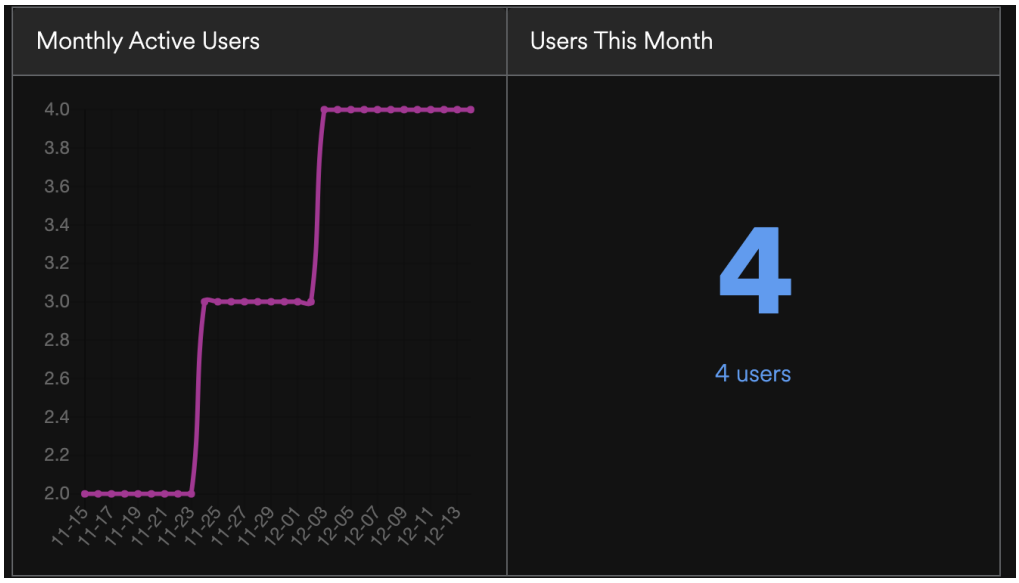
Team Reflections:

- Luisa Watkins & Kejia Wu:

For the start of this project, we first have our opening main page. This page is mainly for information about the application so the user is familiar with how everything works. This page also has credits for all the team members and the APIs used; in this case, we used both a face-API for the face detection software and the Spotify API for the playlist generation.

Once the user is familiar with the application, they can hit the start button labeled "Upload Photo". This then leads them to the next page, where the user can upload the facial image they'd like to use. Ideally, the user would already have a "selfie" ready to go by taking a photo of themselves and uploading it to their computer. By hitting the "Choose File" button, the user can choose this image from their file system. Once chosen, the image appears on the screen and the results from the face detection software are displayed. The top number indicates how much the software detects this image as a human face, while the bottom number indicates the confidence in the specific emotion detected, which is also noted at the bottom of the square. In fact, a confidence number is calculated for each of the possible detected emotions (angry, disgusted, fearful, happy, neutral, sad, surprised), resulting in an array. These values are what are used in the second half of the project in order to generate the corresponding Spotify playlist.



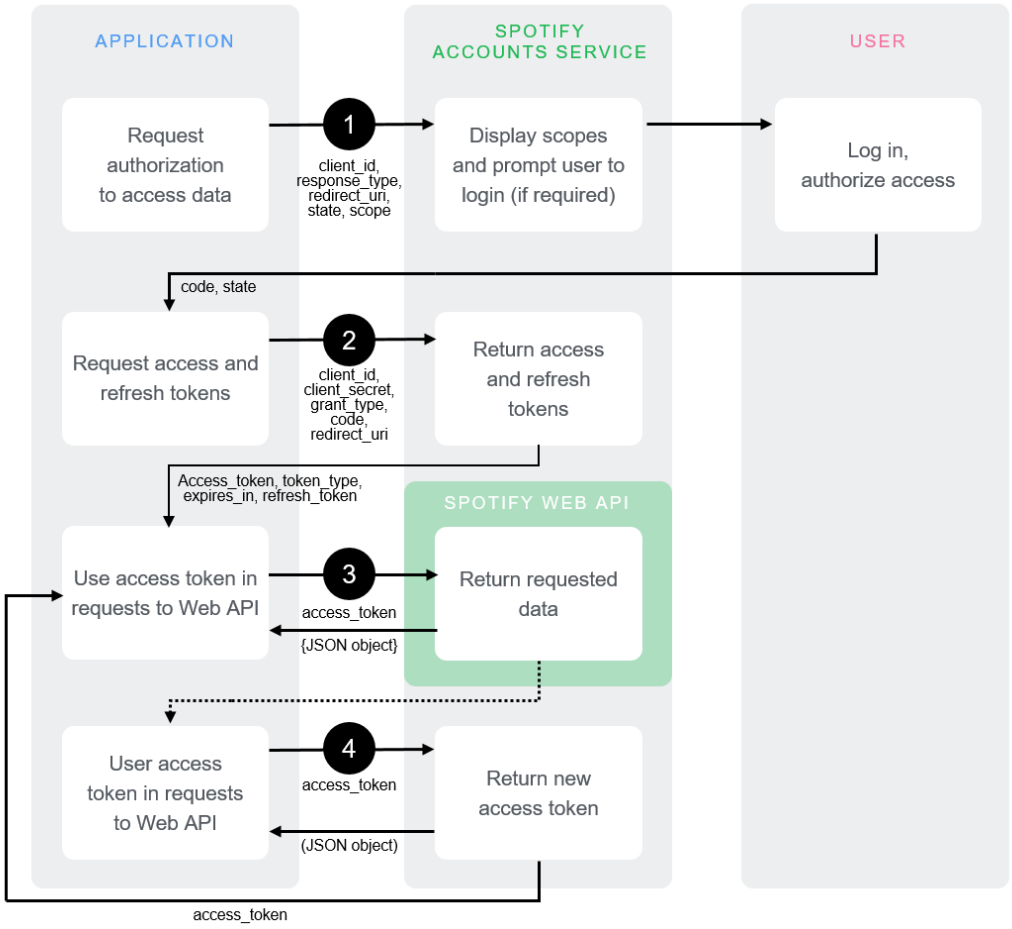
(Above is our Spotify App Statistics)

- Yanzheng Wu:

The entire app is written in Javascript and tested in Node.js. I used the Express API to facilitate manipulating HTTP calls and redirecting our app to multiple HTML webpages. In order to use the app, users need to log in to their Spotify account through OAuth2.0. I also adapted the Spotify APIs for the purpose of fetching user information and preferences(by adjusting the scope setting). If everything functions correctly, the app would receive a string input from the face API, which the string is the current user emotion. Then the app would be made an HTTP POST call to

fetch playlists corresponding to the specific emotion category(for sample outputs please refer to the demo video at the prototype folder). There is also a link that redirects the users to the Spotify default search webpage if the current app is malfunctioning.

Our group meets once a week and each meeting is around 40 minutes. We use Github to maintain source control. Originally we decided to have a relational database such as MySQL to store the user login information. However, we discovered that the HTTP cookies would simply serve the same purpose with much less workload since all we really wanted to remember was the user login info. There are no other major changes from our original user stories and the "server" folder contains our APP code.



(Above is the Authorization Code Flow of our Spotify APIs OAuth2.0)