CSED Coursework 2

Adam George Eddy Dunton Fraser Dwyer Lance Garcia Sam Davidson
Sam Freeman Sandra-Maria Comradi Teodora Dinca

Semester 2, 2019-2020

Contents

Introduction

1.1 Personal Informatics - FD

Personal informatics is a philosophy in which technology can aid the daily lives of people by collecting and processing data. This data can be monitored or manually entered in order to be collected. Personal Informatics tools are used for a variety of reasons, for example, to be therapeutic, or to change or improve one's behaviour or psychology. Essentially, Personal Informatics tools accumulate someone's data and feeds it back to the user through some form of data representation. For example, self-monitoring calorie consumption can help to keep track of diets, using graphs and other visual aids to demonstrate trends in the data. Personal Informatics can aid with "...internal states (such as mood or glucose level in the blood) or indicators of performance (such as the kilometres run)." [Rapp and Cena(2014)] . Evidently, PI covers a vast amount of areas in our day to day lives, hence why PI tracking tools are so helpful.

With ever growing technological advancements, being able to use more and more PI trackers is becoming much easier. With technologies such as smart watches, sensors in our phones, Fitbits, etc... we are constantly surrounded by computers capable of PI tracking. Furthermore, "the pervasiveness of self-tracking in modern smartphones foreshadows an era where Personal Informatics will likely become ubiquitous making personal data available with minimal burden, easing the process of self-monitoring." This means that it'll become effortless for people to have PI capable tracking technologies and hence more people will be able to take advantage of such apps and be able to benefit from them.[Rapp and Cena(2016)]

However, PI tracking tools aren't without their flaws. Many PI tracking tools lack helpful suggestions and consequently don't always give users the helpful insight they were after. A common issue is the "excess of abstract visualisation in the apps" [Rapp and Cena(2016)] which consequently can lead to users losing interest in PI tracking apps. Moreover, "we believe that current PI tools are not yet designed with enough understanding of these users' needs, desires and problems that they may encounter." [Rapp and Cena(2016)] This implies that despite the overload of information we can be provide for these apps, the information given in return isn't as useful as it should be.

1.2 Proposed Solution - SD

Since the problem domain is personal informatics, we have decided to research how music can affect a listeners' mood and perspective based mainly on the type of music they are listening to, whether that is upbeat music or it's a more calming sound, etc. We will also be looking into how different genres will impact on users, and we will take into consideration the time of day as well. Our application will analyse this data and return the mood that a user would generally feel while listening to it. We will be collecting this data from music streaming app Spotify. We would like to have the framework to include different music streaming services such as Apple Music, Deezer, etc.

We know of features which are similar to what we aim to create. For example, Spotify Wrapped, which started as a simple microsite in 2015, showing users how they engaged with the service [Swant(2019)]. In 2018, they introduced a personalised feature based off the original in 2015. This personalised feature shows their most listened-to artists, albums, songs, playlists and features from across the year for all users [Somerville(2019)]. This has required Spotify to start taking users listening data in order for them to create their annual list for each user. This helps us as the Spotify API which we are going to end up using has a lot of features which we can implement into our own work [*Spotify WebAPI Documentation*(2020)Web]. We can use this API to look at the danceability, energy, liveliness, loudness, etc. These will be taken into account when analysing the data.

For this project we will primarily be looking into the Spotify API only in order to show the main features and the capability of our application, however, we will generalise a lot of the code, this will greatly increase the extensibility of the application. This makes it so we able to add other APIs such as Apple Music and Deezer. Doing this, our application would be able to be used by more than one music streaming apps, broadening our target audience, making it accessible to a larger number of potential users. This will maximise the amount of people we can potentially help with our app.

There have been a number of articles related to studies conducted linking people's moods to upbeat or sad songs. According to a study by Psych Central, upbeat music more than likely raises the mood and perception of a listener [Nauert(2018)]. Upbeat music tends to be more happier. It has also been established that people listen to sad music are often sad and are looking for comfort, pleasure or pain [Eerola and Peltola(2016)]. All this type of information will be very useful in determining the mood and perspective of a listener. However, there are other things we must include like what makes exactly makes a song a happy one or a sad one, and whether a genre has an impact on being happy or sad.

Agile Software Process Planning and Management - FD

With the timescale we had to complete our project, we planned our sprints to be 2 weeks long each and to have 3 sprints throughout the duration of the project. We thought this would be the most beneficial to us as we would have enough time between sprints to organise ourselves in order to prepare for the next sprint, whilst still allowing us to have enough time during each sprint to get a sizable chunk of the project done.

Before the start of any sprint, we, as a group, addressed all the tasks we wanted to have completed by the end of the sprint and wrote them down on our sprint backlog. By doing so, it allowed us to easily hand out tasks and to clearly see what jobs still needed to be done. During each meeting during the sprints, we discussed if the work done had thrown up any new tasks to be added either to the product backlog or the sprint backlog depending upon its urgency and dependencies.

We tracked our sprints by using GitHub. GitHub allowed us to create an easy to use sprint progress tracker in the form of ordered lists. This includes a product backlog, sprint backlog, in progress, in review columns and more. Within these columns, we had tasks which were at varying stages of completion and allowed us to easily and visually track each sprint's progress. We managed the development of our requirements through thorough test plans and frequent checking over our list of requirements to see if we had met all the ones we had planned over the course of the relevant sprint. In our Scrum meetings, we addressed project risks and evaluated if any new risks had presented themselves. Our largest project risk developed during the project in the form of COVID-19. Due to the risk of any member of the team catching the virus, and due to the University going online mid-way through our project, we had to take all our meetings online. Whilst this made meetings less convenient and harder to communicate, we managed to make the most of the group face call feature of Microsoft Teams in order to continue our meetings as previously arranged. We managed to discuss our other risks and how we were able to reduce their impact if unfortunately, they were to occur.

Initially, we planned to have three, two-week sprints in our project. Alongside this, we planned to meet up on Mondays for a meeting and Thursdays for another. Once we had started planning our project, we realised that this wasn't enough time per week in order to complete all the work necessary. To increase our contact time, we then decided to go to the lab session allocated to our group on Thursdays as well. Once COVID-19 prevented us from meeting in person, we still continued to have the same timetable of meetings, however, all contact was done online. Whilst this seemed to make proper collaboration more difficult at first, we then took advantage of the fact that each member of the group had a computer in front of them. This meant we were able to all see more clearly what we were referring to during discussions as everyone could access the materials more easily.

Research

3.1 Prior Research - LG

This research was performed before the final solution was decided, it was performed by various members then collated by Lance Garcia.

