Potential Features list:

- Serverside listener

- Listen = 30 seconds+

- Database

- Make an object class

- Setup tables

- Connect to Spotify Web API

- Setup Web API for client

**Essentials:**

* See minute by minute, day by day breakdown of what they listen to and when
  + Can filter through
  + Looks nice
* Weekly / Monthly rewind stats
  + Listen time (number of plays, when they listened)
    - Breakdown by genre and mood
    - Breakdown by song
* User asked how different music makes them feel
* Advanced features:
  + Grouping their music based on mood

**Would be nice:**

* Suggest songs based on mood
  + Maybe using “recommendations by genre”

Functional Requirements

|  |  |  |
| --- | --- | --- |
| ***1. Requirement Name - Server*** | | |
| 1.1 | Send data to client when requested | Author: Sam Davidson |
| **Description:** 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:  Dependencies: |
| ***2. Requirement Name - Data Collection*** | | |
| 2.1 | Get data from Spotify | Author: Joseph Cryer |
| **Description:** 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:  Dependencies: |
| ***3. Requirement Name - Data Storage and Processing*** | | |
| 3.1 | Store data from Spotify for each user | Author: Teodora Dinca |
| **Description:** 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:  Dependencies: |
| ***4. Requirement Name - Client*** | | |
| 4.1 | Create a new account | Author: Sandra-Maria Corradi |
| **Description:** 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:  Dependencies: |
|  | Connect to Spotify | Author: Joseph Cryer |
| **Description:** 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:  Dependencies: |
| 4.3 | Login to the server | Author: Sandra-Maria Corradi |
| **Description:** 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:  Dependencies: |
| 4.4 | Display data to the user | Author: Teodora Dinca |
| **Description:** 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:  Dependencies: |
| 4.5 | Request data from server | Author: Sam Davidson |
| **Description:** 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:  Dependencies: |

Non-Functional Requirements

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |