The IMAGS App

An Android Application for Studying the Effects of Music on Chronic Pain

Project Team:
Joseph Kaiser jtkaiser@wpi.edu
AmyBea Walder atwalder@wpi.edu

Project Advisor:
Professor Wilson Wong
Department of Computer Science

Co-Advisor:
Professor V.J. Manzo
Department of Music

Abstract

The goal of this project was to create an Android application that will facilitate the collection of data for the experimental stage of the Music-induced Analgesia Genome Study (IMAGS). The application provides patients who suffer from chronic pain a tool to rate their current pain level, presented on a universal numeric pain scale both before and while listening to a preferred song as selected from the Spotify library in the application, as well as record whether each patient has taken a dose of their prescribed pain medication at the end of the session and a final severity of pain rate. The study aims to determine if there are certain motifs in music, such as key, tempo, and progressions, that correlate with a decreased perception of pain, and if these musical attributes are specific to the individual or can be applicable to the general population. To this end, users can access the Spotify music service library in order to find and play songs of their choosing.

1. Introduction

According to a recent Institute of Medicine Report on Pain, 100 million Americans suffer from pain. Understandably, the treatment of such pain amounts to great costs, with half a trillion dollars spent per year (Institute of Medicine, 2011). The current mainstay of pain management remains opioid analgesics, which are a class of strong painkiller medication. Over the course of the past several years, however, both medical professionals and the general public have become increasingly concerned about the risks of severe side effects, fatal overdoses, and the high potential for addiction associated with their usage. As a result, greater interest has been shown for the pursuit of nonpharmacological or homeopathic treatment techniques with the goal of reducing the need for opioid intake (Opioid Analgesics, 2015).

One such alternative modality that has been shown to be effective in both reducing the perception of pain and reducing opioid use is the inclusion of music in conjunction with treatment. However, researches are yet uncertain as to whether there are specific musical traits or attributes, such as genre, tempo, dynamic, harmony, timbre, meter, lyrical, that have a greater effect at reducing pain.

The Music-induced Analgesia Genome Study (IMAGS) is a project which is intended to further research into this question by examining data relating patients' self-reported pain levels with information about the music they listen to during various sessions over the course of the experiment. This MQP project will create an internet-connected Android application, the IMAGS App, that the administrators of that study can deploy to participants in order to conduct each listening session and collect all of the pertinent information regarding the song selection(s) listened to, whether that individual took a dose of a prescribed medication, the pain experienced by the participant, and at what time each pain rating was recorded. The participants in the study

will be able to use a Spotify Premium account to login within this application, which will serve dual purposes of identifying that individual in records generated from each session as well as permitting them to access a huge collection of songs offered by the Spotify library. Throughout the session, the participant will update their current level of pain through a self-reporting tool based on the Wong-Baker Faces Pain Rating Scale, consisting of a linear scale with which participants either slide a marker or tap a location to adjust (Wong-Baker FACES Foundation, 2016). Participants are prompted to search for and play one or more songs, which is facilitated by a search screen built upon the Spotify Web API using a modified wrapper for Android. Afterwards, a questionnaire is presented which collects information about what medication the participant may be prescribed or have taken in conjunction with that session.

The report will be organized as so; section 2 discusses our findings from research performed on traditional opioid treatments, music as an alternative treatment approach, the leading streaming music services, and similar projects which attempted to research the integration of music with pain treatment in conjunction with a computer application. Section 3 describes the methodologies which our team followed in our work on the project, with section 4 detailing the various software, tools, and services which we utilized to do so. Section 5 goes on to elaborate the requirements we determined were necessary for the application, as well as early mockups of the user interface. In section 6, important elements of our final design are displayed within the appropriate diagrams. Section 7 explains the development of the application and the various changes which occurred over the course of the project, and section 8 contains the results of the set of evaluations performed on the final product. Lastly, section 9 describes possible extensions and further work that could be done for the project and to improve the application,

and section 10 includes a brief conclusion and summation of the experience and what was accomplished.

2. Research

2.1 Traditional Opioid Analgesics:

Chronic pain is a widespread international problem of enormous prevalence. "Chronic pain lasts longer than three months and is typically intractable, leading to decreased well-being and functional capacity" (Garcia, 2016). Domestically, it is estimated that there are more than 100 million Americans that suffer from chronic pain annually (Institute of Medicine, 2011). In addition to the extent of the population affected, there are a myriad of adverse effects upon which those who suffer from it, such as changes in mood, appetite, and energy levels, in conjunction with interfering with sleep, employment, social life, and other daily activities.

Currently, the primary method of treating moderate to severe pain is the use of opioid analgesics, a group of narcotic medications that work by binding to and blocking pain receptors. However, physicians and other healthcare professionals are becoming reluctant to prescribe such treatments. Despite the proven effectiveness of these drugs, opioid analgesics are also known to have numerous side-effects, such as respiratory depression, sedation, nausea, pruritus, and the risk of addiction (Institute of Medicine, 2015).

Over the past decade, there has been a steady increase in the number of opioid prescriptions given for the treatment of pain, and from 1991 to 2011, that number went from 76 million to an estimated 219 million. This trend has coincided with an alarming increase in the rate of opioid abuse, as well as subsequent opioid-related overdoses and hospitalizations, with an average of more than 90 Americans overdosing each day. Furthermore, prescription opioid use has been linked to the current nationwide heroin epidemic. In light of these concerns, strategies

to reduce the reliance on opioid-analgesics as the sole treatment for chronic pain are of great interest (National Institute on Drug Abuse, 2017).

2.2 Music as an Alternative Treatment

Many studies have shown that music can help individuals by increasing their pain tolerance, which in turn lessens their need for medication to treat such pain. Patients listening to music are observed to have higher analgesic responses. One study, conducted at an Iranian university hospital, showed that patients undergoing cardiac surgery who listened to their preferred music before, immediately after, and at certain intervals following expressed significantly lower ratings of pain than the control group which did not, concluding that "music can be effective as a non-pharmacological, inexpensive, non-invasive and side effect free method for pain management after open heart surgery" (The Effects of Listening to Preferred Music on Pain Intensity After Open Heart Surgery, 2012).

While it is not yet completely understood what specific attributes of music are most responsible or effective in reducing pain, research has determined familiar music provides a series of rewarding musical expectations, which trigger the release of dopamine and endorphins that reduce the perception of pain. Beta-endorphins accomplish this in both the peripheral nervous system (PNS) and central nervous system (CNS). In the PNS, beta-endorphins bind to opioid receptors, which in turn inhibits the release of a key protein substance that is involved in the transmission of pain signals in the body. On the other hand, beta-endorphins in the CNS reduce the release of GABA, an inhibitory neurotransmitter, which results in greater than normal release of dopamine, the chemical dopamine, which is associated with the feeling of pleasure and counteracts feelings of pain (Sprouse-Blum, 2010). Furthermore, there is evidence that relaxing

music can increase the recovery rate for stress response. In a study conducted by researchers Rebecca L. Garcia and Christopher J. Hand, "Analgesic effects of self-chosen music type on cold pressor-induced pain: Motivating vs. relaxing music", they concluded that actively listening to preferred music reduced "pain unpleasantness" ratings (2016). Stated in the discussion section of the research paper by them:

> "This [study] suggests listening to preferred relaxing music decreases physiological and psychological arousal levels, lowering the perception of pain unpleasantness. This may correspond to reduced negative affect, pleasant valence, and reduced tension achieved through mood and memory induction while listening to preferred relaxing music, optimizing pain reduction which may be abetted through active music listening."

The potential applications of music as a treatment for pain are clearly promising and necessitate further inquiry.

2.3 Streaming Music Services

Today, mobile phones are one of the most commonly-used devices which people use to listen to music. About 77% of adults in the United States own smartphones, which possess the capability of music playback either through pre-installed applications or third-party services (Pew Research Center, 2017). The popularity of streaming music services in particular is very high. The two entities comprising the largest market shares, Spotify and Apple Music, collectively capture nearly 90 million subscribers collectively as of the summer of 2017 (McIntyre, 2017).

While Apple Music has a slightly larger catalog of songs, with 40 million to Spotify's 30 million, Spotify's customer base is decidedly larger than that of its competitors, with over 60 million subscriber accounts (versus Apple's 27 million), and 140 million active users utilizing them. Spotify also offers support for developers, such as Android and iOS software development kits, a web application programming interface (API), and multiple varieties of HTML widgets. The web API allows applications to query data from the Spotify music catalog, as well as access user-related information such as playlists and saved music, and allows users to log in with their Spotify account credentials. Using a wrapper (a thin layer of code which translates an existing interface into a compatible one) specifically-created for the Spotify API and Android, this API can effectively be treated as an interface with Java methods that can be called to communicate with the API endpoints from within an Android application. It can then use this data to play and obtain information about specific music tracks from the Spotify library on the mobile device (Spotify, 2017).