A recurring theme throughout our research into personal informatics is the idea that the user must want to track their data, as well understand how to best make use of the system given to them. [Rapp and Cena(2014)] suggests that "common users" with little prior experience can suffer from problems with familiarity and motivation with the tracking of their personal data. They go on to suggest factors that affect self-monitoring such as "motivation for change" and "goal-setting feedback and reinforcement" should be kept in mind during the development of the program in order to maximise user interaction with the app. Through the factors given in this article, we should be able to increase the effectiveness of the user monitoring their own data, as well as collect accurate and important data.

This is supported by [Fritz et al.(2014)Fritz, Huang, Murphy and Zimmermann] who studied the long-term effects of using personal informatics tracking devices such as Nike's "FuelBand" which discovered many users experienced a short-term improvement to their health. This was due to the encouragement that users receive on such devices. This highlights the potential for long term health benefits to users that are motivated enough to continue monitoring their data over longer period. While this is not directly applicable to our proposed solution, it provides insight to the effects of personal informatics in the long and short-term. In a similar manner, being able to track what kinds of music make the user happier or focused may provide benefits to them in the long run as they listen to more of that type of music.

The idea of trust in a system is an important concept; a user's trust in the system is important as that will affect their motivation, and their likelihood of continued use of the system. In the case that the program produces a result that either the user disagrees with or dislikes, this can lead to a loss in trust, and as a result, a loss of interest in the system. To increase this sense of trust, the user should be shown how the data is collected, and to provide them with an understanding of what the data means. In addition to this, meaningful data visualisation and explanation will help to provide this understanding to the user. [Jaimes, Murray and Raij(2013)]

[Li, Dey and Forlizzi(2010)] mentions five steps for developing personal informatics software being: preparation, collection, integration, reflection, and action. It should be a conscious decision to integrate these steps into our software as it will enable the users to not only flit back and forth between these stages with ease. This will also allow us to make as much of the program software driven in order to make it more reliable and through reducing the amount that the user must do/understand will help the user to monitor their own data with minimal effort.

3.2 Security Ethics - ?

This research was performed during the production of the product, in order to assertain ethcal guidelines for users to developers to follow when securing user data.

The idea of responsible disclosure is highlighted by [Security ethics(2010)201] which describe it as: "researchers who uncover a flaw don't go public until the system's developer has had a chance to fix it". This is of note as this not only provides the developer's with time to patch their system, but also to prevent the public from losing trust in said system. Responsible disclosure is a "community accepted" protocol, so while it may not be mandatory for researchers and developers to follow through, it is in the best interest of most companies to align themselves by it.

A growing ethical concern is the phenomenon of social networks. With the group project being involved with music apps, which are commonly linked to users' social media accounts, this is of important note. The issue comes in the place of a lack of "central command" which is compounded by anonymity, as well as virtual/multiple personalities. This brings in a wide range or social and ethical issues with the use of social networking, examples of this being cyberbullying, cyberstalking, and cyber-harassment. According to a survey taken by i-SAFE, more than 53% of children admit to having been abusive over the internet, with 42% having admitted to being victims of said abuse [i SAFE(2004)]. In addition to this, the use of social networking has led to internet addiction being a clinical disorder. Users that are extroverted/unselfconscious, shy, or narcissistic [Kizza(2016)] tend to have higher usages of social networking applications.

3.3 Hashing and Password Storage - AG

This research was performed during the production of the product, in order to assertain security guidelines for the storing of passwords.

Without any sort of hashing or protection a database breach could lead attacks to obtaining users usernames and passwords in plaintext, allowing them to access users accounts on this or any other service which they used the same credientials for in a credential stuffing attack, one of the most common styles of cyber attacks [OWASP(2020)]. Hasing turns a plaintext password into a random string of characters which cannot be unhashed, instead, when a user enters their password it is hashed and the 2 hashes are comapred.

A common principle to follow in system security is Shannon's Maxim: "one ought to design systems under the assumption that the enemy will immediately gain full familiarity with them" [Shannon(1949)].This essentially means that a system shouldn't be secure through obscurity, i.e even if the attackers know the algorithms/protocols that encrypt the data, they still shouldn't be able to break the security. In theory hashing can be brute forced with current technology, however, the necessary computing power is not readily available at this moment, but, this may change in the near future. Another common approach to breaking hashes is to use a lookup table which stores the most commonly used passwords along with their hashes, which makes these passwords trivial to break. For this reason hashing alone is insufficient for password security. [Tsudik(1992)]

For this reason salts are used, which are randomly generated strings added to the password before hashing when the password is created, the salt is stored in the database alongside the password. This helps prevent against these pre-calculated hash tables because the hacker has to then recalculate these tables for each user they wish to hack into as they will have their own unique salt with them which makes these attacks more costly. Brute force attacks can never be fully stopped but using

a modern hashing alogrithm can make it more expensive for the attcker, as these focus less on speed, but instead flexiblity, for exmaple, argon2 has parameters such as memory size, iteration count, parallelism and salt length which can be adjusted to make the hash more (or less) complex making it more expensive to brute force as more resources can be required to perform a hash [Biryukov, Dinu and Khovratovich(2016)]. For these reasons we recommened the use of Argon2 over other modern algorithms such as the SHA family or Bcrypt.

3.4 Ethics of Data Storage - SD

This research was performed during the production of the product, in order to assertain guidelines for the storage and usage of user data.

Data today is something that is a key asset to companies and the day to day operations in the world. Behind it, there are many legal and ethical aspects which are important to the privacy and safety of the users who give up their data. To understand what the responsibilities are we would look into the Association of Computing Machinery's Code of Ethics and Professional Conduct. In section 1.2 it is considered a general ethical principle to "avoid harm", harm meaning any negative or unfair and significant consequences [Association for Computing Machinery (ACM)(2018)]. In terms of the data that we hold, if we pass on or give up the data to anyone else other than just for our program then we would potentially be harming the user, who have put their trust in us using their data. One example of a large company misusing information is the Facebook-Cambridge Analytica scandal in 2018. This is when Facebook allowed enough data for the political consulting company Cambridge Analytica to create psychological profiles on 70.6 million US citizens during the US presidential election in 2016 [Horwitz(2018)]. They used this collected data to create political regressive models on users for advertisement purposes, this led to predicting users personalities which were then presented with adverts which promoted certain political agendas, influencing voters through micro-targeted advertisements [Rathi(2019)]. This incident was considered illegal and unethical as they broke the British law, harvesting almost all data without explicit consent [Adam Satariano and Nicholas Confessore(2018)]. As we are dealing with personal information as well, it is very possible that the data we collect could create psychological profiles. This must be avoided and data collected will be deleted after its intended use.

So when considering the ethical concerns for storing and analysis of data, there is none when the user has given their expressed consent where they are informed fully of what the data will be used for. Additionally, with accordance to the GDPR, which looks for fairness and transparency, if a user wishes to retract their information from being used then we must return or delete the data. We must also only use the data we collect for the use we have clearly said to the user

3.5 Song attributes and mood - SC

This resesarch was performed during the production of the product, in order to assertain the relationship between characteristics of songs and their moods.

