 August 2019).
Google (2019c) *Custom Conversational Actions | Actions on Google | Google Developers*. Available at: https://developers.google.com/actions/conversational/overview (Accessed: 26 August 2019).
Google (2019d) *Entities | Dialogflow Documentation | Google Cloud*. Available at: https://cloud.google.com/dialogflow/docs/entities-overview (Accessed: 26 August 2019).
Google (2019e) *How fulfillment works | Dialogflow Documentation | Google Cloud*. Available at: https://cloud.google.com/dialogflow/docs/fulfillment-how (Accessed: 27 August 2019).
Google (2019f) *Introduction | Actions on Google | What is an Action?* Available at: https://developers.google.com/actions/overview (Accessed: 2 September 2019).
Gou, L. *et al.* (2014) 'KnowMe and ShareMe', in *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*. New York, New York, USA: ACM Press, pp. 955–964. doi: 10.1145/2556288.2557398.
HeadsTogether (2019) *10 Things to Say to Someone with Depression | Heads Together*. Available at: https://www.headstogether.org.uk/10-things-to-say-to-someone-with-depression/ (Accessed: 3 September 2019).
Hirst, G. and Liu, B. (2012) 'Sentiment Analysis and Opinion Mining', in *Synthesis Lectures on Human Language Technologies* . Morgan and Claypool, pp. 1–167. doi: 10.2200/S00416ED1V01Y201204HLT016.
Hu, X. *et al.* (2008) *ISMIR 2008 : proceedings of the 9th International Conference of Music Information Retrieval*. [Drexel University].
IBM (2017) *Tone Analyzer - IBM Cloud API Docs*. Available at: https://cloud.ibm.com/apidocs/tone-analyzer (Accessed: 26 August 2019).
IBM (2019) *The science behind the service*. Available at: https://cloud.ibm.com/docs/services/tone-analyzer?topic=tone-analyzer-ssbts#the-science-behind-the-service (Accessed: 2 September 2019).
JSONLint (2019) *JSONLint - The JSON Validator*. Available at: https://jsonlint.com/ (Accessed: 26 August 2019).
Katayose, H., Imai, M. and Inokuchi, S. (1988) 'Sentiment Extraction in Music', in *9th International Conference on Pattern Recognition*, pp. 1083–1087. doi: 10.1109/ICPR.1988.28447.
Kinsella, B. (2019) *Smart Speaker Installed Base to Surpass 200 Million in 2019, Grow to 500 Million in 2023 - Canalys - Voicebot*. Available at: https://voicebot.ai/2019/04/15/smart-speaker-installed-base-to-surpass-200-million-in-2019-grow-to-500-million-in-2023-canalys/

(Accessed: 8 September 2019).
Klayman, J. (1995) 'Varieties of Confirmation Bias', *Psychology of Learning and Motivation*. Academic Press, 32, pp. 385–418. doi: 10.1016/S0079-7421(08)60315-1.
Lambert, M. J. and Vermeersch, D. A. (2002) 'Effectiveness of Psychotherapy', *Encyclopedia of Psychotherapy*, 1, pp. 709–714.
Laurier, C., Grivolla, J. and Herrera, P. (2008) 'Multimodal music mood classification using audio and lyrics', in *Proceedings - 7th International Conference on Machine Learning and Applications, ICMLA 2008*, pp. 688–693. doi: 10.1109/ICMLA.2008.96.
LedgerNote (2019) *Musical Key Characteristics & Emotions*. Available at: https://ledgernote.com/blog/interesting/musical-key-characteristics-emotions/ (Accessed: 27 August 2019).
Li, K. and Nellis, S. (2019) *Apple Music's U.S. subscriber count overtakes Spotify: source - Reuters*. Available at: https://uk.reuters.com/article/us-apple-spotify-tech/apple-musics-u-s-subscriber-count-overtakes-spotify-source-idUKKCN1RH246 (Accessed: 1 September 2019).
Logan, B., Kositsky, A. and Moreno, P. (2004) 'Semantic analysis of song lyrics', in *2004 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 827–830.
Mdooxey06 (2017) *Receiving 400 error with authorizationCodeGrant*. Available at: https://github.com/thelinmichael/spotify-web-api-node/issues/116.
Miller, G. . (1995) 'WordNet: A Lexical Database for English George A. Miller', *Communications of the ACM*, 38(11), pp. 39–41. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.83.1823&rep=rep1&type=pdf (Accessed: 1 September 2019).
Morris, R. (2007) 'Mathematics and the Twelve-Tone System: Past, Present, and Future'. Springer, Berlin, Heidelberg, pp. 266–288. doi: 10.1007/978-3-642-04579-0_27.
Murdoch, D. (2014) *DanielMurdoch/SentiWord: SentiWordNet based sentiment analysis for node.js focusing on accuracy and POS analysis.* Available at: https://github.com/DanielMurdoch/SentiWord (Accessed: 27 August 2019).
MySQL Fix ERROR 1148: The used command is not allowed with this MySQL version - YouTube (2018) *MySQL Fix ERROR 1148: The used command is not allowed with this MySQL version - YouTube*. Available at: https://www.youtube.com/watch?v=XM2xx-PD4cg&vl=en (Accessed: 23 August 2019).
Neumayer, R. and Rauber, A. (2007) 'Integration of Text and Audio Features for Genre Classification in Music Information Retrieval Robert', in *Advances in Information Retrieval*. Springer, pp. 724–727.
Neuralink (2019) *Neuralink*. Available at: https://www.neuralink.com/ (Accessed: 1 September 2019).
O'callaghan, C. *et al.* (2009) 'Resounding attachment: cancer inpatients' song lyrics for their children in music therapy', *Support Care Cancer*, (17), pp. 1149–1157. doi: 10.1007/s00520-008-0555-5.
Opesanya, B. (2018) *Getting Started with IBM Watson Tone Analyzer - codeburst*. Available at: https://codeburst.io/getting-started-with-ibm-watson-tone-analyzer-3aa3386cff15 (Accessed: 26 August 2019).
Perez, S. (2019) *Google falls to third place in worldwide smart speaker market | TechCrunch*. Available at: https://techcrunch.com/2019/08/26/google-falls-to-third-place-in-worldwide-smart-speaker-market/ (Accessed: 1 September 2019).
Perrin, V. (2018) *IBM/watson-google-assistant: Create an Action for Google Assistant using Watson Assistant*. Available at: https://github.com/IBM/watson-google-assistant (Accessed: 26 August 2019).
Purwins, H. (2005) *Profiles of Pitch Classes Circularity of Relative Pitch and Key-*

Experiments, Models, Computational Music Analysis, and Perspectives vorgelegt von Diplom-Mathematiker Hendrik Purwins aus Münster. Berlin: Elektrotechnik und Informatik der Technischen Universität Berlin. Available at: https://depositonce.tu-berlin.de/bitstream/11303/1499/1/Dokument_8.pdf (Accessed: 27 August 2019).
Quinn, H. (2018) *25 Things People Said That Were Actually Code For 'I'm Depressed' | HuffPost Life*. Available at: https://www.huffpost.com/entry/25-things-people-said-that-were-actually-code-for-im_b_5ba24edbe4b086ac5a9e0884?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_sig=AQAAAGgqh9BWAYv8Dt350fi7zgHr1xhp8qSg8I2XkUH2t1bkmoNd0uD388e5PMw3dyA (Accessed: 3 September 2019).
Rashwan, A. (2012) 'Semantic Analysis of Functional and Non-Functional Requirements in Software Requirements Specifications', in. Available at: https://link.springer.com/content/pdf/10.1007%2F978-3-642-30353-1.pdf (Accessed: 28 August 2019).
Reason, J. T. (1990) *Human error*. Cambridge University Press. Available at: https://books.google.co.uk/books?hl=en&lr=&id=WJL8NZc8lZ8C&oi=fnd&pg=PR9&dq=human+error+questionnaire&ots=AmSm4dbh3e&sig=MKJw-8AYZGk5v-zvQwQcVbsvH7A&redir_esc=y#v=onepage&q=questionnaire&f=false (Accessed: 2 September 2019).
Rennie, J. D. M. *et al.* (2003) *Tackling the Poor Assumptions of Naive Bayes Text Classifiers*.
Saluja, V., Jain, M. and Yadav, P. (2019) 'L,M&A: An Algorithm for Music Lyrics Mining and Sentiment Analysis', in. EasyChair, pp. 475–465. doi: 10.29007/wkj6.
Schäfer, T. *et al.* (2013) 'The psychological functions of music listening.', *Frontiers in psychology*. Frontiers Media SA, 4, p. 511. doi: 10.3389/fpsyg.2013.00511.
Sindhgatta, R. and Thonse, S. (2005) 'Functional and Non-functional Requirements Specification for Enterprise Applications', *Lecture Notes in Computer Science.In: Bomarius F., Komi-Sirviö S. (eds) Product Focused Software Process Improvement. PROFES*. Springer, 3547. Available at: https://link.springer.com/content/pdf/10.1007%2Fb137178.pdf (Accessed: 28 August 2019).
Solsman, J. E. (2019) *Spotify keeps big lead over Apple Music but disappoints with 108M subscribers - CNET*. Available at: https://www.cnet.com/news/spotify-subscribers-earnings-hit-108-million-apple-music/ (Accessed: 1 September 2019).
Spotify (2019a) *Authorization Guide | Spotify for Developers*. Available at: https://developer.spotify.com/documentation/general/guides/authorization-guide/ (Accessed: 27 August 2019).
Spotify (2019b) *Start/Resume a User's Playback | Spotify for Developers*. Available at: https://developer.spotify.com/documentation/web-api/reference/player/start-a-users-playback/ (Accessed: 27 August 2019).
Sun, H. (2019) 'Case Study—Spotify', in *Digital Revolution Tamed*. Cham: Springer International Publishing, pp. 135–170. doi: 10.1007/978-3-319-93022-0_5.
Taras, V., Steel, P. and Kirkman, B. L. (2010) 'Negative practice-value correlations in the GLOBE data: Unexpected findings, questionnaire limitations and research directions', *Journal of International Business Studies*, pp. 1330–1338. doi: 10.1057/jibs.2010.30.
Thelin, M. (2014) *thelinmichael/spotify-web-api-node: A Node.js wrapper for Spotify's Web API.* Available at: https://github.com/thelinmichael/spotify-web-api-node (Accessed: 27 August 2019).
Thomas, H. and Ogden, M. D. (1997) 'Some thoughts on the use of language in psychoanalysis', *Psychoanalytic Dialogues The International Journal of Relational Perspectives* , 7(1), pp. 1–21. doi: 10.1080/10481889709539164.
w3schools (2019) *Node.js MySQL*. Available at:

https://www.w3schools.com/nodejs/nodejs_mysql.asp (Accessed: 24 August 2019).
Wikipedia (2005) *MoodLogic*. Washington, D.C. Available at: https://en.wikipedia.org/wiki/MoodLogic (Accessed: 12 July 2019)
Winchurch, E. (2018) *IBM Watson Tone Analyzer Reviews 2019: Details, Pricing, & Features | G2*. Available at: https://www.g2.com/products/ibm-watson-tone-analyzer/reviews#reviews (Accessed: 21 August 2019).
XAPPmedia (2018) *What is Google Assistant?* Available at: https://xappmedia.com/what-is-google-assistant/ (Accessed: 2 September 2019).
Yang, Y.-H. *et al.* (2008) 'Toward Multi-Modal Music Emotion Classification', *Advances in Multimedia Information Processing - PCM 2008*, pp. 72–79.

10. Appendices

Appendix A – Glossary of terms

API – an Application Programming Interface which is a communication protocol between clients and servers.
API Endpoint – an endpoint is one end of the communication channel between several communication APIs. Endpoints include the URL of a server or service.
Dialogflow – the Google-owned developer of human-computer interaction technologies based on natural language conversations.
Google Assistant – an artificial intelligence-powered virtual assistant comes pre-installed on android and Google Home devices such as smart speakers and mobile phones.
JSON – an acronym of JavaScript Object Notation. This is a standard file format which uses readable text for transmitting data objects consisting of attribute-value pairs and array data types.
IBM's Watson Tone Analyser – a sentiment tone analyser which uses linguistic analysis to detect emotional and language tones in written text.
JavaScript – a high level interpreted scripting language that conforms to the ECMAScript specification.
MySQL – an open source relational database management system.
MySQL Workbench – an application which allows users to view, alter and create tabular data structures on a hosted MySQL database.
Node.js – a JavaScript runtime environment that executes JavaScript code. It consists of many 'modules' which are pre-written JavaScript libraries which can be used without any further installations.
Node.js Wrapper – a library that provides a convenient interface to a service or library by 'wrapping' the native API into a more convenient form.
noSQL – a database which does not fit under the usual SQL tabular relations structure.
Postman – an API client tool which helps users to test their API requests in a suitable environment.
Psycholinguistics – the study of the relationships between linguistic behaviour and psychological processes.
Regex – an abbreviation for Regular Expression. These are a sequence of characters that define a specific search pattern. These are often used to extract specific parts of text.

SentiWordNet – a lexical resource for opinion mining, used within this project via the Node.js library ‘sentiword’.
Spotify – a global music streaming service.

Appendix B - Node.js Libraries and Dependencies Used

"actions-on-google": "2.1.3": a fulfilment library for Actions on Google.
"cookie-parser": "^1.4.3": a Node.js library that parses and stores cookies sent between servers which, in this project, is used to store the verification state after signing in to Spotify via the hosted URL sign-in page.
"crypto": "0.0.3": a module which provides cryptographic functionality allowing the safe passage of authorisation codes and other sensitive data in this project.
"dialogflow-fulfillment": "0.4.1": a library used for connecting a Dialogflow application to its Webhook.
"firebase-admin": "7.1.1": a Node.js Software Developer Kit which enables access to Firebase services from privileged environments such as servers or clouds.
"firebase-functions": "^3.2.0": a package which provides a Software Developer Kit for defining Cloud Functions for Firebase.
"mysql": "^2.17.1": used for creating the connection between the JavaScript code and the SQL database. Containts methods such as ‘.query’ which performs SQL queries on the SQL database.
"request": "^2.88.0": an HTTP and HTTPS Node.js wrapper to aid in making HTTP calls.
"sentiword": "0.0.1" - a Node.js wrapper which accesses the SentiWordNet API and outputs the positivity/negativity score, n-grams and individual n-gram scores of the inputted text.
"spotify-web-api-node": "^2.3.6” – a Node.js wrapper which accesses the Spotify API. It is used in this project to provide audio analysis and audio feature analysis; Playing the music and requesting authorisation codes are completed through HTTP requests.
"watson-developer-cloud": "^4.0.1": used to establish connection with the Watson Tone Analyser IBM cloud hosted service.

Appendix C

i) Angry Unit Tests

AngryChecker & -Angry (with 'Yes' reply to confirm)							
Intent: **-AngryChecker**				Intent: **-Angry**			
User Speech	100% Recorded?	Expected Result	Actual Result	User Speech	100% Recorded?	Expected Result	Actual Result
I feel angry	Yes	I'm detecting that you're angry, is this correct?	I'm detecting that you're angry, is this correct?	Yes	Yes	I'm sorry to hear that you're angry! Would you like to listen to an angry song together?	Don't feel Angry! Let us blow off some steam with an angry song, shall we?
I'm annoyed	Yes	I'm detecting that you're angry, is this correct?	I'm detecting that you're angry, is this correct?	Yes	Yes	I'm sorry to hear that you're angry! Would you like to listen to an angry song together?	I'm sorry to hear that you're angry! Would you like to listen to an angry song together?
Angry	Yes	I'm detecting that you're angry, is this correct?	I'm detecting that you're angry, is this correct?	Yes	Yes	I'm sorry to hear that you're angry! Would you like to listen to an angry song together?	I'm sorry to hear that you're angry! Would you like to listen to an angry song together?
Pissed Off	Yes	I'm detecting that you're angry, is this correct?	I'm detecting that you're angry, is this correct?	Yes	Yes	I'm sorry to hear that you're angry! Would you like to listen to an angry song together?	Don't feel Angry! Let us blow off some steam with an angry song, shall we?
Outraged	Yes	I'm detecting that you're angry, is this correct?	I'm detecting that you're angry, is this correct?	Yes	Yes	I'm sorry to hear that you're angry! Would you like to listen to an angry song together?	I'm sorry to hear that you're angry! Would you like to listen to an angry song together?

I've been stuck in traffic	Yes	I'm detecting that you're angry, is this correct?	I'm detecting that you're angry, is this correct?	Yes	Yes	I'm sorry to hear that you were stuck in traffic, wanna listen to some angry songs together to relieve some stress?	I'm sorry to hear that you were stuck in traffic, wanna listen to some angry songs together to relieve some stress?
Boris is now the Prime Minister	Yes	I'm detecting that you're angry, is this correct?	I'm detecting that you're angry, is this correct?	Yes	Yes	Boris is not the Prime Minister? I am as annoyed as you are. Let's listen to some angry music to think about our country's future together shall we?	Boris is not the Prime Minister? I am as annoyed as you are. Let's listen to some angry music to think about our country's future together shall we?
Liverpool drew against Millwall…	Yes	I'm detecting that you're angry, is this correct?	I'm detecting that you're angry, is this correct?	Yes	Yes	I'm so sorry that your team drew, wanna listen to some music to help release some steam?	I'm so sorry that your team drew, wanna listen to some music to help release some steam?
We lost against Peterborough today!	Yes	I'm detecting that you're angry, is this correct?	I'm detecting that you're angry, is this correct?	Yes	Yes	I'm so sorry that your team lost, wanna listen to some music to help release some steam?	I'm so sorry that your team lost, wanna listen to some music to help release some steam?