2.4 Comparative Analysis of Similar Projects

There are already multiple projects which are related to the study-and/or-application of music as a tool to assist in the treatment of pain. These include software for both desktop and mobile operating systems and take a variety of approaches in how music is used. These endeavors, while showing initial promise, do not appear to have been supported very long or to have produced any meaningful or significant conclusions.

2.4.1 Heartbeat App

The Heartbeat App was a desktop application designed by University of Pennsylvania medical student and software developer Tim Soo. It was demonstrated at "Music Hack Day NYC" in 2010, alongside other novel music-related programs. The basis of the program was that it monitored the user's heart rate using external hardware sensors and would then dynamically create a playlist according to that data. It would begin by selecting music that matches the user's current heart rate and then transition to songs with slower tempos or lower beats per minute (BPM) ratings. This had the effect of gradually reducing the user's heart rate as it slowed to coincide with that of each subsequent track. It is worth noting that the developer of this project shares the belief that familiar music or that of a familiar genre or style is more effective.

The Heartbeat App used the API from the Echo Nest along with the music and multimedia-oriented visual programming language Max. As of March of 2014, the Echo Nest was acquired by Spotify and its functionally merged with the Spotify Web API over the following years (The Echo Nest, 2014). The music played by the Heartbeat App is sourced by accessing videos on YouTube, assembling them in a custom playlist, and then playing that playlist (Parker, 2011).

2.4.2 The ACPA eMTCP Music App

The ACPA is the American Chronic Pain Association, which is an organization with the mission of providing peer support and education in pain management skills to sufferers, family and friends, and healthcare professionals. As part of the Melomics research project, a mobile application was developed in cooperation with the University of Malaga that was designed to perform a similar task in experimenting with music as a treatment for chronic pain and was

partially funded by the Health Program of the Spanish Ministry of Economy and Innovation. As part of a study launched in 2013, the "empathetic music clinical trial on chronic pain" app as it was called, is described as allowing users to self-report their pain while listening to music. It is stated that the application would use the smartphone device to estimate the user's physiological state and then deliver music that would improve it, as a form of treatment known as empathetic music therapy (eMTCP).

      Empathetic music therapy employs an "artificial musician" which uses artificial intelligence algorithms processed on a cloud to stream music according to how the user is feeling. The content is created by music psychologists to be relaxing or distracting as appropriate and is presumably tailored in some way to fit the preferences of most users (the eMTCP App, 2017).

3. Methodology

Currently, there are two major software development paradigms in use; the traditional software development life cycle that has been in use for several decades, and the more modern approach of agile methodologies. This section discusses the details and differences of each, how they apply to our project and team composition, and the justification for our choice.

3.1 Traditional Software Development Life Cycle

The Software Development Life Cycle is an extension of the systems development life cycle (also referred to as SDLC), which is considered the oldest formalized methodology framework for building information systems. The SDLC comprises a set of well-defined, sequential stages which can be followed in order to design and develop a software product efficiently. These stages are intended to be followed one after the other in a specific order. For this reason, this methodology is sometimes also called the "waterfall model". While there are sometimes slight variations to the organization or naming conventions for the stages, they are more or less functionally identical.

Waterfall method, http://www.umsl.edu/~hugheyd/is6840/waterfall.html
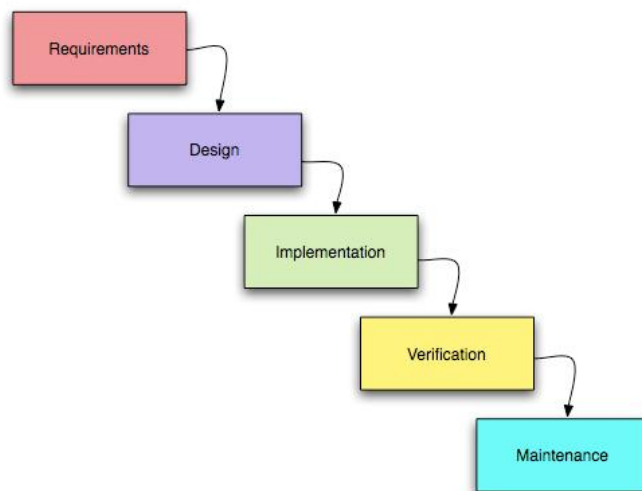
3.1.1 Stages of SDLC

First Stage: Planning and Requirement Analysis

The first stage of the process and most fundamental, this stage consists of conducting domain research and gathering input from industry experts, the customer, or market surveys in order to gain an understanding of the requirements of the product as well as their feasibilities.

Second Stage: Defining Requirements

The second stage consists of explicitly determining the product's requirements and confirming these with the party commissioning it. Each of the functional and nonfunctional requirements are clarified and communicated with the product stakeholders in such a way that they are mutually understandable. This is often presented as an SRS (Software Requirement Specification) document.

Third Stage: Design

This stage is where the architecture of the software is established, which may include a combination of models, diagrams, data dictionaries, or pseudo-code. Often more than one approach is proposed and considered, with the final design being chosen based on how well it is able to meet the criteria set forth by the requirements as well as the budget and time constraints of the project.

Fourth Stage: Building

This is the stage in which the actual coding and programming is performed in whichever language or environment is most suitable for the project.

<u>Fifth Stage: Testing</u>

While testing is expected to be performed regularly throughout the entirety of the project,

it must be completed before the project can move on to the final stage. This stage can be

dedicated strictly to repeated testing and fixing of components until the product meets the

expected quality standards as initially defined. The more intensive testing performed in this

phase is typically delegated to Quality Assurance engineers.

<u>Sixth Stage: Deployment and Maintenance</u>

This final stage is where the final product is delivered to the customer, its operation

initiated, and its performance verified in the real world and in its intended applications. Any

unexpected or undiscovered bugs or other problems are addressed and remedied where possible.

3.2 Agile Methodologies

"We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:
Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan
That is, while there is value in the items on
the right, we value the items on the left more."
        -Manifesto for Agile Software Development, http://agilemanifesto.org/

The Agile methodologies are a group of software design methodologies following the

Agile Manifesto. They are intended to break away from some of the inefficiencies of the rather

rigid organization of the standard SDLC. Agile methodologies are intended to be more flexible,

involve more regular communication with stakeholders, and avoid over-complication, such as by

eschewing excessive documentation requirements.

Within the Agile methodologies, the design process allows for and encourages revisions and refinement of requirements throughout the development of the product.

3.2.1 Scrum

Scrum is a specific application of Agile methodologies that is formulated to be a lightweight process framework, meaning that the overhead management and logistics are kept at a minimum in order to maximize the time available to do actual productive work. It is the most widely-used implementation of the Agile methodologies. Scrum revolves around repeated short, fixed-length iterations which are called "sprints".

During each sprint, a subset of the product backlog (prioritized list of features yet to be implemented) is selected to form the sprint backlog, the completion of which is the specific goal of that iteration period. While the sprint is in progress, team members meet daily or at some other frequent interval in order to keep track of what tasks are being actively worked on, who is working on them, and if there are any issues that need assistance from other members.
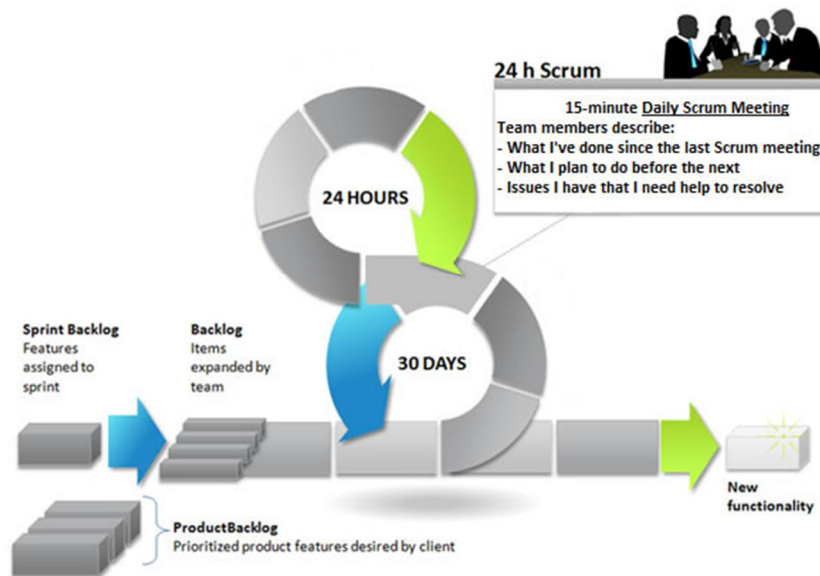
Scrum Process (cPrime, 2017)

4. Software Development Environment

In order to successfully develop software, the development environment should be one open to communication. This fact is extremely important when the software is being developed by more than one person. To maintain a truly team-developed product, the first step is communication. Academic email addresses were used primarily to facilitate correspondence between the team members and the project advisors in between in-person weekly meetings. For the purpose of communication between the team members, a workspace was created with the application Slack, which is a messenger that "unifies your entire team's communication, making your workflow, well, flow a lot better" (Slack). It allows instant messaging, with a scheme that feels more casual, across desktop and mobile platforms. The workspace contains the messages in a specific thread dedicated to the project.