The Mood Table was created based on Thayer's mood mode, Figure 8.1, which considers the energy and the valence of a song to determine basic moods. "In most existing methods of music mood classification, the moods of songs are divided according to psychologist Robert Thayer's traditional model of mood. The model divides songs along the lines of energy and stress, from happy to sad and calm to energetic, respectively (Bhat et al 359). The eight categories created by

Thayer's model include the extremes of the two lines as well as each of the possible intersections of the lines (e.g. happy-energetic or sad-calm). [...]" [Nuzzolo(2015)]. The mood table makes use of this information by working with data provided by Spotify, the energy and the valence. The model has been refined by taking into consideration another 3 variables, acousticness, instrumentalness and danceability. These are considered attributes that create subsets of the already existing moods, making them more specific and detailed. High acousticness and instrumentalness can be used to determine a song is motivational or good for concentration, while danceability can help determine how lively a song is.

This led us to the following matrix, here all values must align with the required values for a song to be catergorised as a certain mood. Required values:

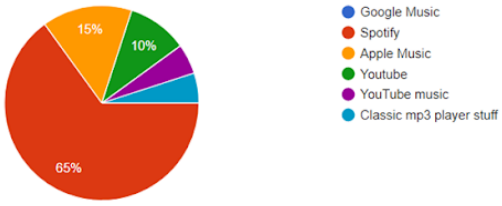
- 'n' - Neutral - Value must be between 0.4 and 0.6
- '+' - Positive - Value must be above 0.6
- '-' - Negative - Value must be below 0.4
- '/' - None dependant - Value does not matter

Mood	**Valency**	**Energy**	**Acoutstic**	**Danceability**
Neutral	n	n	/	/
Happy	+	n	/	/
Sad	-	n	/	/
Calm	n	-	/	/
Energetic	n	+	/	/
Exuberance	+	+	/	/
Lively	+	+	/	+
Joyful	+	n	/	+
Contentment	+	-	/	/
Relaxation	+	-	/	/
Frantic	-	+	/	/
Depressing	-	-	/	/
Melancholic	-	-	+	/
For Concerntraion	n	-	+	/
Motivational	n	+	+	/

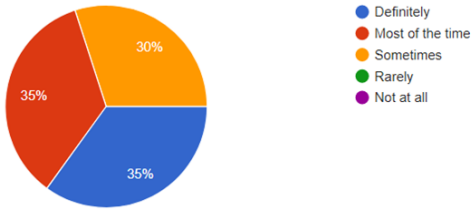
3.6 Questionnaire - SC

A questionnaire has been used to gather data about the correlation between people's moods and their music listening habits. Of our 24 repsonses we found out that around 90% of people agree with the statement "My listening habits change based on my mood" while 40% out of them strongly agree. Furthermore, 70% of people state that music plays a big role in their lives, while 60% of them think the music they listen to helps them understand themselves more. Most people stated that they listen to music at least 4 days a week, while a big part of them (65%) listen to music daily. Generally, people listen to music a couple of hours a day, while some listen to music all the time, or less than an hour a day. Music seems to help people deal with their emotions 3.1a and most people seem to use Spotify as their main music app 3.1b. We have also asked if there were any features they wished music apps included. A couple of responses addressed the way songs are recommended

to them and organised, stating that they wished the recommended songs would fit the way they are feeling at the time more, and that they would like to be able to have direct access to songs they listened to before. Other responses talked about displaying lyrics and tools to help the user recognise songs more easily. This information has strengthened the ideas our research provided and has helped us gain a better understanding of how music affects people's moods. It has prompted us to create an app that helps people track their moods based on what they listen to on Spotify.



(a) Do moods help you express or cope with different emotions and moods?



(b) What music application do you prefer?

Figure 3.1: Reponses to the questionnaire

Risk Assessment - LG & SD

Personal informatics applications already exist within a market, with many apps surrounding the tracking of music, such as Spotify Wrapped, and Last.fm. As a result, one risk we must consider is how our app is going to break into the market, with big music app owners already tracking their own data and sharing it with their customers. A potential solution to this is to have our app cover various music apps and for it to combine the statistics for all the apps. This would allow us to cover a larger audience than other apps would as many users only use one specific app to listen to music due to premium memberships in apps like Apple Music and Spotify. In addition to this, it would enable us to cater to users that use multiple apps as all their stats would be available in a single app rather than having them use various apps for several music streaming services.

Understanding the customer is vital to creating a good product, however as developers, we may not always understand the customers' point of view since we are the ones designing the program. As a countermeasure to this, we have ensured that we have done plenty of research into our audience, including the creation of a questionnaire for simple feedback, and having a focus group for more in depth feedback from a group of people that are unrelated to the development of the app. It is important to get clear and concise feedback where possible as this means that there is less to be misunderstood by the developers, thus allowing us to satisfy the customers' needs.

Furthermore, maintenance post development is important for the system to succeed in the long term. In the event that the app is not maintained after release, then there is a high likelihood of one of the music streaming services updating their services such as their API, this would lead to our app being incompatible with the streaming service, which would lead to a loss of customers. As a result, in order to keep up with rapidly evolving and updating applications, we too must keep pace, and update our system as necessary so that it will be able to function as intended as much as possible.

The threat of coronavirus is something that must be considered. As not only could it potentially put a member of our group out of action, but it can also lead to the quarantine of the university. This would lead to us being unable to meet up in person and could make working together challenging. One way we could work around this is through using online messaging apps, which we already do, but in place of in person meetings, we would need to use the support of social networking apps such as WhatsApp, Microsoft Teams, or Discord. As WhatsApp has a limit of four people in a video call, using Microsoft Teams or Discord would be ideal; it is not required that we see each other's faces when in call however, so this is of little importance.

Following on from the risk of group members being unable to work, there are an infinite number of scenarios that could result in a member being unable to participate. Whether this is because they are unconscious or are simply unable to work. Outside of generally taking care of ourselves and ensuring our health is in good shape, there is not much else that can be done about this. The Scrum methodology which we will be using throughout this project has many techniques which will help us overcome any issues which may impact our group, depending on the severity. The flexibility of the methodology allows us to go over and rethink multiple aspects of the software and

its documentation letting us get the highest quality project possible.

