Developer List

Jasbir Harnal (jlh2av) Watson Spivey (was9p)

Device Name/ Platform

Android: Dodrio

Project Title

Lyricize

Project Pitch

This app is a music tone analyzer tool. Using Spotify's API to access artists, albums, and songs, the app then uses MusixMatch's API to get the lyrics of the song. Then, the lyrics are sent to IBM's Watson Tone Analyzer to determine the emotions present in the song. This tool will be a way for users to analyze the emotions of their music.

Platform Justification

The decision to program on the Android platform came down to two reasons. Both my partner and I do not have Macs, so programming the app would come down to working only in the lab, something we both did not like having to do for the iOS mini app. Also, both of us have Android phones, which makes testing without the Nexus 7 possible.

Key Features

Music Search

Spotify API can be used to search songs, artists, and albums. When a search is performed the list fills with data returned from the Spotify API call. Depending on which search criteria is selected, the list fills with different objects. Clicking on a song launches Lyrics Retrieval, clicking on an artist returns the albums of that artist, and clicking on and albums returns the songs in that album.

Random Search

When a user shakes the device, the app uses a random number generator to populate the search field with the name of a song, album, or artist (depending on the selected radio button).

Lyrics Retrieval

When an item in the list is clicked on, a search is made through the MusixMatch API to retrieve the id of the song in the MusixMatch database. A search is then made using the id to retrieve the lyrics of the song.

Tone Analyzer

With the lyrics retrieved, the app sends the lyrics over to the Watson Tone Analyzer. The Analyzer sends back the emotion scores which are displayed on the results screen.

History

Previous searches are recorded, with the GPS coordinates of where the search was made. Previous searches are persistent between instances of the application through local data storage, and results of previous searches can be used by clicking the item in the history list.

Testing Methodologies

We tested the search functionality a variety of ways. First, we started with normal inputs correct for whatever the search was. We then moved to intentionally bad inputs, commonly misspelled words, and leaving the search field blank. Then, considerations were made for people who would purposely try and break the app by repeatedly clicking items or buttons. Another malicious input that was checked against was people trying to search symbols, something could mess with the API call.

Usage

On the main screen simply select the category of search that you want to do. Then type what you want to search, or shake the device to get a random suggestion, and then hit the search button. This returns on the main page a clickable, scrollable list. When an artist item is clicked the list returns the albums of that artist, and clicking on an albums returns the songs in that album, and clicking on a song in the list it returns, on a new activity, the tone results of the lyrics. Previous searches, and the location from which you searched, can be found in a separate activity, accessed via the History button. In order for the History functionality to work properly, the app must be given the Location permission through settings.