AngryChecker & AngryChecker-no (with 'No' reply to confirm)	
Intent: **-AngryChecker**	Intent: **-Angry**

User Speech	100% Recorded?	Expected Result	Actual Result	User Speech	100% Recorded?	Expected Result	Actual Result
I feel angry	Yes	I'm detecting that you're angry, is this correct?	I'm detecting that you're angry, is this correct?	No	Yes	I'm sorry, I'm still learning. Which of the following four emotions would you like me to play a song from? Angry, Confident, Happy or sad?	I'm sorry, I'm still learning. Which of the following four emotions would you like me to play a song from? Angry, Confident, Happy or sad?

-Angry-Yes & -Angry-No			
			Intent: **-Angry-yes**
User Speech	100% Recorded?	Expected Result	Actual Result
Yes	Yes	I recommend $song by $artist. Would you like to hear the audio analysis?	I recommend Suck My Kiss by Red Hot Chili Peppers. Would you like to hear audio analysis of Suck My Kiss?
			Intent: **-Angry-yes**
No	Yes	Ok, let me know if you need me. END CONVERSATION	I'll be here if you need me for anything at all! *END CONVERSATION*

-Angry-Yes-No			
			Intent: **-Angry-yes-no**
User Speech	100% Recorded?	Expected Result	Actual Result
No	Yes	Ok, shall I still play it?	Ok, would you still like me to play it?
			Intent: **-Angry-yes-no-chooseAgain**
Choose another	Yes	Ok, I'll find another song. Just a moment. *REVERT BACK TO -ANGRY INTENT*	-nothing-
			Intent: **-Angry-yes-no-Yes**
Yes	Yes	Enjoy! *PLAY MUSIC, END CONVERSATION*	

-Angry-yes-yes & -Angry-yes-yes-yes			
			Intent: **-Angry-yes-yes**
User Speech	100% Recorded?	Expected Result	Actual Result

Yes	Yes	The tempo of the song is… It is in the key… this key evokes… emotions. Would you still like me to play it?	The song has a medium tempo with 102 beats per minute. It is in the key of C major... Songs that are written in the key of C major often evoke happy and uplifting emotions... Would you still like to listen to this song?
			Intent: **-Angry-yes-yes-yes**
Yes	Yes	Enjoy! *PLAY MUSIC, END CONVERSATION*	Enjoy! *PLAY MUSIC, END CONVERSATION*
-Angry-Yes-Yes-No & -Angry-Yes-Yes-No-Yes & -Angry-Yes-Yes-Yes-No-No			
			Intent: **-Angry-yes-yes-no**
User Speech	100% Recorded?	Expected Result	Actual Result
No	Yes	Would you like another song of the same emotion instead?	
			Intent: **-Angry-yes-yes-no-yes**
Yes	Yes	Ok, I'll find another song. Just a moment. *REVERT BACK TO -ANGRY INTENT*	
			Intent: **-Angry-yes-yes-no-no**
No	Yes	Ok, let me know if you need me. END CONVERSATION	Ok ask for me if you want another song. *END CONVERSATION*

ii) Sad Unit Tests

SadChecker & -Sad (with 'Yes' reply to confirm)							
Intent: **-SadChecker**				Intent: **-Sad**			
User Speech	100% Recorded?	Expected Result	Actual Result	User Speech	100% Recorded?	Expected Result	Actual Result
I feel sad	Yes	I'm detecting that you're sad, is this correct?	I'm detecting that you're sad, is this correct?	Yes	Yes	I'm sorry to hear that you're sad! Would you like to listen to a sad song together?	I'm sorry to hear that you're sad! Would you like to listen to a sad song together?

I'm depressed	Yes	I'm detecting that you're sad, is this correct?	I'm detecting that you're sad, is this correct?	Yes	Yes	I'm sorry to hear that you're sad! Would you like to listen to a sad song together?	I'm sorry to hear that you're sad! Would you like to listen to a sad song together?
Sad	Yes	I'm detecting that you're sad, is this correct?	I'm detecting that you're sad, is this correct?	Yes	Yes	I'm sorry to hear that you're sad! Would you like to listen to a sad song together?	I'm sorry to hear that you're sad! Would you like to listen to a sad song together?
Down	Yes	I'm detecting that you're sad, is this correct?	I'm detecting that you're sad, is this correct?	Yes	Yes	I'm sorry to hear that you're sad! Would you like to listen to a sad song together?	I'm sorry to hear that you're sad! Would you like to listen to a sad song together?
Miserable	Yes	I'm detecting that you're sad, is this correct?	I'm detecting that you're sad, is this correct?	Yes	Yes	I'm sorry to hear that you're sad! Would you like to listen to a sad song together?	I'm sorry to hear that you're sad! Would you like to listen to a sad song together?
My cat died	Yes	I'm detecting that you're sad, is this correct?	I'm detecting that you're sad, is this correct?	Yes	Yes	I'm sorry to hear about your pet passing away. It might help if we listen to some sad music together - would you like that?	I'm sorry to hear about your pet passing away. It might help if we listen to some sad music together - would you like that?
My brother is unwell	Yes	I'm detecting that you're sad, is this correct?	I'm detecting that you're sad, is this correct?	Yes	Yes	I'm sorry to hear about your brother being unwell. It might help if we listen to some sad music together - would you like that?	I'm sorry to hear about your brother being unwell. It might help if we listen to some sad music together - would you like that?

I'm having car troubles	Yes	I'm detecting that you're sad, is this correct?	I'm detecting that you're sad, is this correct?	Yes	Yes	I'm sorry that you're having car troubles. Would you like to hear a sad song to help get you through it?	I'm sorry that you're having car troubles. Would you like to hear a sad song to help get you through it?
My girlfriend broke up with me	Yes	I'm detecting that you're sad, is this correct?	I'm detecting that you're sad, is this correct?	Yes	Yes	I'm sorry to hear about your break up. Would you like to hear a sad song to help get you through it?	I'm sorry to hear about your break up. Would you like to hear a sad song to help get you through it?

SadChecker & SadChecker-no (with 'No' reply to confirm)							
Intent: **-SadChecker**				Intent: **-Sad**			
User Speech	100% Recorded?	Expected Result	Actual Result	User Speech	100% Recorded?	Expected Result	Actual Result
I feel sad	Yes	I'm detecting that you're sad, is this correct?	I'm detecting that you're sad, is this correct?	No	Yes	I'm sorry, I'm still learning. Which of the following four emotions would you like me to play a song from? Angry, Confident, Happy or sad?	I'm sorry, I'm still learning. Which of the following four emotions would you like me to play a song from? Angry, Confident, Happy or sad?

-Sad-Yes & -Sad-No			
			Intent: **-Sad-yes**
User Speech	100% Recorded?	Expected Result	Actual Result
Yes	Yes	I recommend $song by $artist. Would you like to hear the audio analysis?	I recommend $song by $artist. Would you like to hear the audio analysis?
			Intent: **-Sad-yes**

No	Yes	Ok, let me know if you need me. END CONVERSATION	Ok, let me know if you need me. END CONVERSATION
-Sad-Yes-No			
			Intent: **-Sad-yes-no**
User Speech	100% Recorded?	Expected Result	Actual Result
No	Yes	Ok, shall I still play it?	Ok, shall I still play it?
			Intent: **-Sad-yes-no-chooseAgain**
Choose another	Yes	Ok, I'll find another song. Just a moment. *REVERT BACK TO -Sad INTENT*	Ok, I'll find another song. Just a moment. *REVERT BACK TO -Sad INTENT*
			Intent: **-Sad-yes-no-Yes**
Yes	Yes	Enjoy! *PLAY MUSIC, END CONVERSATION*	Enjoy! *PLAY MUSIC, END CONVERSATION*
-Sad-yes-yes & -Sad-yes-yes-yes			
			Intent: **-Sad-yes-yes**
User Speech	100% Recorded?	Expected Result	Actual Result
Yes	Yes	The tempo of the song is… It is in the key… this key evokes… emotions. Would you still like me to play it?	The tempo of the song is… It is in the key… this key evokes… emotions. Would you still like me to play it?
			Intent: **-Sad-yes-yes-yes**
Yes	Yes	Enjoy! *PLAY MUSIC, END CONVERSATION*	Enjoy! *PLAY MUSIC, END CONVERSATION*
-Sad-Yes-Yes-No & -Sad-Yes-Yes-No-Yes & -Sad-Yes-Yes-Yes-No-No			
			Intent: **-Sad-yes-yes-no**
User Speech	100% Recorded?	Expected Result	Actual Result
No	Yes	Would you like another song of the same emotion instead?	Would you like another song of the same emotion instead?
			Intent: **-Sad-yes-yes-no-yes**
Yes	Yes	Ok, I'll find another song. Just a moment. *REVERT BACK TO -SAD INTENT*	Ok, I'll find another song. Just a moment. *REVERT BACK TO -SAD INTENT*
			Intent: **-Sad-yes-yes-no-no**

No	Yes	Ok, let me know if you need me. END CONVERSATION	Ok, let me know if you need me. END CONVERSATION

iii) Happy Unit Tests

HappyChecker & -Happy (with 'Yes' reply to confirm)							
Intent: **-HappyChecker**				Intent: **-Happy**			
User Speech	100% Recorded?	Expected Result	Actual Result	User Speech	100% Recorded?	Expected Result	Actual Result
I feel happy	Yes	I'm detecting that you're Happy, is this correct?	I'm detecting that you're Happy, is this correct?	Yes	Yes	I feel happy too! Shall we jam out to some happy music together?	I feel happy too! Shall we jam out to some happy music together?
I'm happy	Yes	I'm detecting that you're Happy, is this correct?	I'm detecting that you're Happy, is this correct?	Yes	Yes	I feel happy too! Shall we jam out to some happy music together?	I feel happy too! Shall we jam out to some happy music together?
Happy	Yes	I'm detecting that you're Happy, is this correct?	I'm detecting that you're Happy, is this correct?	Yes	Yes	I feel happy too! Shall we jam out to some happy music together?	I feel happy too! Shall we jam out to some happy music together?
Great mood	Yes	I'm detecting that you're Happy, is this correct?	I'm detecting that you're Happy, is this correct?	Yes	Yes	I feel happy too! Shall we jam out to some happy music together?	I feel happy too! Shall we jam out to some happy music together?