Risk	**Likelihood**	**Impact**	**Measures**
The time required to develop the software is underestimated.	High	Serious	Focus on the key features and the main requirements of the system, if there is spare time we can add additional features.
Complexity of the software may be too hard for us to create.	Moderate	Serious	Before we start anything we must be sure of what we plan to do and make sure it is possible for us to do with our skillset.
A key person in the project may become ill due to a number of reasons, this would delay the project as no one else would be able to do that job.	Moderate	Serious	Split up workload into groups of more than two instead of relying on just one individual.
The database cannot process as many transactions as expected.	Moderate	Serious	Gather information on a number of database systems that could be used and work out which one could work with our requirements.
The specification is not well defined.	Moderate	Minor	Any issues with the specification we will bring up with tutors and lecturers so we can get a clearer picture.
Requirements are not well defined.	Moderate	Minor	We can read through and go over different requirements to see if they are relevant and make sense to the project.
When writing documentation, people may conflict with what they are doing as responsibilities may be unclear resulting in two or more people doing the same job.	Moderate	Minor	Make sure our communication channels are efficient and before we do work we know what everyone is doing.
Everyone in the group become seriously ill from the coronavirus leaving no one able to do anything toward the project.	Moderate	Catastrophic	Follow any guidelines the government set for us to stay safe.
The specification could change for what is required in this project.	Low	Minor	Allow enough time in or after the sprints in order to make changes where necessary.
A member of the group could drop out of the university leaving us with less people to work with.	Low	Minor	Be sure that the person is communicating with us to allow us time to adjust to less people in the group.

Requirements

5.1 Gathering - FD

In order to make our system as useful as possible, we decided to collect our requirements from a wide range of varying resources. Primarily, we researched heavily into the area surrounding Personal Informatics and based a large portion of our requirements off this. By gathering a lot of requirements from articles such as those that have been peer reviewed allows us to have a better understanding about the sorts of things that our final system should and shouldn't be doing.

Moreover, we held a focus group. During this focus group, several requirements that we had missed were made apparent. By having a semi-structured conversation with potential stake holders, we were able to see roughly what a large proportion of people would want from a system like the one that we are developing. In addition to this, the focus group allowed us to quickly and efficiently gather data from people who most greatly reflect the audience we are aiming for. This meant that we had more time to work on constructing the system as opposed to researching for more requirements.

Finally, we also sent out a questionnaire. We received X RESPONSES, from which we were able to gather additional requirements. From the focus group, a mass of information was yielded which showed us a lot of perspectives we hadn't necessarily put a lot of thought into. It also provided many niche requirements which if we have time to get to would really enrich our system and take it a step above where we were aiming before.

5.2 Specific Domain - FD

"... music informatics simply involves research in areas such as the automatic transcription of music, chords, and chord progressions; key detection; and music classification." [Paas and Castro(2013)]. This simply means that music informatics is the analysis of music by means of the fundamental sounds that construct the music. In terms of Personal Informatics, this analysis can illustrate trends in qualities such as moods and behaviours. This is possible by analysing musical attributes such as "[...] melody extraction, chord recognition, beat tracking, tempo estimation, [..] and mood prediction" [Humphrey, Bello and LeCun(2013)]. Moreover, in a forever growing music industry with apps such as Apple Music, Spotify and YouTube Music, there's an increase in the interest people are taking in their own music tastes. This is becoming increasingly popular with features such as Obscurify or Spotify Wrapped "which reveals users' favourite music from the past year, [2019], has been released to [Spotify's] 243 million users" [Izzard(2019)].

Finally, due to other qualities which can be associated with the type of music that you listen to, such as "instrument identification, music similarity, genre classification, [...]" [Humphrey, Bello and LeCun(2013)], it is also possible to recommend songs similar to those that you already listen to. An increasingly smart analysis of these trends is much requested in order to get more accurate song recommen-

dations. It is evident that there is the need for more research into this growing area of personal informatics which is shown in the research that's been carried out over the last decade which has revealed "chord recognition, genre recognition, and mood estimation—are each converging to performance plateaus below satisfactory levels"[Humphrey, Bello and LeCun(2013)].

5.3 Users in this Domain - FD

There are several different types of users in our problem domain. Everyone who uses a music streaming service such as Spotify, Apple Music, and Deezer. According to a study taken in 2019, "89% [of the 34,000 participants], listen to music through on-demand streaming." [IFPI(2019)]. This evidently shows the large audience of stakeholders regarding music streaming services who would be interested in knowing more about their music interests and discovering similar songs to the ones they are already listening to. It is the majority of music listeners who would benefit from using a system like the one we are developing, novice and experienced users alike.

The more experienced music listeners stand to gain a deeper understanding of why they like the music they do, for example by seeing strong correlation between beats per minute and music danceability perhaps. A less experienced music listener may not appreciate this sort of information as much as the more avid music streamer, however, both types of users would benefit from discovering more songs that they might like through song recommendations based on mood, tempo, genre, etc... This is because many people would describe it as being difficult when it comes to finding new music that they actually like.

All types of users may feel concerns regarding the ethics to a server listening constantly to the user's phone / streaming device. There are some clear ethical implications here such as security and privacy. This may stunt the amount of potential stakeholders, but by using hashing and salting of user passwords and storing only the necessary user data on a secure server may convince people to use our music informatic tracking tool. There also may be concerns about selling personal data to third parties or it being stolen, however, in order for our system to be functional, we only need to collect minimal personal data and it would never be sold to third parties. Many users would see that the ethical concerns aren't too major and that the consequences are minimal in the rarity of anything happening.

5.4 Priorities - FD

Upon reviewing all the data we had gathered to compile our list of requirements, we labelled all of our requirements with priorities and dependencies. The dependencies really highlighted how crucial some requirements were and which of our requirements were more of the additional extras which would be nice to include if time allows. We considered the necessity of each requirement and its dependencies for the priority of any given requirement.

5.5 Conflicts - FD

When we had created our list of requirements, we were careful to avoid any conflicting requirements. These could have arisen from varying opinions from different stake holders. In order to avoid including conflicting requirements, we considered all the data that we had collected and then prioritised the one with the highest interest. Finally, we split all of our requirements into a structured indexed list to help illustrate dependencies and also to avoid repetition or conflict between requirements.

5.6 Cutbacks - ED

The COVID-19 pandemic hit roughly half just after our first sprint had ended and caused large amounts of disruption to our project. All our team members were displaced during the duration of the project causing man hours to be lost and the physical distance caused strains in communication as we adapted to the changing circumstances.

For these reasons we decided that the initial scope of our project was too large and would have to be scaled back. We decided to focus on scaling back parts which had little time already invested in them and that were less necessary to the function of the system. This was not too difficult for us as most of the first sprint focused on the backend, collecting data from Spotify and storing it in in our own database, which is an essential part of our project and could not have been scaled back anyway.

Our final product focuses on tracking what the user is listening to and the mood of those songs, the outcome of this analysis is then displayed to the user. However, in earlier versions we also planned to use this to determine trends in user moods and then recommend songs based on these trends and the other songs the user listens to. This part of the project would have been dependant on almost all of the other parts of the project, and would have required additional research into how to dictate trends how then to recommend songs based on this, which would have been quite a large undertaking and most likely would have required an entire additional sprint, for these reasons it seemed natural that this part of the project would have to be scaled back and we instead decided to display the data directly to the user. We do not believe this will end up being much less useful to our users, as with time remaining it seems unlikely that we would be able to develop algorithms which would offer more insight than what the user could draw from the data itself. This caused the focuses of sprint 3 to move towards smart methods of displaying user data rather than laying the groundwork for more advanced data analysis.