Lessons Learned

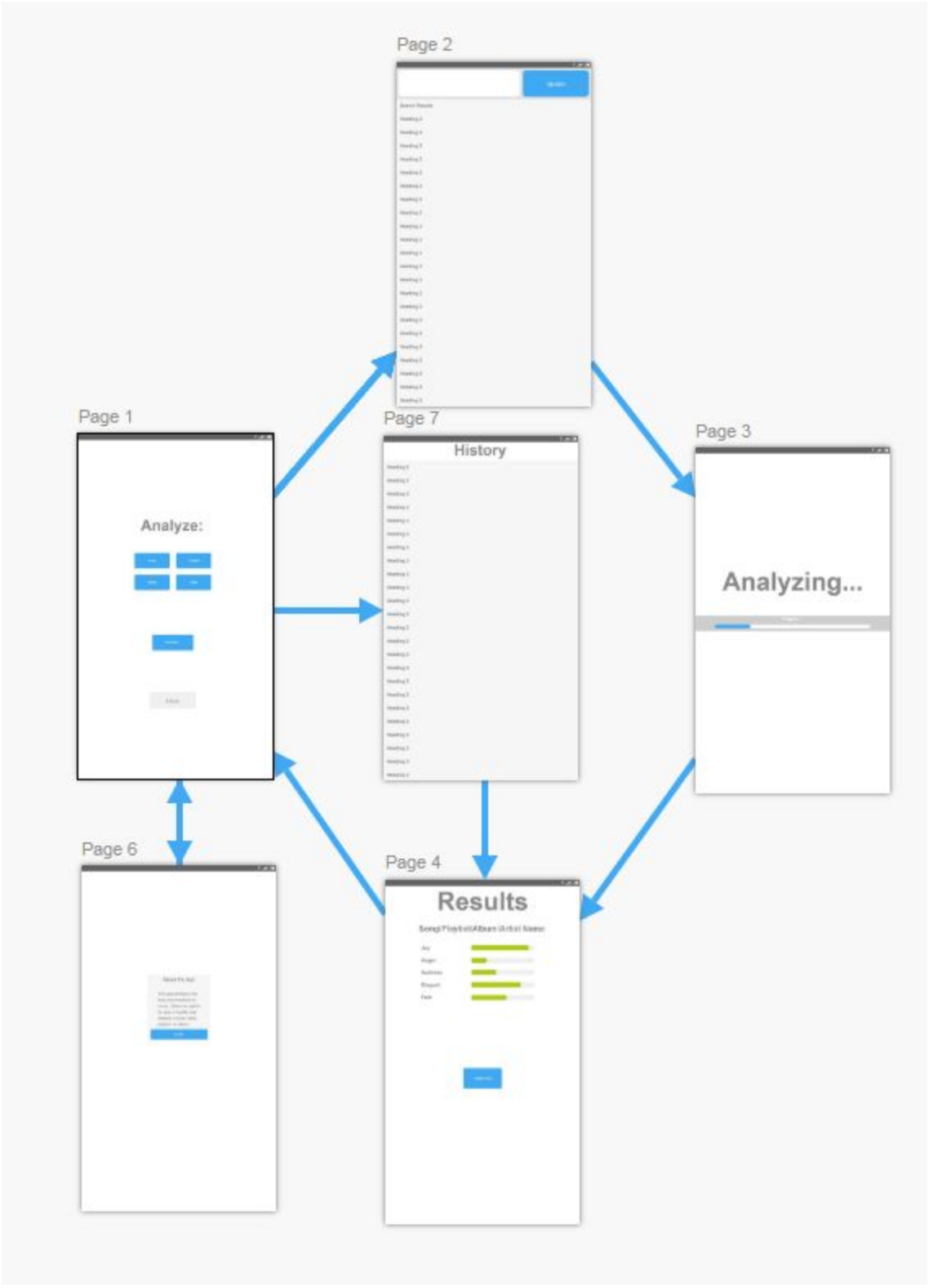
One of the big takeaways we have learned from this project is how to consume endpoints of web services and handle their output. We are using around eight API calls on the web services we are using. We also had trouble getting retrofit to work properly, and decided to build our own JSON parsing methods. It may have not been the smartest idea, but it allowed us to have a good handle on what the data coming back to us looked like. It also allowed for us to build the parsing methods to return only the data we needed for the app and not a bunch of extra information that would have to be sifted through. This process was simplified through online tools which displayed JSON files visually in a folder-like structure.