Not too bad	Yes	I'm detecting that you're Happy, is this correct?	I'm detecting that you're Happy, is this correct?	Yes	Yes	I feel happy too! Shall we jam out to some happy music together?	I feel happy too! Shall we jam out to some happy music together?
Pretty great	Yes	I'm detecting that you're Happy, is this correct?	I'm detecting that you're Happy, is this correct?	Yes	Yes	I feel happy too! Shall we jam out to some happy music together?	I feel happy too! Shall we jam out to some happy music together?
Liverpool won today!	Yes	I'm detecting that you're Happy, is this correct?	I'm detecting that you're Happy, is this correct?	Yes	Yes	Wow congratulations on your team winning! Shall we celebrate with some good-feeling tunes?	Wow congratulations on your team winning! Shall we celebrate with some good-feeling tunes?
It's my birthday	Yes	I'm detecting that you're Happy, is this correct?	I'm detecting that you're Happy, is this correct?	Yes	Yes	Wow congratulations on your birthday! Shall we celebrate with some good-feeling tunes?	Wow congratulations on your birthday! Shall we celebrate with some good-feeling tunes?
I'm looking forward to going on holiday!	Yes	I'm detecting that you're Happy, is this correct?	I'm detecting that you're Happy, is this correct?	Yes	Yes	Wow congratulations on your holiday! Shall we celebrate with some good-feeling tunes?	Wow congratulations on your holiday! Shall we celebrate with some good-feeling tunes?

I got engaged!	Yes	I'm detecting that you're Happy, is this correct?	I'm detecting that you're Happy, is this correct?	Yes	Yes	Wow congratulations on your engagement! Shall we celebrate with some good-feeling tunes?	Wow congratulations on your engagement! Shall we celebrate with some good-feeling tunes?

HappyChecker & HappyChecker-no (with 'No' reply to confirm)							
Intent: **-HappyChecker**				Intent: **-Happy**			
User Speech	100% Recorded?	Expected Result	Actual Result	User Speech	100% Recorded?	Expected Result	Actual Result
I feel happy	Yes	I'm detecting that you're happy, is this correct?	I'm detecting that you're happy, is this correct?	No	Yes	I'm sorry, I'm still learning. Which of the following four emotions would you like me to play a song from? Angry, Confident, Happy or sad?	I'm sorry, I'm still learning. Which of the following four emotions would you like me to play a song from? Angry, Confident, Happy or sad?

-Happy-Yes & -Happy-No			
		Intent:	**-Happy-yes**
User Speech	100% Recorded?	Expected Result	Actual Result
Yes	Yes	I recommend $song by $artist. Would you like to hear the audio analysis?	I recommend $song by $artist. Would you like to hear the audio analysis?
		Intent:	**-Happy-yes**
No	Yes	Ok, let me know if you need me. END CONVERSATION	Ok, let me know if you need me. END CONVERSATION
-Happy-Yes-No			
		Intent:	**-Happy-yes-no**

User Speech	100% Recorded?	Expected Result	Actual Result
No	Yes	Ok, shall I still play it?	Ok, shall I still play it?
		Intent:	**-Happy-yes-no-chooseAgain**
Choose another	Yes	Ok, I'll find another song. Just a moment. *REVERT BACK TO -Happy INTENT*	Ok, I'll find another song. Just a moment. *REVERT BACK TO -Happy INTENT*
		Intent:	**-Happy-yes-no-Yes**
Yes	Yes	Enjoy! *PLAY MUSIC, END CONVERSATION*	Enjoy! *PLAY MUSIC, END CONVERSATION*
-Happy-yes-yes & -Happy-yes-yes-yes			
		Intent:	**-Happy-yes-yes**
User Speech	100% Recorded?	Expected Result	Actual Result
Yes	Yes	The tempo of the song is… It is in the key… this key evokes… emotions. Would you still like me to play it?	The tempo of the song is… It is in the key… this key evokes… emotions. Would you still like me to play it?
		Intent:	**-Happy-yes-yes-yes**
Yes	Yes	Enjoy! *PLAY MUSIC, END CONVERSATION*	Enjoy! *PLAY MUSIC, END CONVERSATION*
-Happy-Yes-Yes-No & -Happy-Yes-Yes-No-Yes & -Happy-Yes-Yes-Yes-No-No			
		Intent:	**-Happy-yes-yes-no**
User Speech	100% Recorded?	Expected Result	Actual Result
No	Yes	Would you like another song of the same emotion instead?	Would you like another song of the same emotion instead?
		Intent:	**-Happy-yes-yes-no-yes**
Yes	Yes	Ok, I'll find another song. Just a moment. *REVERT BACK TO -Happy INTENT*	Ok, I'll find another song. Just a moment. *REVERT BACK TO -Happy INTENT*
		Intent:	**-Happy-yes-yes-no-no**
No	Yes	Ok, let me know if you need me. END CONVERSATION	Ok, let me know if you need me. END CONVERSATION

iv) Confidence Unit Tests

ConfidentChecker & -Confident (with 'Yes' reply to confirm)	
Intent: **-ConfidentChecker**	Intent: **-Confident**

User Speech	100% Recorded?	Expected Result	Actual Result	User Speech	100% Recorded?	Expected Result	Actual Result
I feel confident	Yes	I'm detecting that you want a Confident song, is this correct?	I'm detecting that you want a Confident song, is this correct?	Yes	Yes	How about we listen to a Confident song together then?	How about we listen to a Confident song together then?
I'm in need of some confidence	Yes	I'm detecting that you're feeling Confident, is this correct?	I'm detecting that you want a Confident song, is this correct?	Yes	Yes	How about we listen to a Confident song together then?	How about we listen to a Confident song together then?
I am anxious about tomorrow	Yes	I'm detecting that you're feeling Confident, is this correct?	I'm detecting that you want a Confident song, is this correct?	Yes	Yes	I'm sorry to hear that you feel anxious. I'm here to make you feel better prepared. Shall we listen to a confident song to boost your confidence?	I'm sorry to hear that you feel anxious. I'm here to make you feel better prepared. Shall we listen to a confident song to boost your confidence?
I feel unprepared for my exam tomorrow	Yes	I'm detecting that you're feeling Confident, is this correct?	I'm detecting that you want a Confident song, is this correct?	Yes	Yes	Don't worry you've got this, you will be absolutely fantastic for your exam tomorrow, shall we listen to a song to get you pumped up and ready to go?	Don't worry you've got this, you will be absolutely fantastic for your exam tomorrow, shall we listen to a song to get you pumped up and ready to go?

I am about to go to the gym for the first time in a year!	Yes	I'm detecting that you're feeling Confident, is this correct?	I'm detecting that you want a Confident song, is this correct?	Yes	Yes	Don't worry you've got this, you'll be absolutely fine in the gym, let's get you ready with a confidence boosting song shall we?	Don't worry you've got this, you'll be absolutely fine in the gym, let's get you ready with a confidence boosting song shall we?
I have a date tonight with Jane	Yes	I'm detecting that you're feeling Confident, is this correct?	I'm detecting that you want a Confident song, is this correct?	Yes	Yes	Oh my, so you're having a date with Jane? Let's get you in the mood with some confident songs shall we?	Oh my, so you're having a date with Jane? Let's get you in the mood with some confident songs shall we?
I have a meeting in the morning	Yes	I'm detecting that you're feeling Confident, is this correct?	I'm detecting that you want a Confident song, is this correct?	Yes	Yes	Wow, good luck for your meeting shall we listen to a confident song to get you pumped up and ready to go?	Wow, good luck for your #checker.ConfidenceEvent, shall we listen to a confident song to get you pumped up and ready to go?
I feel uneasy	Yes	I'm detecting that you're feeling Confident, is this correct?	I'm detecting that you want a Confident song, is this correct?	Yes	Yes	How about we listen to a Confident song together then?	How about we listen to a Confident song together then?
I don't feel ready for tomorrow	Yes	I'm detecting that you're feeling Confident, is this correct?	I'm detecting that you want a Confident song, is this correct?	Yes	Yes	How about we listen to a Confident song together then?	How about we listen to a Confident song together then?