5.7 Functional

1 - Server	
1.1 - Send data to client when requested	**Author: SD**
The web app must attempt to send data to the client when requested. This data could be specified so it needs to be able to handle the specific request. The web app will then deal with the data it is given.	Priority: HIGH Dependencies: 2.1
2 - Data Collection	
2.1 - Get data from Spotify	**Author: JC**
The server must be able to request data from Spotify's public web API for a user. This can only happen once this user has connected to Spotify through our system successfully. Our system must send a message to the Spotify web API containing the authorisation token for that user (see 'Connect to Spotify'). On success, Spotify's web API will return an access token that can then be used to make requests to the API for a certain amount of time.	Priority: HIGH Dependencies: None
3 - Data Storage and Processing	
3.1 - Get data from Spotify	**Author: TD**

Continued on next page

Continued from last page

The server must process the data received from the Spotify API and take all the data that will be used and put it into a database in an efficient intermediate format keeping only the data we need and not storing any redundant data which offers little to no value to our client.	Priority: HIGH Dependencies: 2.1
3.2 - Recommend songs based on trends	**Author: FD**
The system should be able to recommend songs to the user based off of mood patterns or based off of genre that the user likes to listen to.	Priority: MED Dependencies: None
4 - Client	
4.1 - Create a new account	**Author: SC**
The server must allow the user to create a new account. The user will be required to enter a username and a password when creating an account. The information will be sent to the server which will check it against the existing users' usernames. If the username they entered already exists, the user will receive an error asking them to enter another username. If the username they entered is valid, a new account will be created by adding the new user's information to the users list (the username and the hashed password). A new file will be created and linked to the user in which future data will be stored. The user will receive a message saying that their account has been created.	Priority: HIGH Dependencies: 3.1
4.2 - Connect to Spotify	**Author: JC**
The client must request that the user connects their account to their Spotify account. The user will be redirected to a Spotify webpage showing the data that is being requested by our system, and will be asked to first login to Spotify, and then authorise our system to use the relevant data. If successful, a 'success' callback message will be sent to the server containing an authorisation code, meaning our system can now make requests to the Spotify web API to request that user's music data. If unsuccessful, a 'fail' callback message will be sent to the server.	Priority: HIGH Dependencies: None
4.3 - Login to the Server	**Author: SC**
The server must allow users to log into their accounts. The user will need to enter their username and password. The server will then try to find the given username in the usernames list. If it is not found the user will receive an error. If it is found, then the supplied password will be hashed and compared with the stored hash. If the password is correct the user will be logged in and will have access to his data, otherwise the user will receive an error.	Priority: HIGH Dependencies: 1.1, 3.1, 4.1
4.4 - Display data to the user	**Author: TD**
Reformat data into a set of interesting visual representations which can be displayed to the user, data for these visual representations should be provided only when they are actually requested by the user so as to save the user bandwidth and reduce server load, these should also follow the guidelines for visual data as set out in the research and each should have a specific aim such that they provide the user with understandable and usable data	Priority: HIGH Dependencies: 1.1, 3.1, 4.1, 4.2, 4.3
4.5 - Request data from server	**Author: SD**

Continued on next page

Continued from last page

The web app must be able to request data from the server for whatever process it would like the data for. This data should be able to be requested for specific data, for example, song details, song name or even everything in a database, etc. This is useful as it can restrict the amount of data to be sent over to the network.	Priority: HIGH Dependencies: 2.1, 4.3

Design

6.1 High Level Architecture Diagram

TODO

6.2 Project Outline - JC

6.2.1 Classes

Server:

Spotify

The Spotify class contains all methods for interacting with the Spotify API. This is used in a variety of situations, such as the monitor thread that constantly gets updated information on users or the methods to retrieve the mood information about tracks.

Processing

The Processing class contains all methods for processing the data either saved in the database or pulled from the Spotify API. This processed data is usually then passed to the API and given to the client.

Database

The Database class contains all methods and data structures for the database. It includes the data structure for each table within the database, the connection code to the online MongoDB database we are using, and all methods to get and put data to and from the database as needed.

API

The API class handles all requests from the client for data. It ensures security using web tokens and sessions, and only allows access to sensitive data when the user connecting successfully authenticates themselves.

Client:

GUI

The GUI will be responsible for displaying the data sent from the server side. It will provide the interface for the user to interact with. The Google Charts API will be used to display the data received from the server as useful, graphically pleasing metrics.

API

The API methods will be responsible for interfacing with the server side REST API, both passing data to the server and requesting data from the server.

6.2.2 Methods

Spotify:

FinaliseAuth

FinaliseAuth passes the user specific authorisation code to the Spotify API, upon which an access token and a refresh token should be received and returned.

GetTracksInfo

GetTracksInfo is a method for getting information (track name, artists, link) from the Spotify API about a list of tracks from their IDs.

GetAudioFeatures

GetAudioFeatures is a method for getting audio information (valence, energy, acousticness, instrumentalness, danceability) from the Spotify API about a list of tracks from their IDs.

GetAccessToken

GetAccessToken is a method that checks if the access token is still valid. If it is, it simply returns the access token. Otherwise, it uses the refresh token to generate a new access token, and then returns that token.

DataListener

The DataListener method runs on a timer from when the server starts running, and every thirty minutes requests all recent tracks that each user in the database has listened to. It then passes these to the database functions to save them into the database.

GetAllTracks

GetAllTracks is a method that returns a list of track IDs that contain every song in every playlist the user has, as well as all of their liked songs.

Database:

AddListenInfo

AddListenInfo adds a new "listen" to the "Listens" table - it adds a record for each track that a user has recently listened to, including the track's ID and the user's ID.

GetListenInfo

GetListenInfo gets all tracks that a user has listened to within a specified time period.

AddUser

AddUser adds a user to the "Users" table.

GetUsers

GetUsers gets all users from the "Users" table.

GetAuthToken

GetAuthToken gets the Spotify API auth token for a given user.

GetRefreshToken

GetRefreshToken gets the Spotify API refresh token for a given user.

UpdateAuthToken

UpdateAuthToken modifies the Spotify API auth token for a given user.

CheckLoginDetails

CheckLoginDetails checks that, given a username, such a user exists in the "Users" table, and given a hashed password, a hash exists in that user record that is the same.

CheckIfUserExists

CheckIfUserExists checks if a given username already exists in the database.

Processor:

SortUserSongs

The SortUserSongs method uses the GetAllTracks method to get all of a user's tracks, before using the GetTracksInfo method to get those tracks' names and artists, and then the GetAudioFeatures method to get those tracks' audio features. Each track is then classified as a specific mood using the Classify method, and is returned through an API call.

