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 Song attributes and mood - SD

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

SANDRA PLEASE DO THIS LOVE YOU XXX

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	+	+	/

Risk Assessment - LG

Personal informatics applications are already a market that exists, 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 incapacitate a member of our group, 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. Coronavirus would not only affect our group's members but would also affect our potential audience. This may lead to a change in the types of music that users would listen to, as well as potentially change how much they listen to said music.

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.

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 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.

5.3 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.

5.4 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**
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**

Continued on next page

Continued from last page

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**
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

Testing

6.1 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

Bibliography

[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.

[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.

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 Unpublished.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

[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.

[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.