I feel under pressure as there are a lot of people counting on me.	Yes	I'm detecting that you're feeling Confident, is this correct?	I'm detecting that you want a Confident song, is this correct?	Yes	Yes	How about we listen to a Confident song together then?	How about we listen to a Confident song together then?

ConfidentChecker & ConfidentChecker-no (with 'No' reply to confirm)							
Intent: **-ConfidentChecker**				Intent: **-Confident**			
User Speech	100% Recorded?	Expected Result	Actual Result	User Speech	100% Recorded?	Expected Result	Actual Result
I feel confident	Yes	I'm detecting that you're confident, is this correct?	I'm detecting that you're confident, is this correct?	No	Yes	I'm sorry, I'm still learning. Which of the following four emotions would you like me to play a song from? Angry, Confident, Happy or sad?	I'm sorry, I'm still learning. Which of the following four emotions would you like me to play a song from? Angry, Confident, Happy or sad?

-Confident-Yes & -Confident-No			
			Intent: **-Confident-yes**
User Speech	100% Recorded?	Expected Result	Actual Result
Yes	Yes	I recommend $song by $artist. Would you like to hear the audio analysis?	I recommend $song by $artist. Would you like to hear the audio analysis?
			Intent: **-Confident-yes**
No	Yes	Ok, let me know if you need me. END CONVERSATION	Ok, let me know if you need me. END CONVERSATION
-Confident-Yes-No			
			Intent: **-Confident-yes-no**
User Speech	100% Recorded?	Expected Result	Actual Result
No	Yes	Ok, shall I still play it?	Ok, shall I still play it?
			Intent: **-Confident-yes-no-chooseAgain**

Choose another	Yes	Ok, I'll find another song. Just a moment. *REVERT BACK TO -Confident INTENT*	Ok, I'll find another song. Just a moment. *REVERT BACK TO -Confident INTENT*
			Intent: **-Confident-yes-no-Yes**
Yes	Yes	Enjoy! *PLAY MUSIC, END CONVERSATION*	Enjoy! *PLAY MUSIC, END CONVERSATION*
-Confident-yes-yes & -Confident-yes-yes-yes			
			Intent: **-Confident-yes-yes**
User Speech	100% Recorded?	Expected Result	Actual Result
Yes	Yes	The tempo of the song is… It is in the key… this key evokes… emotions. Would you still like me to play it?	The tempo of the song is… It is in the key… this key evokes… emotions. Would you still like me to play it?
			Intent: **-Confident-yes-yes-yes**
Yes	Yes	Enjoy! *PLAY MUSIC, END CONVERSATION*	Enjoy! *PLAY MUSIC, END CONVERSATION*
-Confident-Yes-Yes-No & -Confident-Yes-Yes-No-Yes & -Confident-Yes-Yes-Yes-No-No			
			Intent: **-Confident-yes-yes-no**
User Speech	100% Recorded?	Expected Result	Actual Result
No	Yes	Would you like another song of the same emotion instead?	Would you like another song of the same emotion instead?
			Intent: **-Confident-yes-yes-no-yes**
Yes	Yes	Ok, I'll find another song. Just a moment. *REVERT BACK TO -Confident INTENT*	Ok, I'll find another song. Just a moment. *REVERT BACK TO -Confident INTENT*
			Intent: **-Confident-yes-yes-no-no**
No	Yes	Ok, let me know if you need me. END CONVERSATION	Ok, let me know if you need me. END CONVERSATION

v) 'playMusic' Function Testing Results for each emotion

	Angry	Sad	Happy	Confident
Success Rate	100%	100%	100%	100%
Fail Rate	0%	0%	0%	0%

Appendix D

i) Comparing the Console Log scores of 'Dance' by 'Abba' with the Output SQL Table

----------"Dance"----------

```
{
 "document_tone": {
  "tones": [
   {
    "score": 0.570108,
    "tone_id": "joy",
    "tone_name": "Joy"
   },
   {
    "score": 0.607081,
    "tone_id": "sadness",
    "tone_name": "Sadness"
   }
  ]
 },
 "sentences_tone": [
  {
   "sentence_id": 0,
   "text": "Oh  my love it makes me sad.",
   "tones": [
    {
     "score": 0.832847,
     "tone_id": "sadness",
     "tone_name": "Sadness"
    }
   ]
  },
  {
   "sentence_id": 1,
   "text": "Why did things turn out so bad?",
   "tones": [
    {
     "score": 0.582756,
     "tone_id": "sadness",
     "tone_name": "Sadness"
    },
    {
     "score": 0.898327,
     "tone_id": "confident",
     "tone_name": "Confident"
    },
    {
     "score": 0.620279,
     "tone_id": "analytical",
     "tone_name": "Analytical"
    }
```

```
    ]
   },
   {
    "sentence_id": 2,
    "text": "Was it just a dream  everything we did  everything we had?  .",
    "tones": [
     {
      "score": 0.560996,
      "tone_id": "joy",
      "tone_name": "Joy"
     },
     {
      "score": 0.786991,
      "tone_id": "tentative",
      "tone_name": "Tentative"
     }
    ]
   },
   {
    "sentence_id": 3,
    "text": "Baby  give me one more Dance while the music still goes on Don't think about tomorrow Dance and forget our time is gone Tonight's a night we borrow.",
    "tones": [
     {
      "score": 0.679537,
      "tone_id": "joy",
      "tone_name": "Joy"
     },
     {
      "score": 0.659886,
      "tone_id": "analytical",
      "tone_name": "Analytical"
     }
    ]
   },
   {
    "sentence_id": 4,
    "text": "Let's make it a memory  a night of our own A thing to remember when we're all alone So dance  it's our way to say goodbye  .",
    "tones": [
     {
      "score": 0.667445,
      "tone_id": "sadness",
      "tone_name": "Sadness"
     },
     {
      "score": 0.509368,
      "tone_id": "confident",
      "tone_name": "Confident"
     }
```

```
    ]
   },
   {
    "sentence_id": 5,
    "text": "Yes  all we have to do is Dance while the music still goes on This is no time for crying Dance  don't you hear them play our song God knows that we've been trying But we didn't make it 'cause nothing's the same We just couldn't help it  nobody's to blame So dance while the music still goes on And let it be our last goodbye  .",
    "tones": [
     {
      "score": 0.572319,
      "tone_id": "sadness",
      "tone_name": "Sadness"
     }
    ]
   },
   {
    "sentence_id": 6,
    "text": "Yet it seems to make me sad.",
    "tones": [
     {
      "score": 0.862792,
      "tone_id": "sadness",
      "tone_name": "Sadness"
     },
     {
      "score": 0.91961,
      "tone_id": "tentative",
      "tone_name": "Tentative"
     }
    ]
   },
   {
    "sentence_id": 7,
    "text": "Why did things turn out so bad?",
    "tones": [
     {
      "score": 0.582756,
      "tone_id": "sadness",
      "tone_name": "Sadness"
     },
     {
      "score": 0.898327,
      "tone_id": "confident",
      "tone_name": "Confident"
     },
     {
      "score": 0.620279,
      "tone_id": "analytical",
      "tone_name": "Analytical"
```

```
      }
     ]
    },
    {
     "sentence_id": 8,
     "text": "Was is just a dream  everything we did  everything we had?  .",
     "tones": [
      {
       "score": 0.560996,
       "tone_id": "joy",
       "tone_name": "Joy"
      },
      {
       "score": 0.786991,
       "tone_id": "tentative",
       "tone_name": "Tentative"
      }
     ]
    },
    {
     "sentence_id": 9,
     "text": "Baby  give me one last Dance while the music still goes on Just like the night I met you Dance and believe me  when you're gone You know I won't forget you Our love was a snowbird  it's flying away You tell me it's over  what more can I say?",
     "tones": [
      {
       "score": 0.760704,
       "tone_id": "joy",
       "tone_name": "Joy"
      }
     ]
    },
    {
     "sentence_id": 10,
     "text": "So dance while the music still goes on It's gonna be our last goodbye  .",
     "tones": [
      {
       "score": 0.7313,
       "tone_id": "sadness",
       "tone_name": "Sadness"
      }
     ]
    },
    {
     "sentence_id": 11,
     "text": "Dance while the music still goes on Don't think about tomorrow Dance and forget our time is gone Tonight's a night we borrow.",
     "tones": [
      {
       "score": 0.643482,
```

```
        "tone_id": "joy",
        "tone_name": "Joy"
       },
       {
        "score": 0.731735,
        "tone_id": "analytical",
        "tone_name": "Analytical"
       }
     ]
    },
    {
     "sentence_id": 12,
     "text": "Let's make it a memory  a night of our own A thing to remember when we're all alone So dance while the music still goes on And let it be our last goodbye  .",
     "tones": [
       {
        "score": 0.65629,
        "tone_id": "sadness",
        "tone_name": "Sadness"
       }
     ]
    },
    {
     "sentence_id": 13,
     "text": "Dance while the music still goes on This is no time for crying Dance  don't you hear them play our song God knows that we've been trying Dance while the music still goes on Just like the night I met you Dance and believe me  when you're gone You know I won't forget you  .",
     "tones": [
       {
        "score": 0.732132,
        "tone_id": "joy",
        "tone_name": "Joy"
       },
       {
        "score": 0.5538,
        "tone_id": "tentative",
        "tone_name": "Tentative"
       }
     ]
    }
  ]
}
```


this is the number of document scores: 2

0.607081

There are 1 scores for this sentence.
These are the individual scores for each sentence: 0.832847
There are 3 scores for this sentence.
These are the individual scores for each sentence: 0.582756

These are the individual scores for each sentence: 0.898327
These are the individual scores for each sentence: 0.620279
There are 2 scores for this sentence.
These are the individual scores for each sentence: 0.560996
These are the individual scores for each sentence: 0.786991
There are 2 scores for this sentence.
These are the individual scores for each sentence: 0.679537
These are the individual scores for each sentence: 0.659886
There are 2 scores for this sentence.
These are the individual scores for each sentence: 0.667445
These are the individual scores for each sentence: 0.509368
There are 1 scores for this sentence.
These are the individual scores for each sentence: 0.572319
There are 2 scores for this sentence.
These are the individual scores for each sentence: 0.862792
These are the individual scores for each sentence: 0.91961
There are 3 scores for this sentence.
These are the individual scores for each sentence: 0.582756
These are the individual scores for each sentence: 0.898327
These are the individual scores for each sentence: 0.620279
There are 2 scores for this sentence.
These are the individual scores for each sentence: 0.560996
These are the individual scores for each sentence: 0.786991
There are 1 scores for this sentence.
These are the individual scores for each sentence: 0.760704
There are 1 scores for this sentence.
These are the individual scores for each sentence: 0.7313
There are 2 scores for this sentence.
These are the individual scores for each sentence: 0.643482
These are the individual scores for each sentence: 0.731735
There are 1 scores for this sentence.
These are the individual scores for each sentence: 0.65629
There are 2 scores for this sentence.
These are the individual scores for each sentence: 0.732132
These are the individual scores for each sentence: 0.5538
This is the max score and corresponding text for the current emotion:
0.862792
Yet it seems to make me sad.
"sadness"
1 record (0.607081 for "sadness")inserted for "Dance" by " ABBA"

ii) Output SQL Table showing Abba's 'Dance' Sentiment Tone Analysis Scores

artist	song	maxEmotion	score	maxSentence	maxSentenceScore
ABBA	Dance	sadness	0.607081	Yet it seems to make me sad.	0.862792

ii) Output SQL Table Showing the Same Values After 15 Repetitions for Elis Presley's 'All Shook Up'

artist	song	maxEmotion	score	maxSentence	maxSentenceScore
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096
Elvis Presley	All Shook Up	joy	0.651745	Well she touched my hand what a chill I got Her lips are like a volcano when it's hot I'm proud to say that she's my buttercup I'm in love I'm all shook up Mm mm mm mm yay yay yay .	0.955096

Appendix E

i) Copy of the Email Containing the questionnaire and the instructions on how to Complete It

Dear Respondent,

Thank you for agreeing to take part in this research. Your support for this project is highly valued. Please take some time to read the project's GDPR fulfilled Privacy Policy document before using the application: https://docs.google.com/document/d/1_mBl9TvejvFY0OaBB-eJwg8J63P0pkjpZtTdN5ybvkA/edit#

First, to use the application you must link your Google and Premium Spotify accounts by following this link: spotifyauth-cad4c.firebaseapp.com. Since the app is currently in sandbox mode, you will need to login using the Spotify login details provided (see attachment). You will also need to log into the Spotify Web Player or Application using the same credentials. Please ensure Spotify, either the web app version or the desktop application, is open and active in order for the system to play music.

Once the accounts are connected, you will be able to use the application on any device with Google Assistant installed on it. For the purposes of this project, I kindly ask that you use a device with microphone as the input.

There are two methods to launch the application:

1. state clearly "Ok Google, Talk To Mood Magic Therapy".
2. Or, follow this link to the actions on the Google Assistant, https://assistant.google.com/explore, and type in Mood Magic Therapy in the search box. Then navigate to the application by clicking on the Mood Magic Therapy app. This will bring you to the application's page: https://assistant.google.com/services/a/uid/00000096e92c62db . Here, you can click 'send to device' where you can choose any available device to send the app to. The application will now launch and you will be guided through the conversation. Please read the remainder of this email to find out how to conduct the tests.

Below is a table containing all the 'Use Cases' for you to complete; These are simply different scenarios in which users may want to use the app. Each Use Case has a Use Case Identifier (UCID) and a brief description of the path to be taken. You must complete each Use Case for each of the four intents. It is up to your discretion what exact words you speak, as long as they are different ways you might say the required emotion.

UCID	Use Case

1	User plays first song recommended to them with audio analysis
2	User plays first song recommended to them with no audio analysis
3	User chooses another song after hearing the audio analysis, hears the next recommendation's audio analysis but decides not to play that one either
4	User listens to the audio analysis of the recommendation and says 'no' to both playing it and choosing another song
5	Users says 'no' to the song being played after not wanting the first recommended song (without wanting audio analysis of the second song)
6	User asks for another song after hearing audio analysis of the first recommendation, then asks for another song before hearing audio analysis of the second song, then finally says 'yes' to playing that song
7	User asks for another song after hearing audio analysis of the first recommendation, then asks for another song before hearing audio analysis of the second song, then finally says 'no' to playing that song
8	User asks for another song after hearing audio analysis of the first recommendation, then wants the next song to be played after hearing audio analysis of it
9	User asks for another song after hearing audio analysis of the first recommendation, then chooses another song after hearing its audio analysis and then finally wants the next song played without audio analysis
10	User continually requests another song before hearing audio analysis - also tests randomiser
11	User continually requests another song after hearing audio analysis - also tests randomiser
12	User wants an emotion which is not the one they are first offered, they swap to this emotion and continue through to the end in both cases

13	User wants an emotion which is not the one they are first offered, they swap to another emotion and continue through to the end in both cases

When completing each use case, please complete the questionnaire (see below) with your ratings.

	Original Starting Intent															
UC ID	Angry				Sad				Happy				Confident			
	Did the agent respond correctly throughout the conversation?	How Appropriate were the responses? (0-10)	How close were the responses to being ‘human’? (0-10)	What percentage did the first recommended song fit the desired emotion? (0-100%)	Did the agent respond correctly throughout the conversation?	How Appropriate were the responses? (0-10)	How close were the responses to being ‘human’? (0-10)	What percentage did the first recommended song fit the desired emotion? (0-100%)	Did the agent respond correctly throughout the conversation?	How Appropriate were the responses? (0-10)	How close were the responses to being ‘human’? (0-10)	What percentage did the first recommended song fit the desired emotion? (0-100%)	Did the agent respond correctly throughout the conversation?	How Appropriate were the responses? (0-10)	How close were the responses to being ‘human’? (0-10)	What percentage did the first recommended song fit the desired emotion? (0-100%)
1																
2																
3																
4																
5																
6																
7																
8																
9																
10																
11																
12																
13																

For the meanings behind each criterion, please read the following:

Did the agent respond correctly throughout the conversation?
Based on your utterances, did the Google agent reply with the correct response? I.e. did the conversation flow naturally or were there any unexpected responses which did not fit the context of the conversation?

How Appropriate were the responses? (0-10)
Based on your utterances, did the Google agent reply with a degree of appropriateness? For instance, if you told the agent that you were feeling sad, did the agent respond in a suitable tone, or could her response have been more suitable given the circumstances? This should be rated from 0, which is a not at all appropriate and perhaps a rude response, to 10 which is a perfectly suitable response which would not seem out of place during an actual therapy session with a professional therapist.

How close were the responses to being 'human'? (0-10)
Based on your utterances, did the Google agent's reply sound like a reply which your family or friends would give you? Or was it too obviously a pre-programmed robot's reply? This should be rated from 0, where the response was entirely robotic and unnatural, to 10, where the response was succinctly spoken with a tone as real as that of a friend.

What percentage did the first recommended song fit the desired emotion? (0-100%)
Based on the emotion of song which you asked for, upon listening to the song, how well did it fit that emotion? This should be rated as a percentage out of 100, with 0% being a completely inappropriate song given the desired emotion, and 100 being the perfect song to fit the desired emotion.

Once complete, please email back your completed questionnaires to me at: marc.jwatts@gmail.com. As a final task, please could you include at the bottom of the email, the name and artist of the song which you felt least matched the desired emotion. Any other comments or suggestions are greatly appreciated.