ProcessRecentTracks

ProcessRecentTracks uses the GetListenInfo method from the database to get all recently listened to tracks, before using the GetTracksInfo method to get those tracks' names and artists, and then the GetAudioFeatures method to get those tracks' audio features. Each track is then classified as a specific mood using the Classify method, and is returned through an API call.

Classify

The Classify method looks at the audio features of a song and classifies it as a specific mood using a mood defining algorithm.

API:

Login

The Login API endpoint is given a username and password from the client, and uses the CheckLoginDetails method in the database to confirm that a user is valid. If it is, a web token is generated and sent as a cookie to the user, creating a login session for the client.

Register

The Login API endpoint is given a username and password from the client, and uses the CheckIfUserExists method in the database to confirm that no user exists with that name. If they do, "false" is sent back, alerting the client that a user already exists with that username. Otherwise, the AddUser method in the database is ran, with the username and password given.

YesterdayMood

The YesterdayMood API endpoint uses the ProcessRecentSongs method to get a list of songs that the user listened to in the last day, information about each song and a mood classification for each song.

MonthMood

The MonthMood API endpoint uses the ProcessRecentSongs method to get a list of songs that the user listened to in the last month, information about each song and a mood classification for each song.

Callback

The Callback API endpoint is used exclusively for Spotify to return an authorisation code for a given user after that user connects their Spotify account to our system. It then uses the UpdateAuthToken method in the database.

GetTracks

The GetTracks API endpoint uses the GetAllTracks method and returns that data, for use with the timeline on the front end.

MoodClassification

The MoodClassification API endpoint uses the SortUserSongs method and returns that data, for use with the mood charts on the front end.

6.3 Design Sketches

Omit Earlier Sketches TODO?

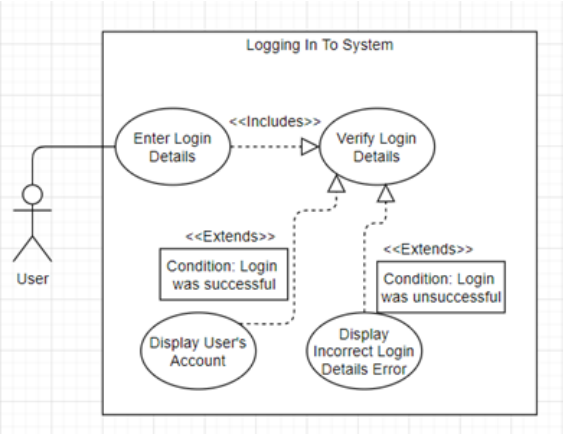
6.4 Class Diagram

Needs updating

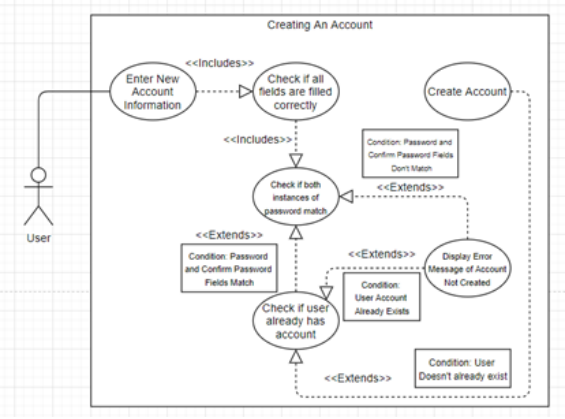
6.5 Use Case Diagrams - FD

There are not a large number of use cases for our program. All of the data enrtry is automatic so there are no use cases that allow for the user to edit their data. This means we end up with a small number of use case scenarios, which helps to simplify user experience design

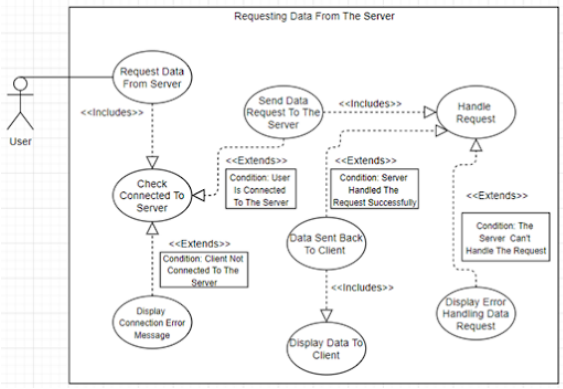
6.1a shows the typical path a user would take when logging in, should they not have an account then they would have to follow 6.1b to create one first and then they can view their data, following 6.1c



(a) Use Case Scenario 1



(b) Use Case Scenario 2



(c) Use Case Scenario 3

Figure 6.1: Different Use Case Scenarios

Testing

7.1 Testing Plans

Name	Description	Data Type	Actual Data	Expected Outcome
Sprint 1				
Acquiring data from Spotify				
Authorise Spotify user account	Program should be able to request access to a user's Spotify account and store the returned access token for later use	Valid Spotify username / password combination		Valid Spotify access Token
Data successfully obtained from Spotify	Show that data collection from Spotify functions correctly, this does not include the account authorisation stage, but rather that the server can obtain data form Spotify, what data does not matter as long as it is the same as requested	Valid access token and API query		Spotify returns specified data
Login Pages				
Login and Create Account pages	Login and Create Account pages are sent to the user and are visually functional when the user requests them using http	Page URLS		Server should send the pages through http
Create Account successfully creates account when valid account data is provided	Create Account page successfully sends correct data to the server, allowing it to correctly create accounts, this must also contain all of the necessary data, and return a correct Account Creation page and message, when all the data sent is valid	Valid and not already used user data		Server should send a success message or send the user to an account created page

Continued on next page

Continued from last page

Create Account correctly displays error message when invalid account data is provided	Create Account page throws an error when the user enters data which is not valid or uses data which has been already used and does not overwrite or data the already existing data	Invalid or already used user data		Server should stay on page and send an error message to the user, telling them what caused the error
Login is successful when correct account credentials are entered	Login page should log the user in when they enter the correct username / password pair	Valid username / password combination		The user is logged in, redirecting to a landing page
Login is unsuccessful when incorrect credentials are entered	Login should fail when the user enters credentials which are not valid, the page, should not give away whether an account exists with the given username / email but rather a generic failed message in order to comply with security best practises	Invalid username / password combination		The user is not logged, however is left on the log in page with a generic error message
Sprint 2				
Security				
Passwords are hashed and stored	Passwords are hashed and stored as outlined in password storage guidelines (Research: 3.3): Salted and hashed (argon2 was recommended), with only these stored in databased	Register account with valid credentials, check database to ensure plaintext password not stored	N/A - Can be checked on database	Password stored in database will be hashed, stored alongside the salt
Hash salts should be safely generated	Password salts should be generated in such a way which makes them hard to guess (ex. Not generated based on other account info, or using Cs generator without a seed)	Register account with valid credentials, check database to show	N/A - Can be checked on database	List of randomly generated salts, (Note that even easy to guess salts will appear random, so code review is best to established this has been met)