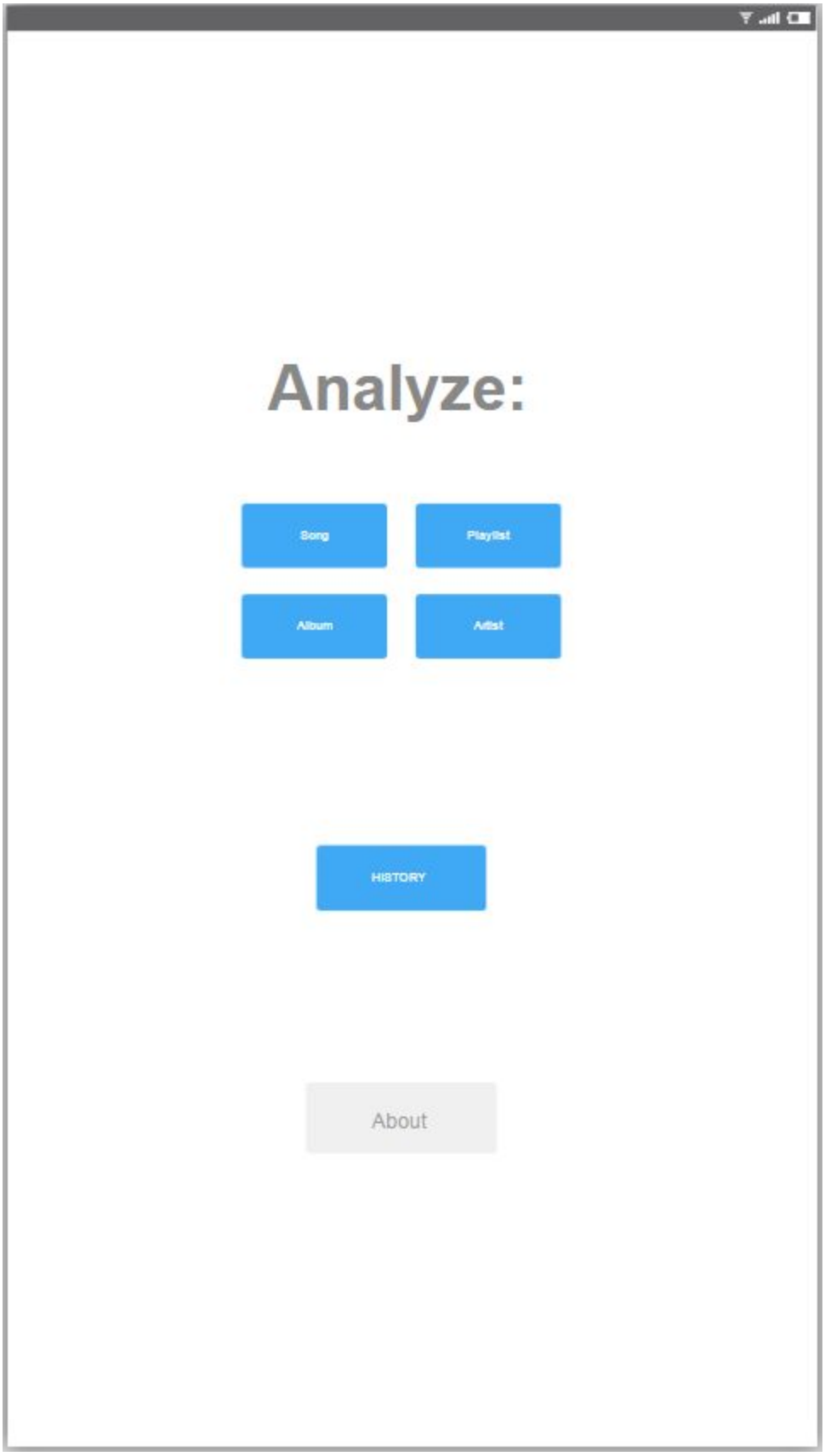
We also learned about how to be good programmers, and in the last few weeks we have been expanding the functionality of our app so that the album and artists lists can be clickable in the same way that songs are. We were able use some of the code we had already written because it was setup for almost exactly what we need to do. This made expanding the core functionality of our app easier, and we were happy that we made use of good programming practices early on that benefited us greatly later down the line. Lastly we learned about how much thought has to go into error checking, we were not the best testers for our own app because we knew exactly how it works. When it was given to roommates and friends though numerous usage problems and crashes cropped up. Because of it we became much smarter

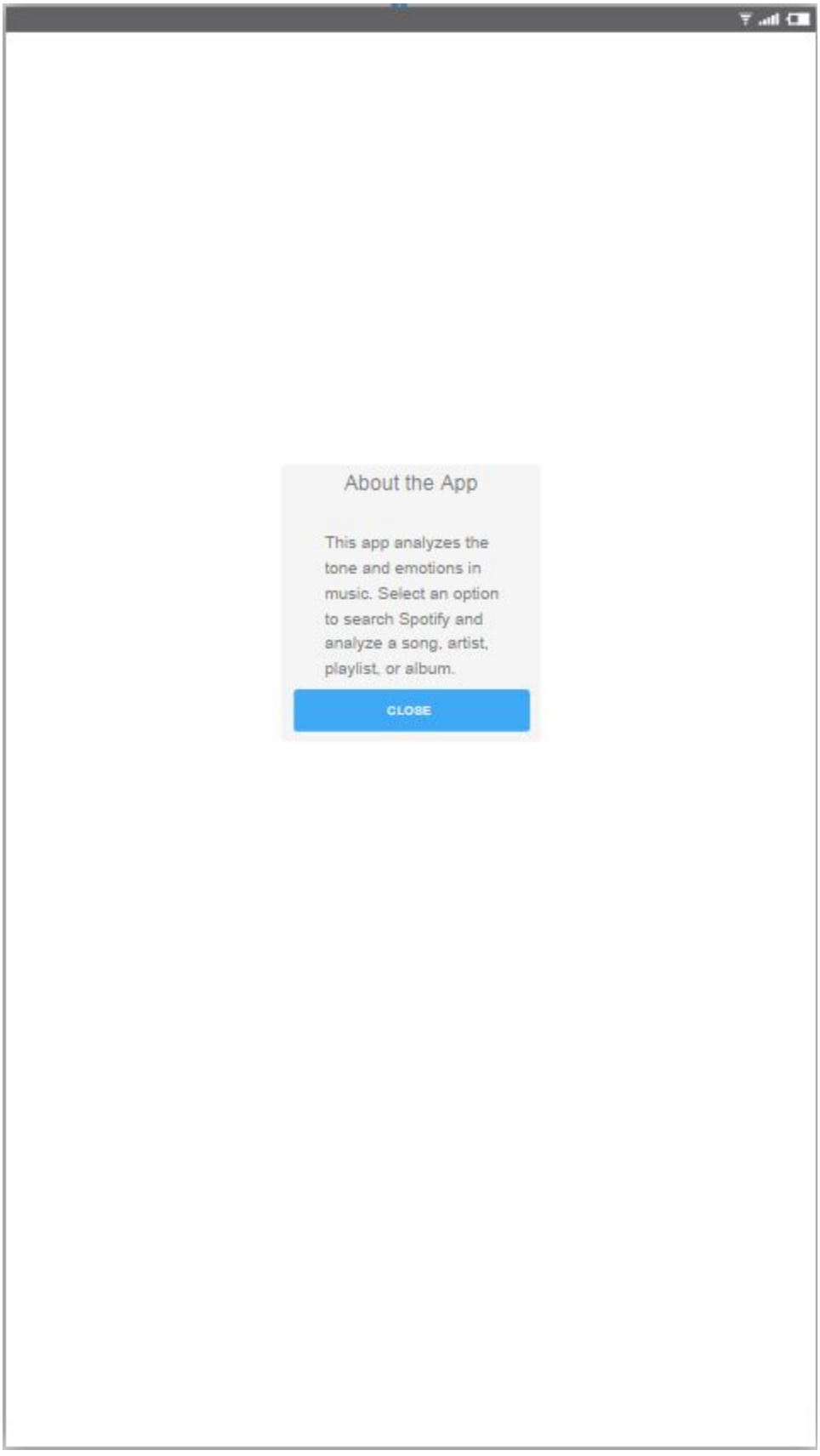
about how we handle the search field and how robust the app is to intentional or unintentional user abuse.