Thank you very much for your time in supporting this project!

Kind regards,

Marcus

ii) Averages of the Results from all the responses

	Original Starting Intent															
UC ID	Angry				Sad				Happy				Confident			
	Did the agent respond correctly throughout the conversation?	How Appropriate were the responses? (0-10)	How close were the responses to being 'human'? (0-10)	What percentage did the first recommended song fit the desired emotion? (0-100%)	Did the agent respond correctly throughout the conversation?	How Appropriate were the responses? (0-10)	How close were the responses to being 'human'? (0-10)	What percentage did the first recommended song fit the desired emotion? (0-100%)	Did the agent respond correctly throughout the conversation?	How Appropriate were the responses? (0-10)	How close were the responses to being 'human'? (0-10)	What percentage did the first recommended song fit the desired emotion? (0-100%)	Did the agent respond correctly throughout the conversation?	How Appropriate were the responses? (0-10)	How close were the responses to being 'human'? (0-10)	What percentage did the first recommended song fit the desired emotion? (0-100%)
1	Yes	8.78	7.76	60.77	Yes	8.45	5.76	66.63	Yes	7.43	6.09	66.63	Yes	9.78	5.76	65.78
2	Yes	7.43	7.89	66.89	Yes	8.43	6.89	66.99	Yes	7.43	7.89	66.63	Yes	8.78	7.89	64.89
3	Yes	9.45	8.89	68.76	Yes	9.05	9.89	78.76	Yes	9.45	8.89	68.76	Yes	8.78	7.23	69.76
4	Yes	6.89	8.9	93.09	Yes	6.09	7.98	73.09	Yes	7.65	8.99	93.09	Yes	8.78	8.9	60.78
5	Yes	8.01	8.56	78.98	Yes	7.01	7.56	88.98	Yes	6.01	8.56	65.78	Yes	8.78	7.01	78.98
6	Yes	8.79	8.23	66.65	Yes	8.99	7.23	86.65	Yes	7.65	8.23	69.76	Yes	8.78	8.23	62.65
7	Yes	7.6	7.67	70.78	Yes	6.6	7.88	60.78	Yes	7.6	7.67	86.65	Yes	8.78	8.89	76.78
8	Yes	9.65	6.89	88.98	Yes	7.65	8.89	78.98	Yes	9.65	6.89	76.78	Yes	8.78	6.89	82.43
9	Yes	6.44	7.23	92.02	Yes	7.44	7.83	82.43	Yes	6.44	7.53	92.02	Yes	8.78	7.23	91.01
10	Yes	6.34	8.72	88.89	Yes	7.64	8.78	60.94	Yes	6.34	8.72	82.43	Yes	8.78	8.72	50.96
11	Yes	7.56	9.34	67.44	Yes	8.56	8.36	60.78	Yes	7.9	7.47	60.94	Yes	8.78	8.78	67.44
12	Yes	6.9	8.88	89.33	Yes	9	7.9	79.53	Yes	6.9	8.88	89.33	Yes	8.78	8.88	58.69
13	Yes	8.8	7.45	55.99	Yes	6.68	7.47	58.69	Yes	8.8	7.45	55.99	Yes	8.78	7.45	79.53

14	Yes	7.6	6.98	78.89	Yes	5.66	6.99	88.89	Yes	7.6	6.98	60.78	Yes	8.78	7.47	78.89
15	Yes	9.91	5.78	76.84	Yes	7.91	6	77.64	Yes	9.91	5.78	88.89	Yes	8.78	5.78	88.89

iii) Total of the Results from all Responses

	Original Starting Intent															
	Angry				**Sad**				**Angry**				**Confident**			
	Did the agent respond correctly throughout the conversation?	**How Appropriate were the responses? (0-10)**	**How close were the responses to being 'human'? (0-10)**	**What percentage did the first recommended song fit the desired emotion? (0-100%)**	**Did the agent respond correctly throughout the conversation?**	**How Appropriate were the responses? (0-10)**	**How close were the responses to being 'human'? (0-10)**	**What percentage did the first recommended song fit the desired emotion? (0-100%)**	**Did the agent respond correctly throughout the conversation?**	**How Appropriate were the responses? (0-10)**	**How close were the responses to being 'human'? (0-10)**	**What percentage did the first recommended song fit the desired emotion? (0-100%)**	**Did the agent respond correctly throughout the conversation?**	**How Appropriate were the responses? (0-10)**	**How close were the responses to being 'human'? (0-10)**	**What percentage did the first recommended song fit the desired emotion? (0-100%)**
TOTAL	100	7.90	8.19	76.04	100	7.81	7.88	72.56	100	7.63	7.94	74.98	100	8.86	7.84	69.98

Appendix F

i) The lyrics of Celine Dion's Cover of 'All By Myself'

When I was young
I never needed anyone
And making love was just for fun
Those days are gone
Livin' alone
I think of all the friends I've known
But when I dial the telephone
Nobody's home
All by myself
Don't wanna be
All by myself
Anymore
Hard to be sure
Sometimes I feel so insecure
And love so distant and obscure
Remains the cure
All by myself
Don't wanna be
All by myself
Anymore
All by myself
Don't wanna live
All by myself
Anymore
When I was young
I never needed anyone
And making love was just for fun
Those days are gone

All by myself
Don't wanna be
All by myself
Anymore
All by myself
Don't wanna live
Oh
Don't wanna live
By myself, by myself
Anymore
By myself
Anymore
Oh
All by myself
Don't wanna live
I never, never, never
Needed anyone

ii) SentiWordNet Analysis of Celine Dion's cover of 'All By Myself'

{ sentiment: 3,
  avgSentiment: 0.1,
  objective: 24.5,
  positive: 4.25,
  negative: 1.25,
  ngrams:
   ['young',
     'love',
     'fun',
     'telephone',
     'home',
     't',
     'be',
     'be',
     'sure',
     'feel',
     'insecure',
     'love',
     'distant',
     'obscure',
     'cure',
     't',
     'be',
     't',
     'live',
     'young',
     'love',
     'fun',
     't',

```
   'be',
   't',
   'live',
   't',
   'live',
   't',
   'live' ],
 words:
  [ { '# POS': 'a',
      ID: '1649720',
      PosScore: '0.625',
      NegScore: '0.25',
      SynsetTerms: 'young' },
    { '# POS': 'n',
      ID: '7488340',
      PosScore: '0.25',
      NegScore: '0',
      SynsetTerms: 'love' },
    { '# POS': 'n',
      ID: '429949',
      PosScore: '0.375',
      NegScore: '0',
      SynsetTerms: 'fun' },
    { '# POS': 'n',
      ID: '6272290',
      PosScore: '0',
      NegScore: '0',
      SynsetTerms: 'telephone' },
    { '# POS': 'n',
      ID: '8490199',
      PosScore: '0',
      NegScore: '0',
      SynsetTerms: 'home' },
    { '# POS': 'n',
      ID: '5413647',
      PosScore: '0',
      NegScore: '0',
      SynsetTerms: 't' },
    { '# POS': 'v',
      ID: '2744820',
      PosScore: '0',
      NegScore: '0',
      SynsetTerms: 'be' },
    { '# POS': 'v',
      ID: '2744820',
      PosScore: '0',
      NegScore: '0',
      SynsetTerms: 'be' },
    { '# POS': 'a',
      ID: '724397',
```

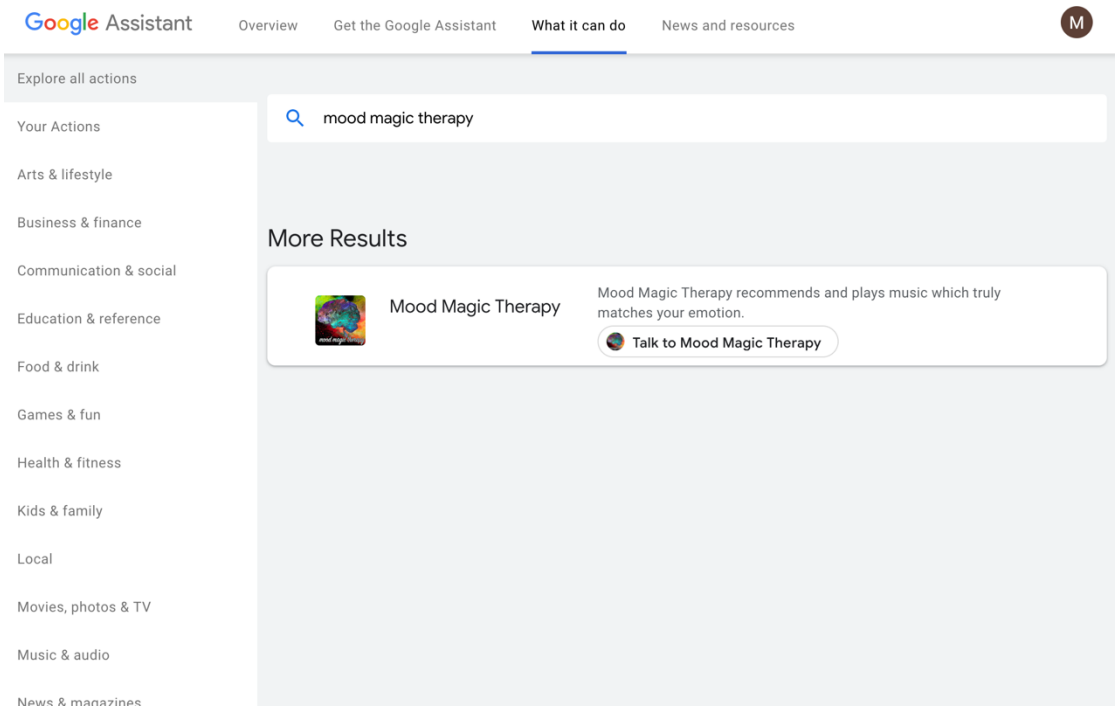
```
  PosScore: '0.625',
  NegScore: '0',
  SynsetTerms: 'sure' },
{ '# POS': 'v',
  ID: '715239',
  PosScore: '0.125',
  NegScore: '0',
  SynsetTerms: 'feel' },
{ '# POS': 'a',
  ID: '2094755',
  PosScore: '0.25',
  NegScore: '0.375',
  SynsetTerms: 'insecure' },
{ '# POS': 'n',
  ID: '7488340',
  PosScore: '0.25',
  NegScore: '0',
  SynsetTerms: 'love' },
{ '# POS': 'a',
  ID: '1987646',
  PosScore: '0.25',
  NegScore: '0',
  SynsetTerms: 'distant' },
{ '# POS': 'a',
  ID: '431004',
  PosScore: '0.25',
  NegScore: '0.375',
  SynsetTerms: 'obscure' },
{ '# POS': 'n',
  ID: '4074482',
  PosScore: '0',
  NegScore: '0',
  SynsetTerms: 'cure' },
{ '# POS': 'n',
  ID: '5413647',
  PosScore: '0',
  NegScore: '0',
  SynsetTerms: 't' },
{ '# POS': 'v',
  ID: '2744820',
  PosScore: '0',
  NegScore: '0',
  SynsetTerms: 'be' },
{ '# POS': 'n',
  ID: '5413647',
  PosScore: '0',
  NegScore: '0',
  SynsetTerms: 't' },
{ '# POS': 'v',
  ID: '2614181',
```

```
  PosScore: '0',
  NegScore: '0',
  SynsetTerms: 'live' },
{ '# POS': 'a',
  ID: '1649720',
  PosScore: '0.625',
  NegScore: '0.25',
  SynsetTerms: 'young' },
{ '# POS': 'n',
  ID: '7488340',
  PosScore: '0.25',
  NegScore: '0',
  SynsetTerms: 'love' },
{ '# POS': 'n',
  ID: '429949',
  PosScore: '0.375',
  NegScore: '0',
  SynsetTerms: 'fun' },
{ '# POS': 'n',
  ID: '5413647',
  PosScore: '0',
  NegScore: '0',
  SynsetTerms: 't' },
{ '# POS': 'v',
  ID: '2744820',
  PosScore: '0',
  NegScore: '0',
  SynsetTerms: 'be' },
{ '# POS': 'n',
  ID: '5413647',
  PosScore: '0',
  NegScore: '0',
  SynsetTerms: 't' },
{ '# POS': 'v',
  ID: '2614181',
  PosScore: '0',
  NegScore: '0',
  SynsetTerms: 'live' },
{ '# POS': 'n',
  ID: '5413647',
  PosScore: '0',
  NegScore: '0',
  SynsetTerms: 't' },
{ '# POS': 'v',
  ID: '2614181',
  PosScore: '0',
  NegScore: '0',
  SynsetTerms: 'live' },
{ '# POS': 'n',
  ID: '5413647',
```

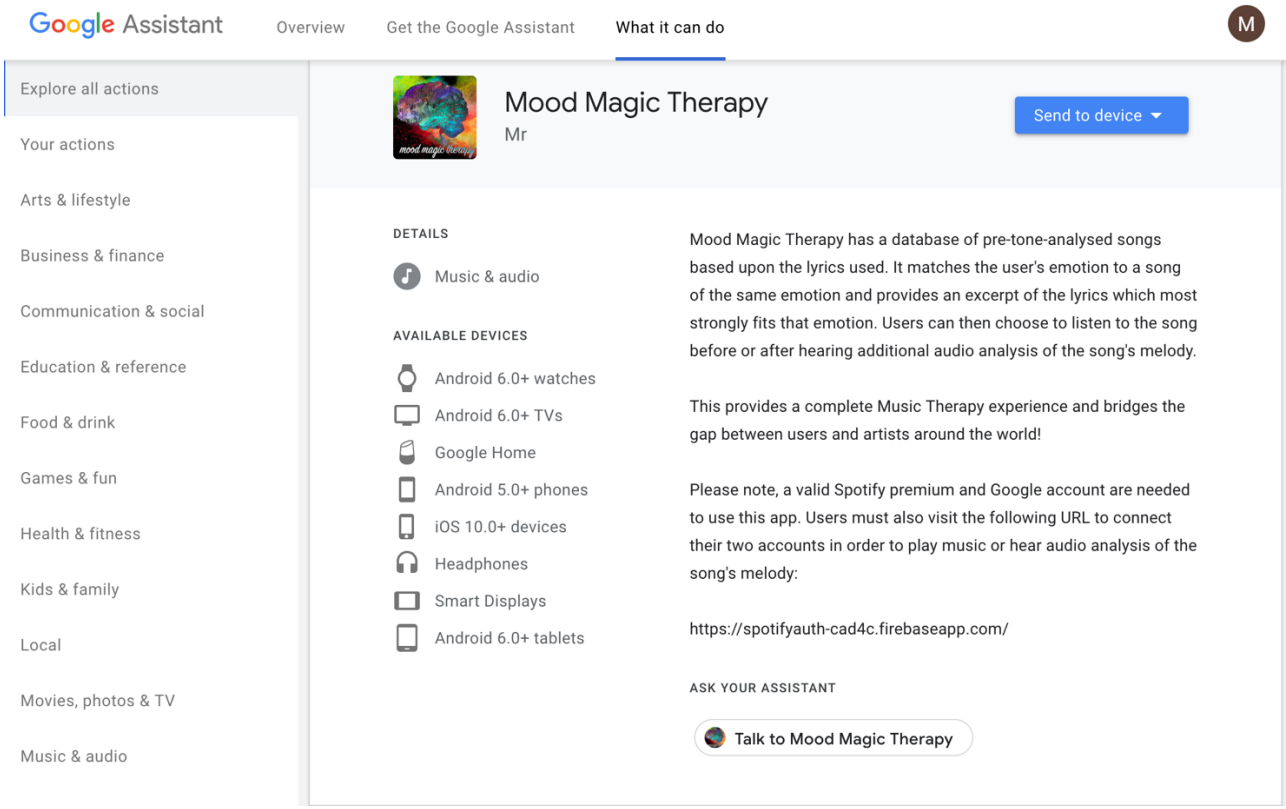
```
    PosScore: '0',
    NegScore: '0',
    SynsetTerms: 't' },
  { '# POS': 'v',
    ID: '2614181',
    PosScore: '0',
    NegScore: '0',
    SynsetTerms: 'live' } ] }
```

Appendix G

i) Searching for the application on the Google Assistant Actions-on-Google Console



ii) The application's page on Google Assistant showing the list of Supported Devices



Appendix H

Setup

This section outlines how to access Mood Magic Therapy. Despite the music playing functions requiring the developer's Spotify account (since Spotify did not allow for the Spotify app to be submitted to their website), the application can still be used via the Google Assistant. The application will let users converse with the system as normal and will recommend music based off the emotions that they are feeling. The application will provide audio analysis since this will be spoken by the Google agent, however, it will not be able to play the recommended song on a new Spotify account.

To launch the application, simply speak or type 'Talk to Mood Magic Therapy' into a Google Assistant device. Or, to find the application page, users need only search for 'Mood Magic Therapy' on the Google Assistant https://assistant.google.com/explore, as shown in Appendix G. Once found (https://assistant.google.com/services/a/uid/00000096e92c62db), users can click 'send to device'. The app will launch and the user will be guided through the conversation.

Repository

Please follow the link to this project's repository below. The folder 'SpotifyAuth' contains the full application code. Navigate through to functions and then to index.js to see the entirety of the Webhook code. A .zip file containing all the Dialogflow intents is also held within the 'SpotifyAuth' folder. The folder 'DBTestApp/DatabaseAPiApp/npm-global' contains the

code uses to create and manipulate the SQL databases and request Watson tone analysis on song lyrics.

This is the link to the full repository:

https://git-teaching.cs.bham.ac.uk/mod-msc-proj-2018/mxw817