The project's software development environment is composed of carefully selected technological tools that will aid toward the complete project. Most of these tools were chosen based on suggestion and further research into comparative options. Tools include basic email along with the subsections of this chapter.

4.1 Project Management Software

To stay on top of a time sensitive project, it is useful to employ project management software that condenses all the tasks that need to be complete, are in progress, or are completely finished. Often, team members names are assigned to a task to increase accountability and flexibility of those working on tasks. Trello is a tool that does just that with a highly customizable visual interface (PC Mag Digital Group, 2017). Asana is a comparative project management tool, advertising to "organize everything you need to keep track of individual and team tasks" (Task Management Software). While both the authors' have experience with one of

the mentioned project managers, and they both are a free web-based tool, Trello was selected as the project management software due to its upfront graphics and ease to add different software development stages.
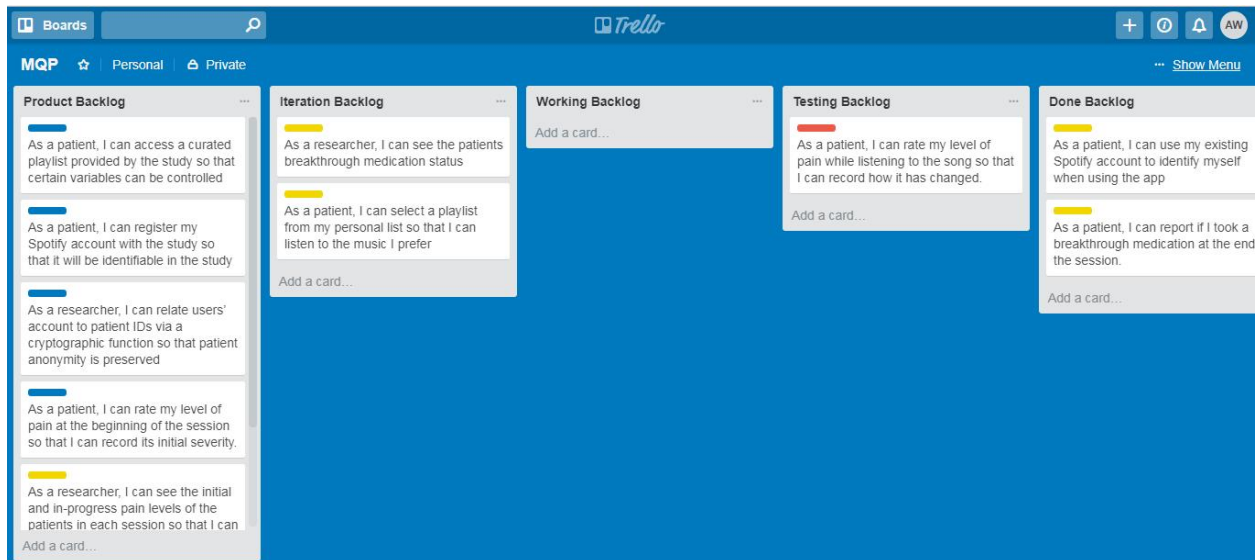
Screenshot of project (MQP) Trello 'board'

4.2 Repositories

When completing a project with a team the project itself must be easily accessible to all members. The following tools, Canvas, Google Drive, and GitHub, are addressed in this section and made the project easy access for all parties involved.

4.2.1 Canvas

Canvas, a learning management system, was used due to a request from the project advisor (Try Canvas, 2017). Documents that were more finalized, along with any pertinent information were uploaded to the projects canvas page, which 'enrolled' everyone connected to the project.

4.2.2 Google Drive

Google Drive was utilized in maintaining shared work-in-progress written documents and as a storage for any related project information. Google Drive was favored for this kind of storage due to its versatility (spreadsheets, docs, ease of uploading most file types), cloud-based sharing, saving, and editing attributes, and status as a popular platform. Storage capacity is 15GB but can be expanded with a monthly payment plan (Official Google Blog, 2014).

4.2.3 GitHub

To host the collaborative code, GitHub was the selected tool. GitHub promotes learning by sharing capabilities, "from open source to business", and offering free hosting for work to build software, ability to review code, and preference for managing projects (GitHub, 2017). It held a preference due to past experience utilizing it, and its popularity of use. The IDE used by the project can be configured to use Git, "a free and open source distributed version control system", along with a specified GitHub repository to facilitate version control for the application (Git). The official website for Git states that it was "designed to handle everything from small to very large projects with speed and efficiency" (Git).

4.3 Database

The application will need to be able to record data collected during the session in a way that is accessible to the researchers. For each session, this includes the email address identifying the participant, the songs listened to and their audio features, the pain levels indicated during the study and the time they were reported, and information about whether the participant took any medication. The advisors for this project suggested we consider MySQL to serve this role.

MySQL is an open-source relational database management system (RDBMS). Due to its widespread popularity and being open-source, it is easy to find tutorials and other assistance online to assist in its implementation. However, Android Studio has a built in SQLite library which will serve the purpose of saving a local copy of this data on the device. Furthermore, this data will then be copied to a central MySQL server database hosted on a virtual machine on the WPI network[1]. This database will contain all data collected by the app and serve as a comprehensive source for access by administrators of the study.

4.4 Integrated Development Environment

Since the application being developed is to be run on Android devices, the IDE of choice would be Android Studio.

4.4.1 Android Framework

Android Studio is the first-party, official IDE for Android. It is built upon the platform of JetBrains's IntelliJ IDEA software. The Android Software Development Kit is a library which provides tools and resources for creating user interface elements, with which the user interacts with the application through programs known as "activities". Android Studio also incorporates a virtual device for testing and debugging programs. Gradle is an open-source build automation system that is integrated within Android Studio and is used to build projects to form the Android application package (Android Application Kit/APK).

---

[1] configured with the assistance of Ermal Toto, a PhD student in the Computer Science department

4.5 Language

The executable code of Android applications is written in the Java programming language. Layouts, widgets, and other UI components are configured using Extensible Markup Language (XML). XML was designed to be self-descriptive, and to store and transport data (World Wide Web Consortium). To establish the connection to the central MySQL database, a PHP script is accessed on the WPI server. "PHP is a popular general-purpose scripting language that is especially suited to web development" (The PHP Group, 2018).

5. Software Requirements

The software requirements were heavily influenced by the provided research proposal draft from Professor Manzo prior to the commissioning of the MQP. This document detailed the ideal checklist for what the sponsors envisioned from the IMAGS-App, and later discussion with the project advisors during weekly meetings clarified, refined, and expanded upon this. The following sections describe the IMAGS-App requirements.

5.1 Functional and Nonfunctional Requirements

A classification system, by Robert Grady at Hewlett-Packard, for requirements goes by the acronym *FURPS+*; "Functionality, usability, reliability, performance, supportability, plus design requirements, implementation requirements, interface requirements and physical requirements" (Eeles, 2005).

5.1.1 Functional Requirements

The research proposal representing part of the project detailed what the 'IMAGS-App' specification needs were according to the researchers. Assuming that the application is run on an Android device, the following requirements must be met:

- Randomly assign patients to a listening/non-listening response
- Allow listening group to play music on their phone.
- Provide timed interface for patients to continually report their pain level through a touch sensor on the phone service
- Prompt patients to report if they did or did not take breakthrough medication at the end of the session.

- Report the following data to secure servers

  - Patient ID

  - Date and time of measurement

  - Music File ID

  - CRDI Pain Data

  - Breakthrough medication status

- Provide secure means of transmitting and storing the data

The finalized functional requirements fulfilled the core requests of the researchers:

- Timed interface for patients to continually report their pain level through an Android page on the application that permits them to register their pain.

- Allow the patients to play selected music.

- Prompt patients to report if they did or did not take breakthrough medication at the end of the session.

- Prompt patients to report an initial pain level at the start of the session and a final pain level at the end of the session.

- Provide a secure means of transmitting and storing the application data.

- Report the following data to secure servers:

  - Patient ID

  - Date and time of pain measurements

  - Music file ID and audio features

  - Breakthrough medication status

  - CRDI pain data

The goal of the project was to create an internet-connected Android application for use by both participants and IMAGS administrators. The application would be designed to instruct the participants in its use. During each listening session, the application would also collect pertinent information regarding the participants' song selection(s), as well as (1) the pain level experienced by the participant, (2) the timestamp of each recorded pain rating, and (3) the name and dosage of any prescribed medication taken at the end of the listening session. The researchers and administrators should be able to view the necessary information from the application database.

5.1.2 Nonfunctional Requirements

Security: Our application will need to securely handle any processes which uniquely identify participants. As this study will involve patients with certain medical conditions, it is within our interest to prevent unauthorized access to collected data. To facilitate logging in using the participant's Spotify account credentials, we will be using a token-based authorization system as part of the Spotify API which maintains secrecy of password information. All data collected is stored in protected locations on the Android device and then copied to a password-secured server on the WPI network.

Scalability: With the overarching IMAGS study still being developed, the scale of the project is yet undetermined. Therefore, we developed the application so that individual organizations could easily configure it to communicate with their own designated servers for data collection, and so eliminate the overhead of handling increased traffic loads or distributing that data to multiple destinations.

Usability: The participants of this study will be expected to use it on a mobile device, though their display size, connectivity, and even operating system versions will surely vary. Furthermore, there will be a wide array of different types of users with a range of familiarities with computing. Lastly, the participants will be assumed to in some way suffer from chronic pain, and so their dexterity may be impacted. We will design the interfaces of the application with consideration for how to best fit these aspects.

Performance: As this application involves collecting time-sensitive data and a degree of interactivity in the administration of the study, it is important that it be able to continue functioning despite any minor fluctuations in connectivity that may arise as is typical with mobile devices communicating with the internet.

5.2 Epics

Our requirements analysis incorporates the following user roles:

● Patient: persons suffering from chronic pain that is a participant in the study

● Researcher: personnel administering and conducting the research for the study

1. As a patient, I can log in with either my own or a provided account when using the app

2. As a patient, I can listen to music so that I can possibly alleviate pain

3. As a patient, I can rate my level of pain so that the study can track it

4. As a researcher, I can access a database so that I can view a variety of data pertaining to the experiment

5. As a researcher, I can register the Spotify account for a user so that they can be identified when using the app

5.3 User Stories

- As a patient, I can use the Android application so that I can participate in the study

  - As a patient, I can create a profile so that I can log in with it when using the app.

    - As a patient, I can log in with a profile so that I can identify myself when using the app.

    - As a patient, I can register my Spotify account with the study so that it will be identifiable in the study

    - As a patient, I can use my existing Spotify account to identify myself when using the app

  - As a patient, I can listen to music so that I can possibly alleviate pain.

    - As a patient, I can use the Spotify API so that I can listen to music.

      - As a patient, I can select a playlist from my personal list so that I can listen to the music I prefer

      - As a patient, I can access a curated playlist provided by the study so that certain variables can be controlled

  - As a patient, I can rate my level of pain so that the study can track it.

    - As a patient, I can use a slider to rate my pain on a scale so that it can be compared to other instances during my participation in the study

      - As a patient, I can rate my level of pain at the beginning of the session so that I can record its initial severity.

      - As a patient, I can rate my level of pain while listening to the song so that I can record how it has changed.

- ○ As a patient, I can report my breakthrough medication status.
  - ■ As a patient, I can report if I took a breakthrough medication at the end of the session so that researchers can consider its effects and interactions.

- As a researcher, I can employ the web application so that I can research the effects of music on chronic pain
  - ○ As a researcher, I can access a database so that I can view a variety of data pertaining to the experiment.
    - ■ As a researcher, I can see what songs were listened to in each session so that I can determine their respective effectivenesses.
    - ■ As a researcher, I can see the initial and in-progress pain levels of the patients in each session so that I can determine how they changed
    - ■ As a researcher, I can see the participant's breakthrough medication status so that I can consider its effects and interactions

5.4 Use Cases

Use cases serve as a methodology in system analysis. They follow a specific format detailing how a user would interact with a system. They are especially helpful in clarifying, organizing, and identifying system requirements. The following use cases model the final application:

Use Case: Login Patient

Participating Actor: Initiated by Patient

Entry Condition: The application welcome screen exists

Exit Criteria: Patient credentials are accepted and stored

Flow of Events:

      1. The Patient requests to proceed

      2. The System verifies and performs request

Alternate Flow of Events:

[Incorrect Credentials]

      1. The Patient enters invalid or incorrect credentials

      2. System's Spotify authorization window notifies the user and returns to previous state

Use Case: Start Session

Participating Actor: Initiated by Patient

Entry Condition: Login success screen exists and is present

Exit Criteria: System presents pain scale and prompts initial pain level rating

Flow of Events:

    1. The Patient requests to continue from current view

    2. System presents pain scale and prompts initial pain rating

Use Case: Rate Pain

Participating Actor: Initiated by Patient

Entry Condition: Patient scale is present

Exit Criteria: Patient rated pain information is stored (to database)

Flow of Events:

      1. The Patient requests to move pain level on scale

      2. The System presents the updated pain scale and saves the change to database

Use Case: Search Music

Participating Actor: Initiated by Patient

Entry Condition: Search activity screen present

Exit Criteria: Search activity screen presented with results of the search query

Flow of Events:

      1. The Patient inputs search criteria and requests to search

      2. The System presents the results of the search query

Use Case: Play Music

Participating Actor: Initiated by Patient

Entry Condition: Search activity screen present with some song data

Exit Criteria: Session screen with selected song playing

Flow of Events:

    1. The Patient requests to play music from search

    2. System plays music and presents currently playing info and pain scale (session screen)

Alternate Flow of Events:

[Pause Music]

1. The User requests to pause currently playing music

2. System pauses music

Use Case: Report Medication

Participating Actor: Initiated by User

Entry Condition: User has finished listening session and reached breakthrough medication activity screen

Exit Criteria: User completed medication activity and logout screen presented.

Flow of Events:

1. The User reports final pain record

2. The system presents the breakthrough medication activity screen

Use Case: Quit Application

Participating Actor: Initiated by User

Entry Condition: User has finished session and reached final activity screen

Exit Criteria: Applications closes

Flow of Events:

3. The User requests to logout of application

4. User is logged out of Spotify and System terminates

5.4.1 Use Case Diagram

IMAGS APP

Play Music

Pause Music

Search Music

Login Patient

Incorrect Credentials

USER

Start Session

Quit Application

Report Medication

Rate Pain
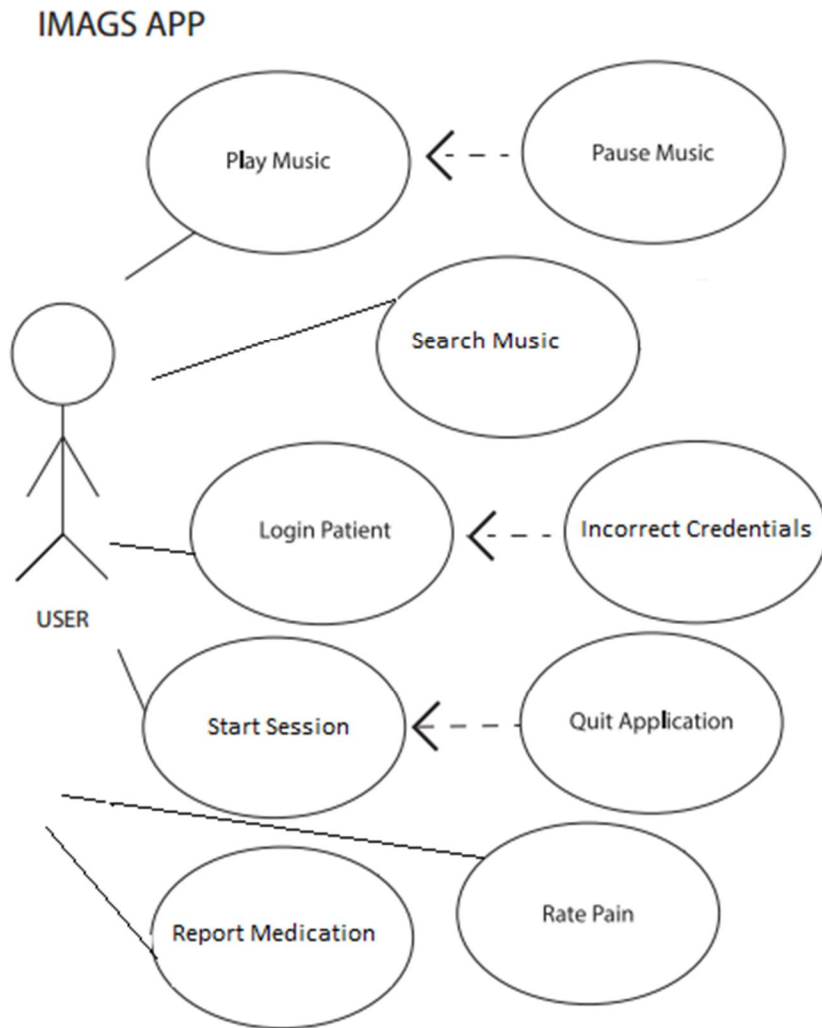
5.5 Interface Mockups

In the planning phase of the project, an important task was creating mockups of the user interface, including all of the screens the user would navigate through during the session. To accomplish this, the Adobe Illustrator software was used.

5.5.1 Welcome

Here, the user is informed that they will be using their Spotify account to login and must be registered in the study.
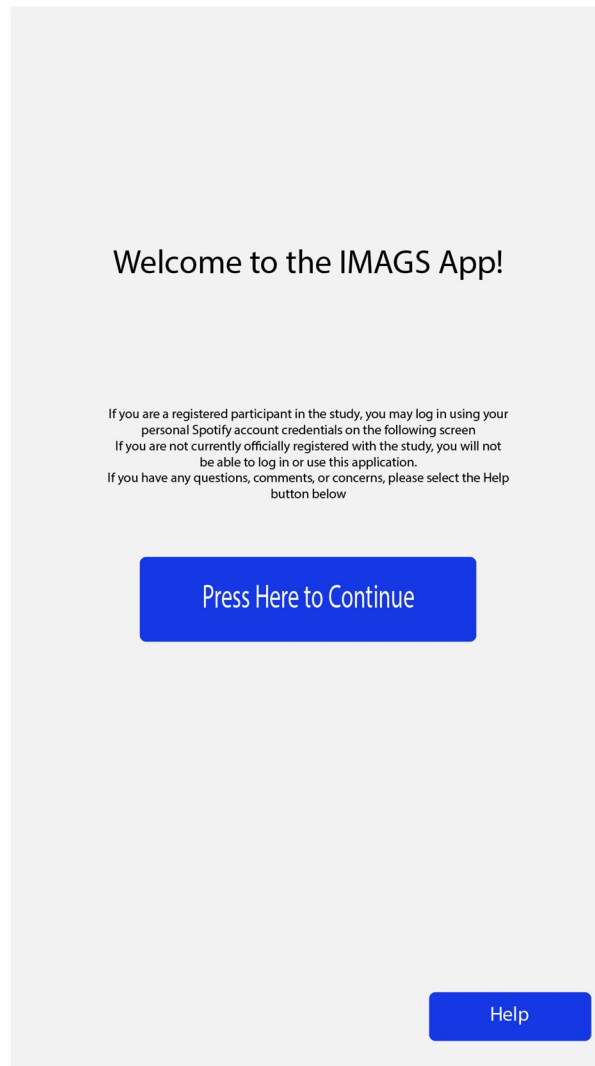
5.5.2 Spotify Sign In

The user is directed to a Spotify login page implementing the OAuth library, which opens a window for entering either their Spotify or associated Facebook account credentials.
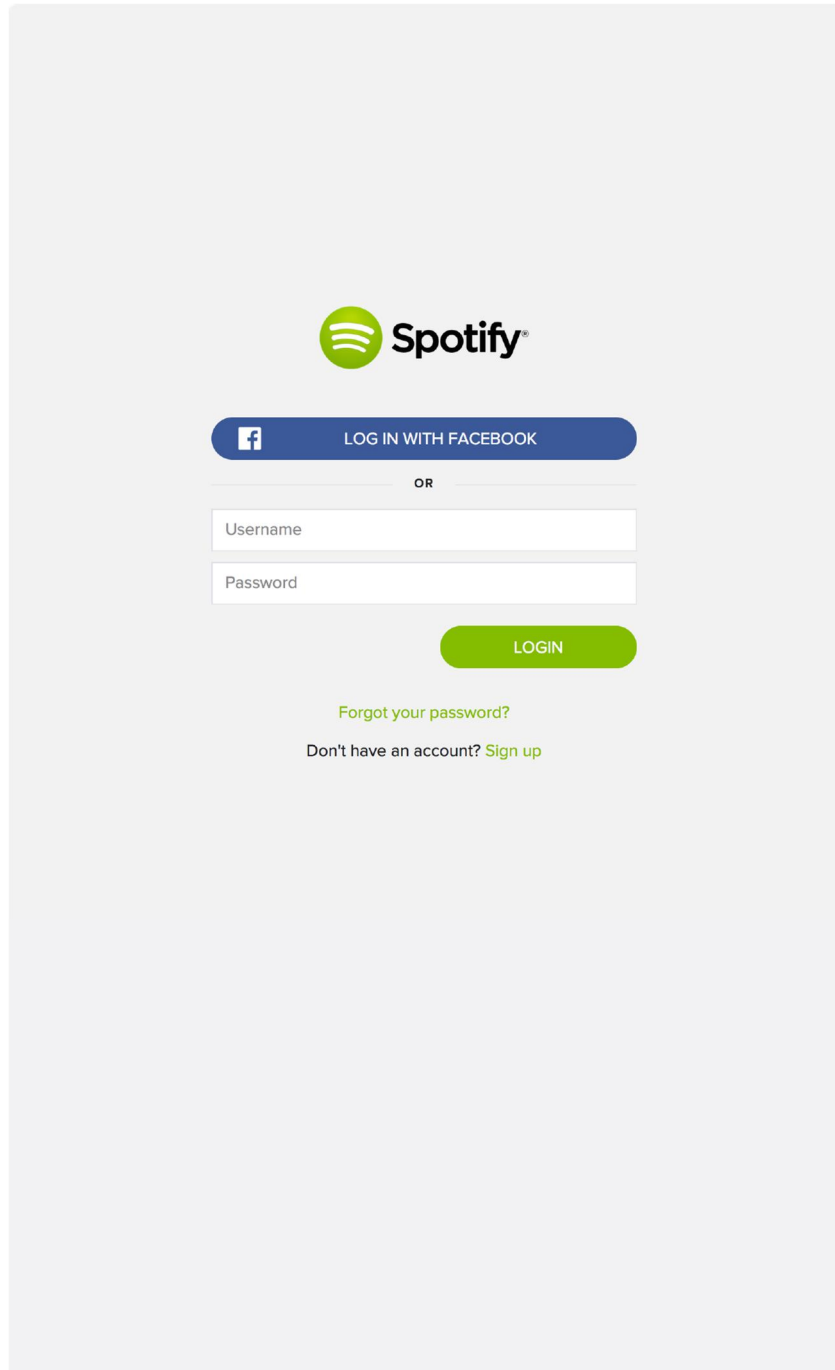
5.5.3 Pre-Session Screen

At this screen, the user is introduced to the pain-rating slider and asked to set it to the initial intensity before they continue.

5.5.4 Search Screen

Here, the user is prompted to enter a song title in the search bar. Results from the Spotify catalog are shown below, which the user selects from by tapping a result entry from among them and then tapping the button at the bottom.

Search for a Song

Enter a song title here...

Results:

Press Here to Continue

Help

10
Hurts
Worst

8
Hurts
Whole Lot

6
Hurts
Even More

4
Hurts
Little More

2
Hurts
Little Bit

0
No
Hurt
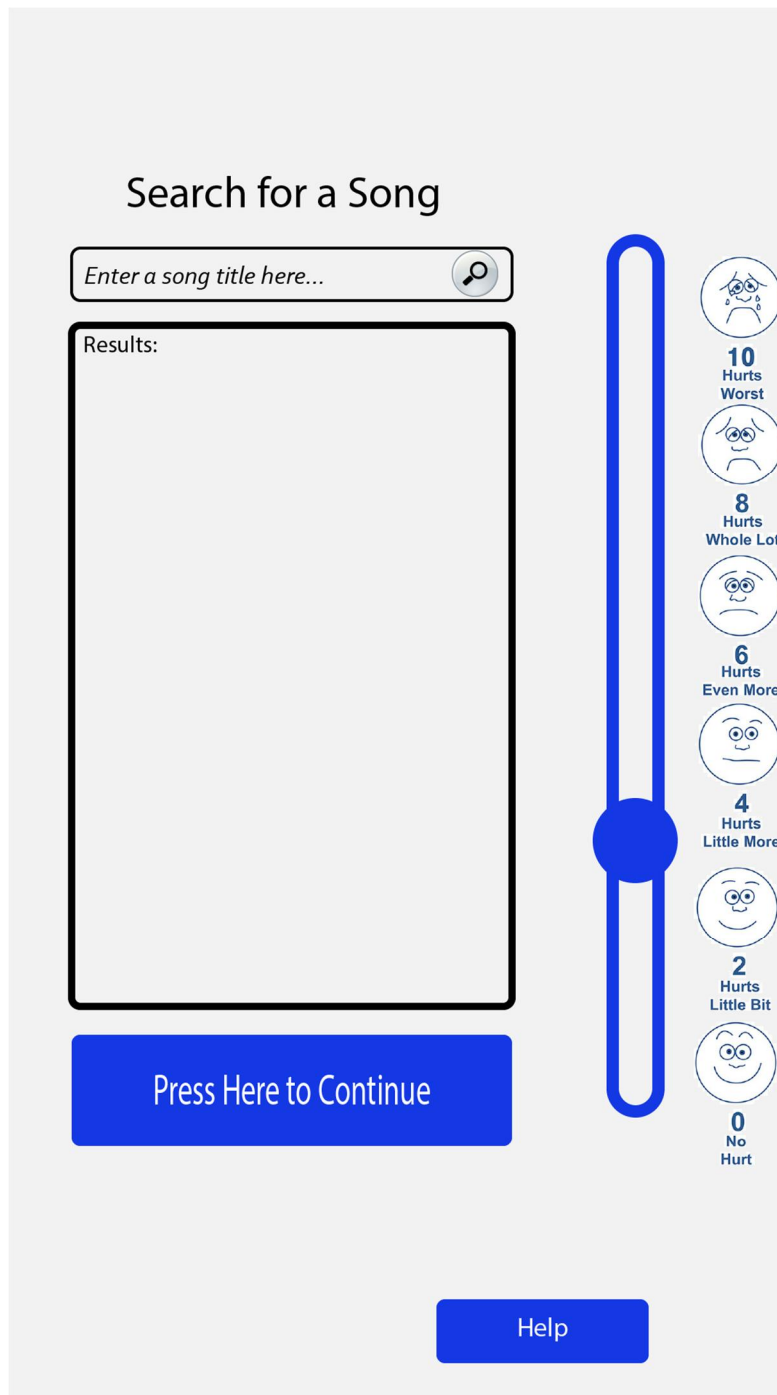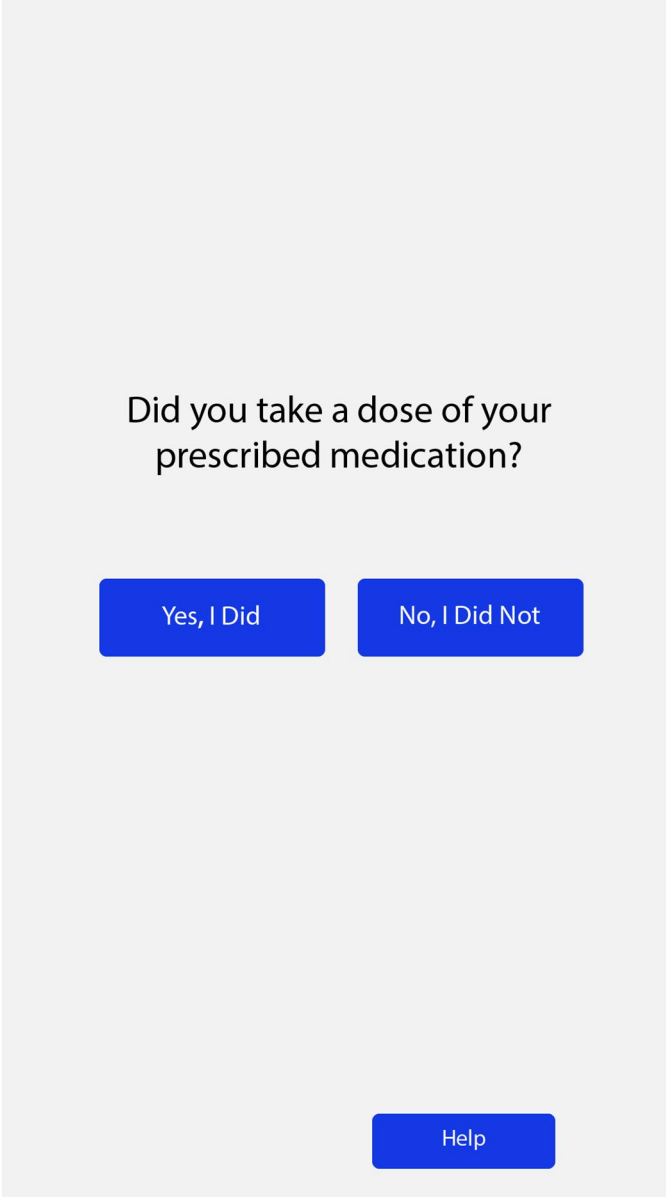
5.5.5 Session Screen

During the session, information about the current track is displayed along with the associated album artwork. There is a player interface with buttons to control playback. The PainTracker slider remains on-screen below so that the user can adjust it to match any changes in their level of pain experienced while listening.

5.5.6 Medication Screen

After they are finished listening to music, the user is asked at the end of the session whether they took their prescribed medication.

5.5.7 Post-Session Screen

Finally, the user is informed that the session is over and presented with a button which will log them out of their Spotify account.

Thank you for you for your
participation in the study!

Please be sure to press the
button below to log out of your
Spotify account

Log Out

Help

6. Design

6.1 Context Diagram
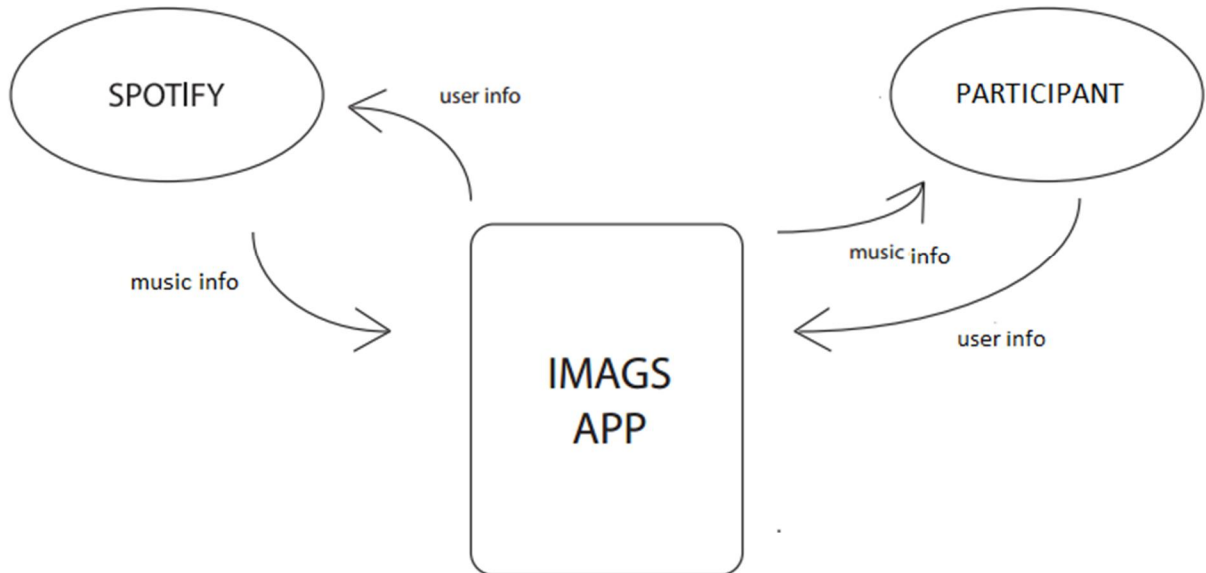
User info encompasses their Spotify Account information along with their assigned participant ID. The music info from Spotify to the IMAGS app is all of the attributes listed in the 'song' table in the database. The music info from the IMAGS app to the user is the title, artist(s), album, artwork, and the audio content.

6.2 Class Diagram

IMAGS

MainActivity
+mProceedButton: Button
+mHelpButton: Button
+onCreate(savedInstanceState: Bundle): void

LoginActivity
+CLIENT_ID: String
+REDIRECT_URI: String
+mTitle: TextView
+mText: TextView
+mContinueButton: Button
+mHelpButton: Button
+mToken: String
+mPID: String
+mSID: String
+onCreate(savedInstanceState: Bundle): void
+onActivityResult(requestCode: int, resultCode: int, intent: Intent): void
+whiteListCheck(): boolean
+processLogin(): void

PresessionActivity
+mContinueButton: Button
+mHelpButton: Button
+mToken: String
+mSID: String
+mSeekBar: SeekBar
+mPainTracker: PainTracker
+mFacesScale: ImageView
+onCreate(savedInstanceState: Bundle): void
+newIntent(context: Context, SID: String, token: String): Intent
+processTouch(xPos: float): void

SearchActivity
+mActionListener: Search.ActionListener
+mLayoutManager: LinearLayoutManager
+mScrollListener: ScrollListener
+mAdapter: SearchResultsAdapter
+mSID: String
+mToken: String
+newIntent(context: Context, token String): Intent
+onCreate(savedInstanceState: Bundle): void
+reset(): void
+addData(items: List<Track>): void
+onPause(): void
+onResume(): void
+onSaveInstanceState(outState: Bundle): void
+onDestroy(): void

Search

ResultsScrollListener
+mLayoutManager: LinearLayoutManager
+SCROLL_BUFFER: int
+mCurrentItemCount: int
+mAwaitingItems: boolean
+reset(): void
+onScrolled(recyclerView: RecyclerView, dx: int, dy: int): void

SearchPager
+mSpotifyApi: SpotifyService
+mCurrentOffset: int
+mPageSize: int
+mCurrentQuery: String
+getFirstPage(query: String, pageSize: int, listener: CompleteListener): void
+getNextPage(listener: CompleteListener): void
+getData(query: String, offset: int, limit: int, listener: CompleteListener): void

HelpDialogFragment
+onCreateDialog(savedInstanceState: Bundle): Dialog

MedicationActivity
+mYesButton: Button
+mNoButton: Button
+mSubmitButton: Button
+mText: TextView
+mEditText: EditText
+mSID: String
+mTookMed: boolean
+mName: String
+mDosage: String
+mQuestionNum: int
+onCreate(savedInstanceState: Bundle): void
+nextActivity(): void
+newIntent(context: Context): Intent

SessionActivity
+mPlayToggle: Button
+mFinishButton: Button
+mHelpButton: Button
+mNewSongButton: Button
+mPlayer: Player
+mSongDataManager: SongDataManager
+mTrackTitle: TextView
+mTrackArtist: TextView
+mTrackAlbum: TextView
+mTrackImage: ImageView
+mSeekBar: SeekBar
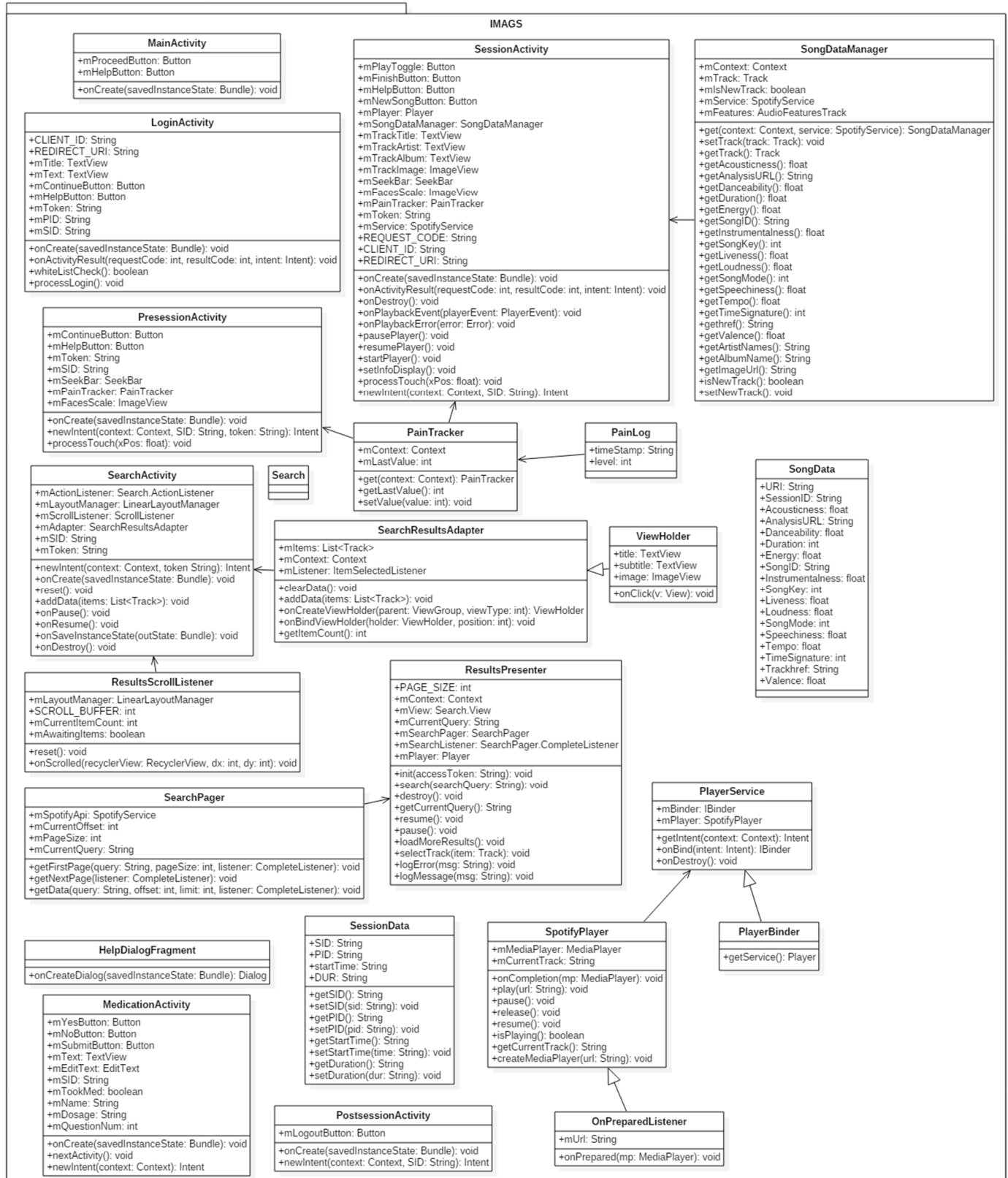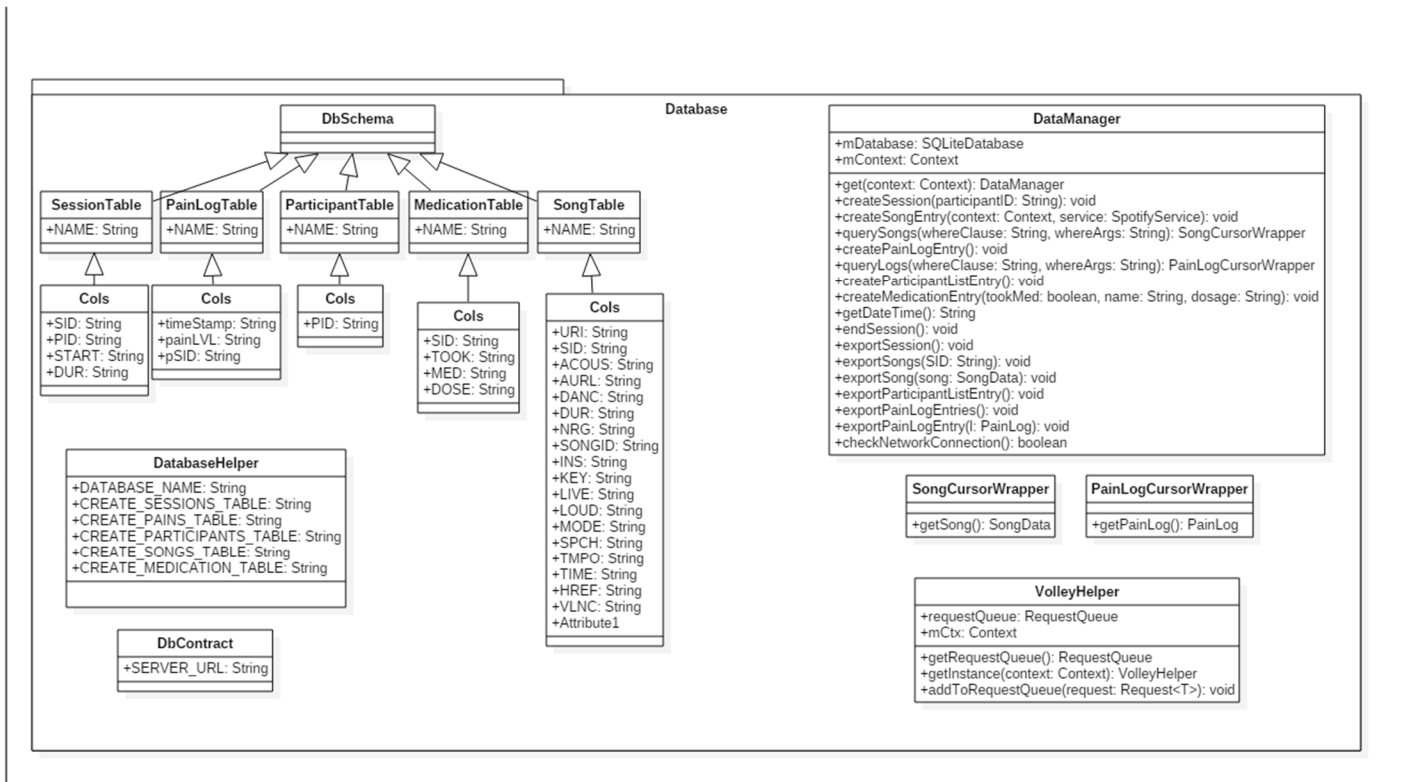+mFacesScale: ImageView
+mPainTracker: PainTracker
+mToken: String
+mService: SpotifyService
+REQUEST_CODE: String
+CLIENT_ID: String
+REDIRECT_URI: String
+onCreate(savedInstanceState: Bundle): void
+onActivityResult(requestCode: int, resultCode: int, intent: Intent): void
+onDestroy(): void
+onPlaybackEvent(playerEvent: PlayerEvent): void
+onPlaybackError(error: Error): void
+pausePlayer(): void
+resumePlayer(): void
+startPlayer(): void
+setInfoDisplay(): void
+processTouch(xPos: float): void
+newIntent(context: Context, SID. String): Intent