Continued on next page

Continued from last page

Security				
Server is able to retrieve song characteristics from Spotify	Server can gather characteristics on a song using the Spotify API, these can then be used in order to classify the songs mood	Song ID passed to Spotify	https://api.spotify.com/v1/audio-analysis/51xPEClVo9DI6GhrtnFc6DF	Relevant data for song (Sally Cinnamon, The Stone Roses)
Server is able to characterise songs	Server can characterise any given song based on the statistics provided by the Spotify API	Song data provided by Spotify	N/A - can be checked through API	All songs users have in library should be categorised
Server tracks users listening habits	Server periodically checks to see what the client has listened to and then and adds these to the database	Check through API (See API Section)	N/A - can be checked through API	Users listened to songs should appear
Server tracks contents of user's library	Server gathers user's library of songs from the playlists they have	Check through API (See API section)	N/A - can be checked through API	All songs in a user's playlists should appear
Server characterises user listening habits	Server should characterise songs the user has listened into moods based on the characteristics of those songs as outlined in mood table (Research: 3.5)	Check through API (See API section)	N/A - can be checked through API	Mood should appear alongside listened songs, these do not have to be perfect but should give a good representation
API				
API should show no data if the user is not logged in	Requests made without a valid auth token should either be turned away or ignored	Test API pages before logging in	Try: JOE CORRECT URL PLZ without auth token	No data will be displayed
API allows for access to categorisation of user songs	API should provide a method of viewing user data of library (note that this does not have to be the final page with a user facing interface)	API page link	JOE CORRECT URL PLZ	All user data for library and its characterisation should be displayed
Sprint 3				
Frontend Navigtion				

Continued on next page

Continued from last page

Frontend navigation functions correctly	User can navigate to all of the different pages of the frontend using just the buttons provided	Website	¡Insert landing page here¿	Page should appear as per design and should be easy to navigate with on-site tools (no manual entering of URLs should be required)
Frontend redirects user if not logged in	Frontend redirects the user should they try and access pages they do not have access to	Website	¡Insert landing page here¿	User will be redirected to the login / create account page should they try and access any page which requires account credentials
Security				
User should not be able to view / access other users' data	Frontend should not expose any user data unless the user is logged into that user account with valid credentials	Website	¡Insert landing page here¿	User will be shown no data if they do not have valid credentials and should be redirected (see above)
Data Representations				
User should be able to view all songs in their library	Webpage should provide a method for the user to view all the items in their library along with moods and view counts	Website	¡Insert landing page here¿	User library should be displayed if user has correct login credentials
User should be able to view a timeline of their recently listened to songs	Webpage should provide a method for the user to view all their recently listened to track in a timeline format	Website	¡Insert landing page here¿	Timeline of user listens should be displayed if user has correct login credentials

Continued on next page

Continued from last page

User should be able view statistics on the makeup of their library and listening habits	Webpage should provide a method for the user to view statistics such as the mood makeup of their library and last day of listening	Website	¡Insert landing page here¿	Statistics on library and listen makeup should be displayed on page if the user has correct login credentials

Appendix

8.1 Figures

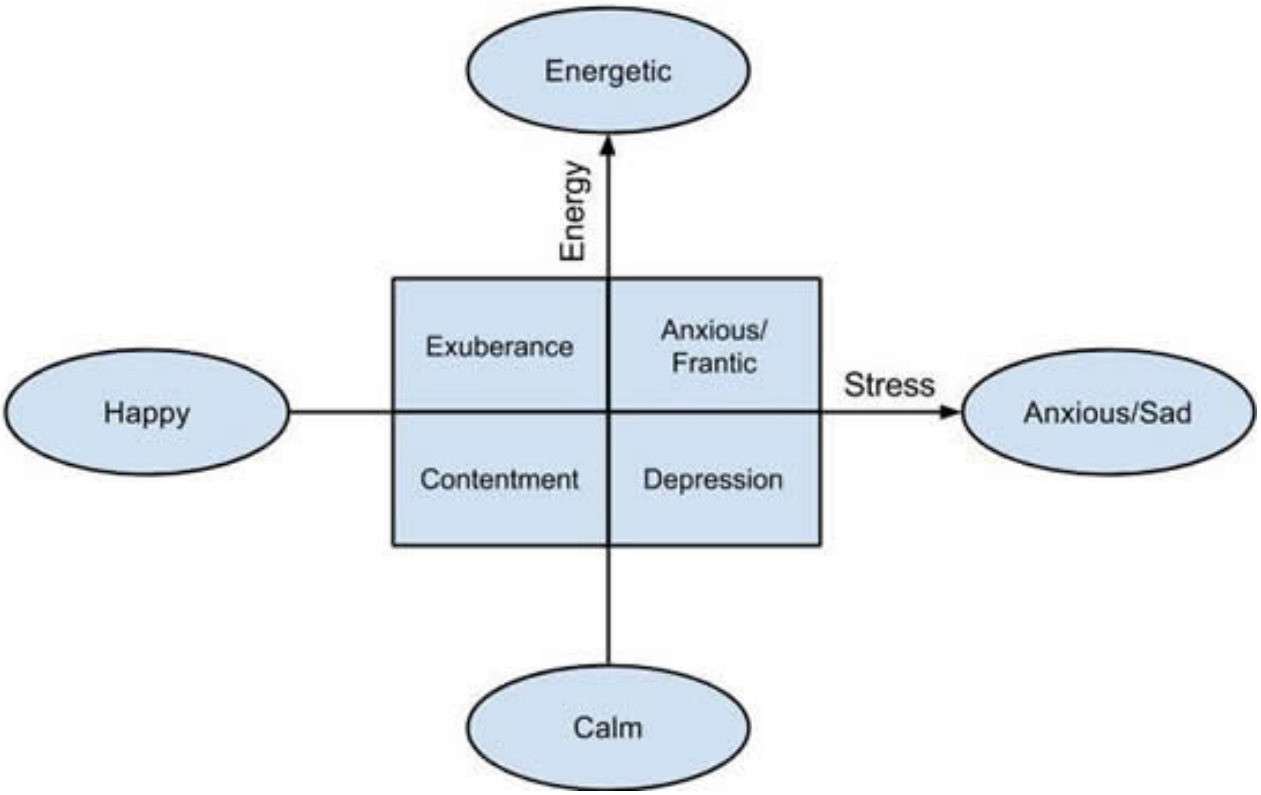


Figure 8.1: Thayer Mood Model

Bibliography

[Adam Satariano and Nicholas Confessore(2018)] Adam Satariano and Nicholas Confessore, 2018. *Cambridge Analytica's Use of Facebook Data Broke British Law, Watchdog Finds* [Online]. Unpublished. Available from: https://www.nytimes.com/2018/11/06/technology/cambridge-analytica-arron-banks.html.

