Project Evaluation Report

Chris Doster, Max Quaranta, Jacob Middaugh, Tanner Coleman

Capstone Project Spring, 2021

The University of West Florida

4/30/2021

CIS 4592

Dr. Bernd Owsnicki-Klewe

Table of Contents

Table of Contents

1 Executive Summary

BerndBeats is an application that will create a custom playlist based on a user's social media profile (Twitter). The app will extract information about a user from previous tweets, profile details, or accounts that the user follows. Our application will take this extracted information and compile it to form a custom playlist with a set number of songs that the user may enjoy. The data gathered about the user must stay private and not be shared, although the user can decide if the playlist itself should be shared.

2 Requirements

2.1 Initial

Our group's initial requirements were decided based around the core functionality of our app which was the ability to input social-media data, analyze that data, and create playlists on Spotify according to this analysis. Features that were going to be implemented included a user system that stored user profiles in a database, including information about the users liked/disliked songs, preferred genres, custom settings, and anything else that might have needed to be stored. We wanted the algorithm to be able to take feedback from the user and update the user profile accordingly as well as adjustable settings so that a user could change their preferences about how many songs to include in playlists, and how many playlists should be outputted.

2.2 Final

Our group's final requirements were altered based on core functionality of our app. The app still requires a Twitter and Spotify account to work with the app. The final features that are implemented include an algorithm that creates a random selection of music based on Twitter data from the user profile. Playlists with their music that is added to the user's profile.

2.3 Comparison

Throughout the development process, our requirements changed as our group began to research important functionality of the project. Our original plans had to change drastically as we learned more about the tools we were using, and as we encountered new and unforeseen issues. The majority of the planned functionality was left unimplemented, due to time constraints. The scope of our planned functionality turned out to be much greater than anticipated, and implementation became impossible as a result.

3 Timeline

3.1 Initial

The project timeline consists of 3 sprints; the first 2 sprints being 3 weeks long, and the final sprint being 4 weeks long. The goal of each sprint was laid out as follows:

1. Establish means of data input and playlist creation
2. Design and implement analysis algorithm / Implement privacy measures for sensitive data
3. Review Progress and assess what was left to be done

3.2 Final

Due to difficulties in development as well as conflicts in personal schedules, tasks were completed on a schedule lagging behind that of the expected timeline. Strides were made during each sprint, but the speed of progress was slower than needed for the project. Some major points of accomplishment during each sprint were as follows:

1. Data input method established
2. Playlist creation implemented / Token system created
3. Analysis algorithm designed, partially implemented

3.3 Comparison

Due to many bumps and roadblocks during the development process, less progress was made than initially planned. As development progressed farther, it became apparent that the scope of the project was greater than was initially believed. Learning to use the Spotify token authentication system proved to be more difficult than expected, being implemented a sprint later than planned. Entering sprint 3, we reviewed our progress and began prioritizing high-value functionality like that of the Analysis algorithm. With the time available to us, and a full-time school schedule for each team member we were unable to implement all functionality that we initially intended.

4 Expectations vs. Results

Our expectations changed dramatically as more of the project was implemented. At first, we planned to have a GUI for our application, but as we continued development on the token system we decided it would be beneficial to instead use Flask to create an application more like a web application rather than a desktop application. Of course with this change, new hurdles arose for us to overcome. We would have liked for there to have been more functionality, such as an in-depth feedback system for recommended songs that are generated. Features like this would display more functionality to a user or client interested in the project.

5 Process Review

The development process chosen for this project was based on the SCRUM model of project development. This choice was not necessarily an inappropriate choice, but not enough detail was provided to the actual process of development. The vague nature of the initial process outline led to a less focused development process during the development timeline. In retrospect, a more specific process outline could have been instrumental in completing a greater number of tasks. If development were to continue, the process outline would need to be reviewed extensively and reworked to be more specific in how development tools are used and how development is done from a developer’s perspective. Additionally, including procedures to log work could prove helpful in ensuring regular progress.

6 Remaining Work

There is quite a bit of work to be done on the application as of now. The first priority if work were to continue would be a large-scale refactor of the project’s source code to reduce coupling, increase code reuse, and improve the overall readability of the project code. After this, multiple less vital features would be implemented, such as ‘Like & Dislike’ buttons that directly influence the user’s profile. A user system would be implemented to keep track of each user’s preferences, both in the form of user-selected settings and preferences calculated by the algorithm. Our original intentions were to not add the generated playlist directly to the user’s Spotify account in case the user does not like the playlist. Realizing that we would need a refresh on our HTML skills, this became a problem that would just need to be fixed in the future, and for now, just adding the playlist directly to the user’s Spotify account upon generation would allow for testing of further functionality. We also do not have a robust token system for Twitter accounts, but would be a high-priority item if development were to continue. In the current build of the software, a particular twitter user’s information is hard-coded into the application, which is not ideal. Ultimately we would like to have a user enter their Twitter login, without needing to sign-in, to avoid the need to handle sensitive data that was unneeded. Currently, there exists a bug in song generation, where 10 songs are generated as expected, but some number of them fail to be added to the user’s playlist on Spotify. This problem is likely the result of an API endpoint for Spotify behaving in ways that we are not familiar with, but would require further testing and research to determine with any certainty. Lastly, although it is lower priority than the items previously mentioned, an improvement to the visual elements of the App, like improved styling on elements and the inclusion of logos and other images, would also be of interest to our team.