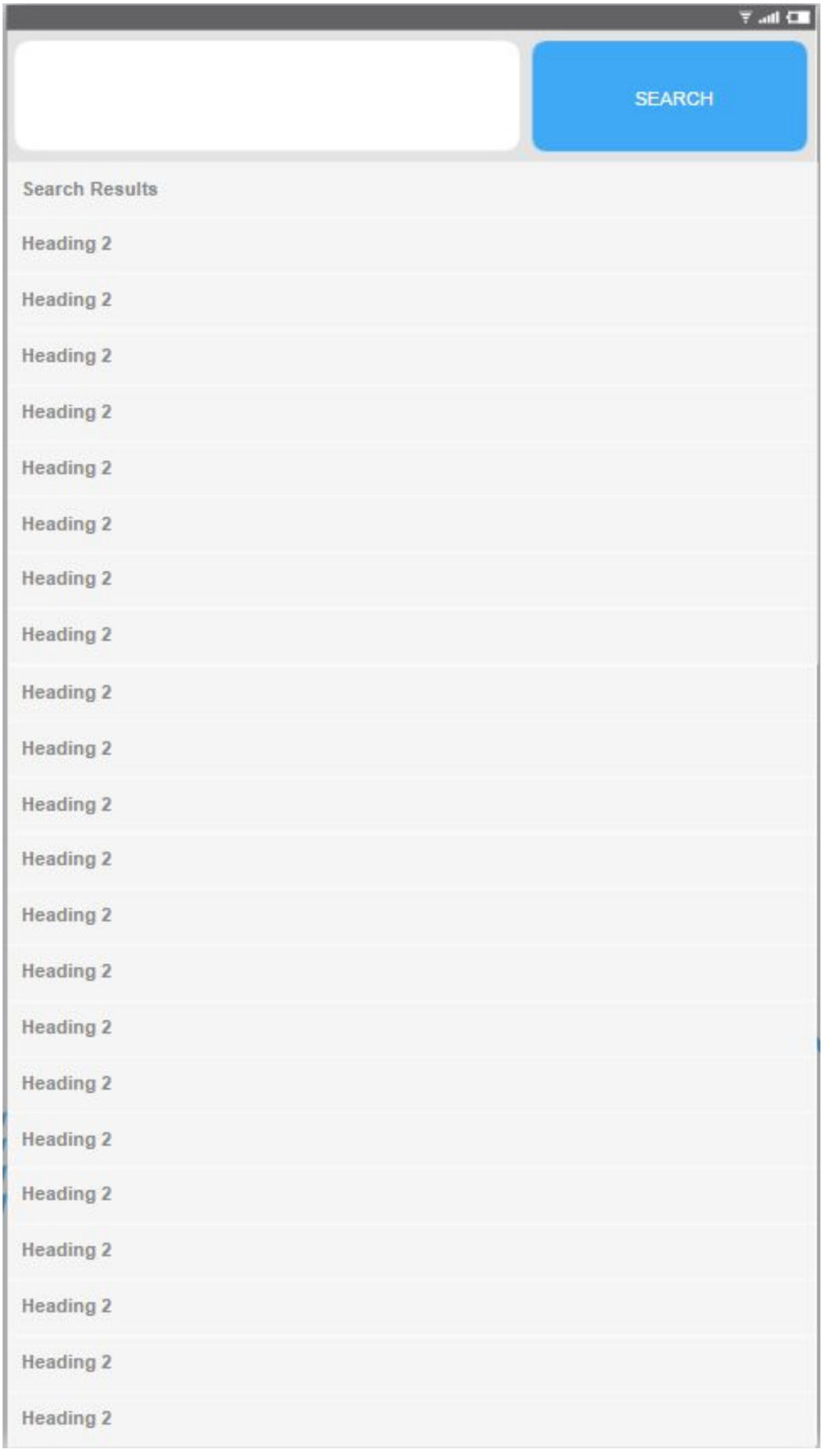
Lastly, we gained a better understanding of data flow through Android apps. Using services to retrieve data, broadcast receivers to transmit that data between parts of the app, background functions to retrieve data, and intents to pass information were all crucial to our app functionality.

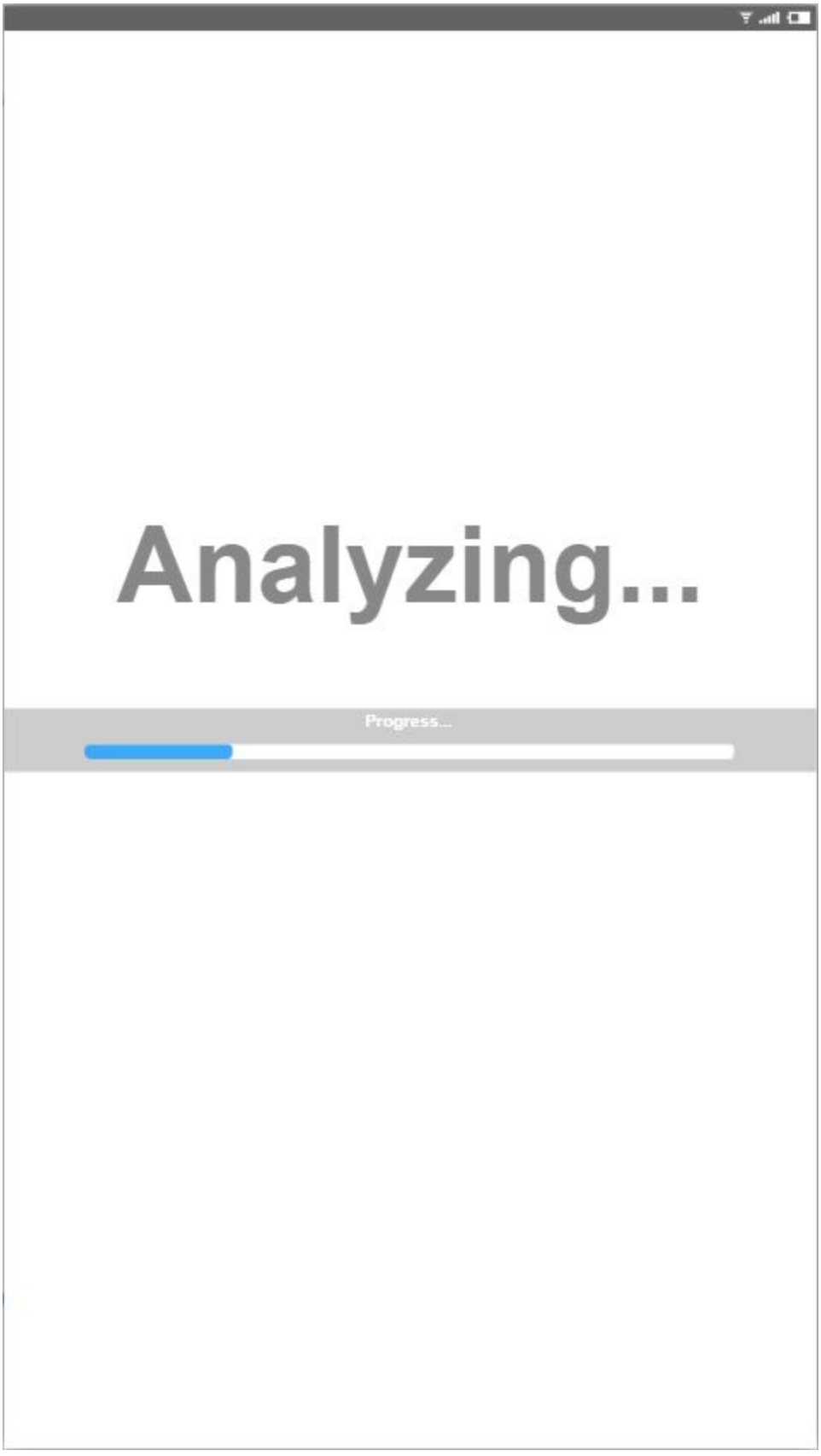
Wireframe

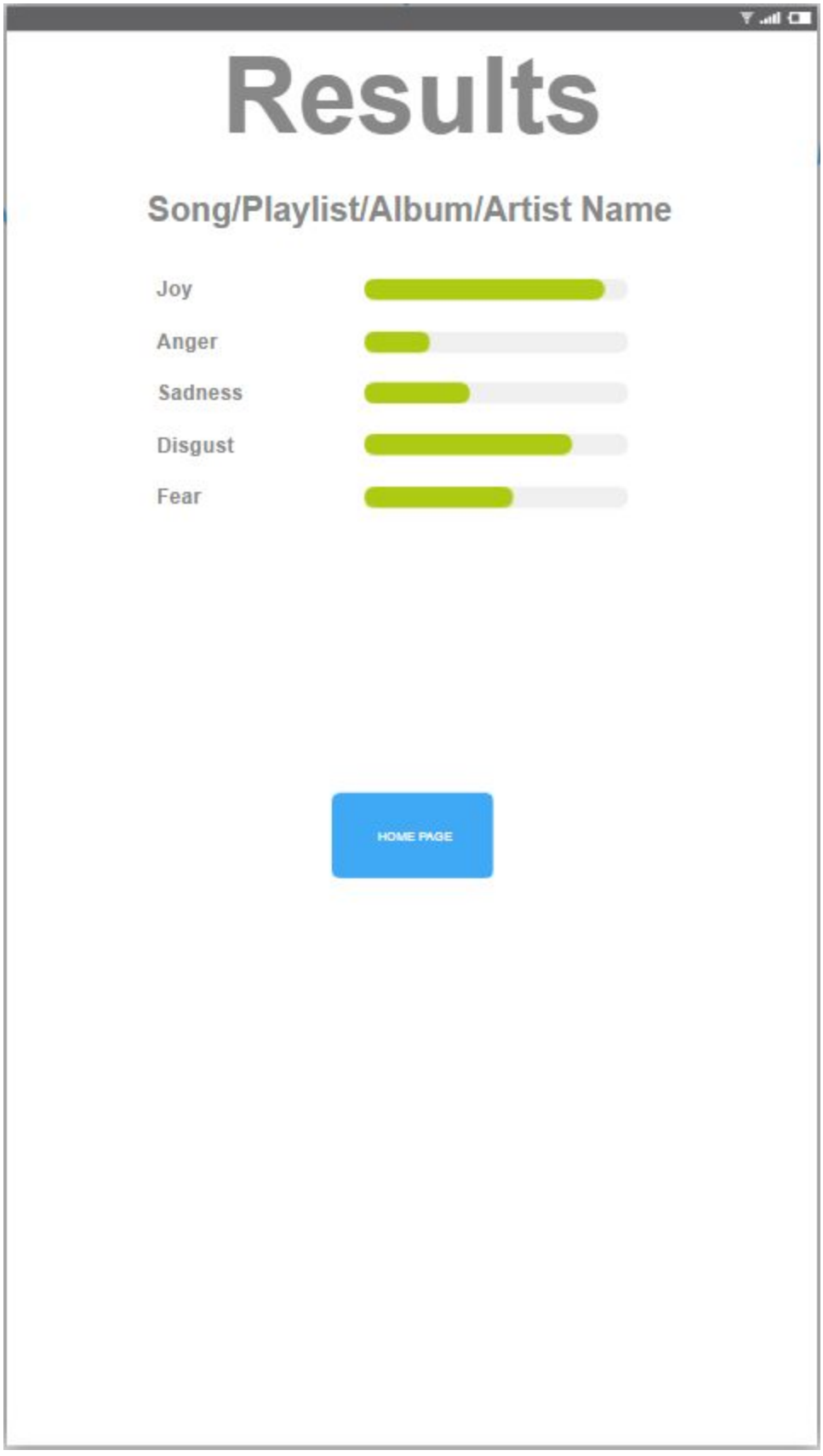

Page 2
Page 1
Page 7
History
Page 3
Analyze:
Analyzing...
Page 6
Page 4
Results


Analyze:
Song
Playlist
Album
Artist
HISTORY
About


About the App
This app analyzes the tone and emotions in music. Select an option to search Spotify and analyze a song, artist, playlist, or album.
CLOSE


SEARCH
Search Results
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2
Heading 2


Analyzing...
Progress...


Results
Song/Playlist/Album/Artist Name
Joy
Anger
Sadness
Disgust
Fear
HOME PAGE

History

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

Heading 2

The wireframe shows the basic flow and functionality of the app. The home screen presents the user with the option to search for a song, artist, album, or playlist. There is also a History button which allows the user to view previously analyzed songs, and an About button which explains what the app does. Once the user has selected the type of item that they want to search for, they are presented with a search screen where they can type in the name of what they are looking for. The search results will show up beneath the search bar. The user can then select an item from the list, and it will lead them to the Analyzing screen which displays the progress of the analysis. Once analysis has completed, the user will be presented with a Results screen which shows the levels of each emotion present in the music selected. The user can then return to the home page.