[Association for Computing Machinery (ACM)(2018)] Association for Computing Machinery (ACM), 2018. *ACM Code Of Ethics And Professional Conduct* [Online]. Unpublished. Available from: https://www.acm.org/code-of-ethics.

[Biryukov, Dinu and Khovratovich(2016)] Biryukov, A., Dinu, D. and Khovratovich, D., 2016. Argon2: New generation of memory-hard functions for password hashing and other applications [Online]. *2016 IEEE european symposium on security and privacy (EuroS&p)*. IEEE. Available from: https://doi.org/10.1109/eurosp.2016.31.

[Eerola and Peltola(2016)] Eerola, T. and Peltola, H.R., 2016. Memorable Experiences with Sad Music—Reasons, Reactions and Mechanisms of Three Types of Experiences. *Plos one* [Online]. Introduction. Available from: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0157444.

[Fritz et al.(2014)Fritz, Huang, Murphy and Zimmermann] Fritz, T., Huang, E.M., Murphy, G.C. and Zimmermann, T., 2014. Persuasive technology in the real world [Online]. *Proceedings of the 32nd annual ACM conference on human factors in computing systems - CHI '14*. ACM Press. Available from: https://doi.org/10.1145/2556288.2557383.

[Horwitz(2018)] Horwitz, J., 2018. *Outside the US, the Philippines saw the most Facebook user data go to Cambridge Analytica* [Online]. Unpublished. Available from: https://qz.com/1245355/outside-us-philippines-saw-most-facebook-user-data-go-to-cambridge-analytica/.

[Humphrey, Bello and LeCun(2013)] Humphrey, E.J., Bello, J.P. and LeCun, Y., 2013. Feature learning and deep architectures: new directions for music informatics. *Journal of intelligent information systems* [Online], 41(3), pp.461–481. Available from: https://doi.org/10.1007/s10844-013-0248-5.

[IFPI(2019)] IFPI, 2019. *Music Listening 2019: A look at how recorded music is enjoyed around the world* [Online]. Unpublished. Available from: https://www.ifpi.org/downloads/Music-Listening-2019.pdf.

[Izzard(2019)] Izzard, H., 2019. *https://www.theguardian.com/culture/2019/dec/06/spotify-wrapped-users-alarmed-by-their-own-listening-habits* [Online]. Unpublished. Available from: https://www.theguardian.com/culture/2019/dec/06/spotify-wrapped-users-alarmed-by-their-own-listening-habits.

[Jaimes, Murray and Raij(2013)] Jaimes, L.G., Murray, T. and Raij, A., 2013. Increasing trust in personal informatics tools [Online]. *Design, user experience, and usability. user experience in novel technological environments*. Springer Berlin Heidelberg, pp.520–529. Available from: https://doi.org/10.1007/978-3-642-39238-2_57.

[Kizza(2016)] Kizza, J., 2016. *Ethics in computing : a concise module*. Switzerland: Springer.

[Li, Dey and Forlizzi(2010)] Li, I., Dey, A. and Forlizzi, J., 2010. A stage-based model of personal informatics systems [Online]. *Proceedings of the 28th international conference on human factors in computing systems - CHI '10*. ACM Press. Available from: https://doi.org/10.1145/1753326.1753409.

[Nauert(2018)] Nauert, R., 2018. Upbeat Music Helps Improve Mood. *The journal of positive psychology* [Online]. Introduction. Available from: https://psychcentral.com/news/2013/05/16/upbeat-music-helps-improve-mood/54898.html.

[Nuzzolo(2015)] Nuzzolo, M., 2015. *Music Mood Classification* [Online]. Tufts University. Available from: https://sites.tufts.edu/eeseniordesignhandbook/2015/music-mood-classification/.

[OWASP(2020)] OWASP, 2020. *Credential stuffing* [Online]. Unpublished. Available from: https://owasp.org/www-community/attacks/Credential_stuffing.

[Paas and Castro(2013)] Paas, A. and Castro, A., 2013. *Music Informatics: A Modern Approach to Studying Music* [Online]. Unpublished. Available from: https://musicmindandbrain.wordpress.com/2013/01/27/music-informatics-a-modern-approach-to-studying-music/.

[Rapp and Cena(2014)] Rapp, A. and Cena, F., 2014. Self-monitoring and technology: Challenges and open issues in personal informatics [Online]. *Universal access in human-computer interaction. design for all and accessibility practice*. Springer International Publishing, pp.613–622. Available from: https://doi.org/10.1007/978-3-319-07509-9_58.

[Rapp and Cena(2016)] Rapp, A. and Cena, F., 2016. Personal informatics for everyday life: How users without prior self-tracking experience engage with personal data. *International journal of human-computer studies* [Online], 94, pp.1–17. Available from: https://doi.org/10.1016/j.ijhcs.2016.05.006.

[Rathi(2019)] Rathi, R., 2019. *Effect of Cambridge Analytica's Facebook ads on the 2016 US Presidential Election* [Online]. Unpublished. Available from: https://towardsdatascience.com/effect-of-cambridge-analyticas-facebook-ads-on-the-2016-us-presidential-election-dacb5462155d.

[i SAFE(2004)] SAFE i, 2004. *Cyber Bullying: Statistics and Tips* [Online]. Unpublished. Available from: https://auth.isafe.org/outreach/media/media_cyber_bullying.

[Security ethics(2010)201] 2010. Security ethics. *Nature* [Online], 463(7278), pp.136–136. Available from: https://doi.org/10.1038/463136a.

[Shannon(1949)] Shannon, C.E., 1949. Communication Theory of Secrecy Systems. *Bell system techinal journal* [Online], 28. Available from: https://archive.org/details/bstj28-4-656/mode/2up.

[Somerville(2019)] Somerville, E., 2019. *Spotify Wrapped 2019: here's how to see your most played song statistics for the year and decade* [Online]. Unpublished. Introduction. Available from: https://inews.co.uk/culture/spotify-wrapped-2019-when-date-statistics-year-decade-review-top-songs-most-played-1330397.

[*Spotify WebAPI Documentation*(2020)Web] *Spotify WebAPI Documentation*, 2020. [Online]. Introduction. Available from: https://developer.spotify.com/documentation/web-api/.

[Swant(2019)] Swant, M., 2019. *Spotify Rolls Out New 'Wrapped' Campaign To Help Users Remember Their Decade Of Music* [Online]. Unpublished. Introduction. Available from: https://www.forbes.com/sites/martyswant/2019/12/17/spotify-rolls-out-new-wrapped-campaign-help-users-remember-their-decade-of-music/.

[Tsudik(1992)] Tsudik, G., 1992. Message authentication with one-way hash functions. *ACM SIGCOMM computer communication review* [Online], 22(5), pp.29–38. Available from: https://doi.org/10.1145/141809.141812.