PainTracker
+mContext: Context
+mLastValue: int
+get(context: Context): PainTracker
+getLastValue(): int
+setValue(value: int): void

SearchResultsAdapter
+mItems: List<Track>
+mContext: Context
+mListener: ItemSelectedListener
+clearData(): void
+addData(items: List<Track>): void
+onCreateViewHolder(parent: ViewGroup, viewType: int): ViewHolder
+onBindViewHolder(holder: ViewHolder, position: int): void
+getItemCount(): int

ResultsPresenter
+PAGE_SIZE: int
+mContext: Context
+mView: Search.View
+mCurrentQuery: String
+mSearchPager: SearchPager
+mSearchListener: SearchPager.CompleteListener
+mPlayer: Player
+init(accessToken: String): void
+search(searchQuery: String): void
+destroy(): void
+getCurrentQuery(): String
+resume(): void
+pause(): void
+loadMoreResults(): void
+selectTrack(item: Track): void
+logError(msg: String): void
+logMessage(msg: String): void

SessionData
+SID: String
+PID: String
+startTime: String
+DUR: String
+getSID(): String
+setSID(sid: String): void
+getPID(): String
+setPID(pid: String): void
+getStartTime(): String
+setStartTime(time: String): void
+getDuration(): String
+setDuration(dur: String): void

PostsessionActivity
+mLogoutButton: Button
+onCreate(savedInstanceState: Bundle): void
+newIntent(context: Context, SID: String): Intent

PainLog
+timeStamp: String
+level: int

ViewHolder
+title: TextView
+subtitle: TextView
+image: ImageView
+onClick(v: View): void

SpotifyPlayer
+mMediaPlayer: MediaPlayer
+mCurrentTrack: String
+onCompletion(mp: MediaPlayer): void
+play(url: String): void
+pause(): void
+release(): void
+resume(): void
+isPlaying(): boolean
+getCurrentTrack(): String
+createMediaPlayer(url: String): void

OnPreparedListener
+mUrl: String
+onPrepared(mp: MediaPlayer): void

SongDataManager
+mContext: Context
+mTrack: Track
+mIsNewTrack: boolean
+mService: SpotifyService
+mFeatures: AudioFeaturesTrack
+get(context: Context, service: SpotifyService): SongDataManager
+setTrack(track: Track): void
+getTrack(): Track
+getAcousticness(): float
+getAnalysisURL(): String
+getDanceability(): float
+getDuration(): float
+getEnergy(): float
+getSongID(): String
+getInstrumentalness(): float
+getSongKey(): int
+getLiveness(): float
+getLoudness(): float
+getSongMode(): int
+getSpeechiness(): float
+getTempo(): float
+getTimeSignature(): int
+gethref(): String
+getValence(): float
+getArtistNames(): String
+getAlbumName(): String
+getImageUrl(): String
+isNewTrack(): boolean
+setNewTrack(): void

SongData
+URI: String
+SessionID: String
+Acousticness: float
+AnalysisURL: String
+Danceability: float
+Duration: int
+Energy: float
+SongID: String
+Instrumentalness: float
+SongKey: int
+Liveness: float
+Loudness: float
+SongMode: float
+Speechiness: float
+Tempo: float
+TimeSignature: int
+Trackhref: String
+Valence: float

PlayerService
+mBinder: IBinder
+mPlayer: SpotifyPlayer
+getIntent(context: Context): Intent
+onBind(intent: Intent): IBinder
+onDestroy(): void

PlayerBinder
+getService(): Player

Each of the Activity classes (MainActivity, LoginActivity, PresessionActivity, SearchActivity, SessionActivity, MedicationActivity, and PostsessionActivity) correspond with a specific screen with which the user interacts with the app and contain methods for setting the visual components of the user interface as well as the functionality of the widgets such as buttons and text entry fields.

There are multiple classes which are associated specifically with the search action in the application, which facilitate querying the Spotify library and allowing the user to view the results. These classes are comprised of the ResultsPresenter, SearchPager, SearchResultsAdapter, ViewHolder, and ResultsScrollListener.

Two classes are responsible for the manipulation of data pertaining to the participant's pain records, which are PainTracker and PainLog. The PainTracker is a singleton

implementation that handles all input from the SeekBar widget the user interacts with and creates PainLog objects which contain the data.

Data relating to the songs played from the Spotify library is handled by another singleton class, SongDataManager, which uses the Spotify API service to obtain the specific audio features pertaining to the track and record SongData objects for each song played during the session. Lastly, there are a handful of classes which are imported from the Spotify Player library, including SpotifyPlayer, PlayerService, PlayerBinder, and an OnPreparedListener, which are responsible for the audio playback of the selected tracks during the session.

In addition to the primary contents of the package, there is a Database sub-package. This contains: DbContract, which contains information pertaining to the network addresses for communication with the central server; the DbSchema class, which is composed of subclasses for each of the tables in the SQLite database and contains constants for the table structures and column titles; the DatabaseHelper class, which opens and creates the tables for the SQLite database; the DataManager class, which handles exporting all of the necessary data to the central server and the master MySQL database; Cursor wrappers which handle the conversion of database entries into the appropriate objects; and a VolleyHelper class which has methods that take care of managing and adding HTTP requests to the queue.

6.3 Entity Relationship Diagram
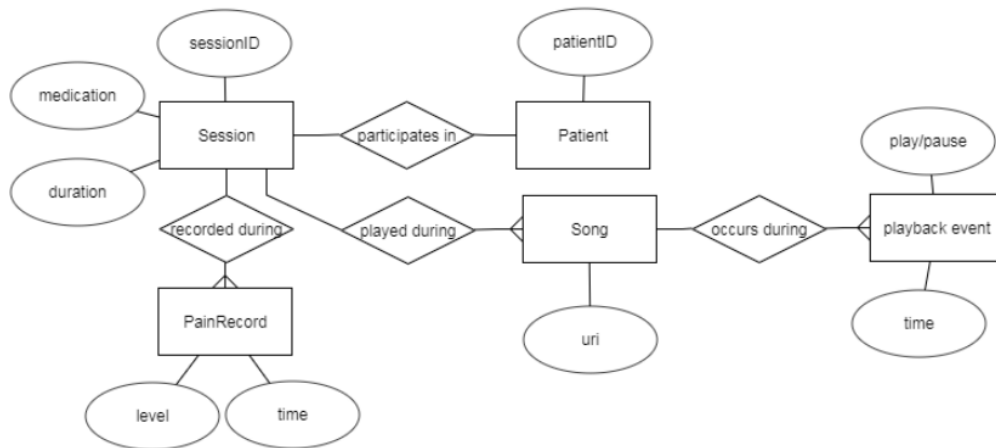
6.4 User Interface

6.4.1 Welcome
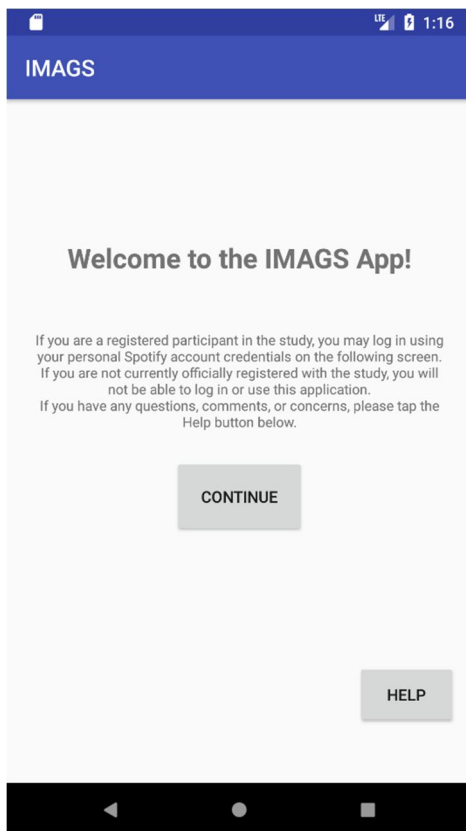
Here, the user is informed that they will be using their Spotify account to login and must be registered in the study.

6.4.2 Spotify Sign In

The user is directed to a Spotify login page implementing the OAuth library, which opens

a window for entering either their Spotify or associated Facebook account credentials.
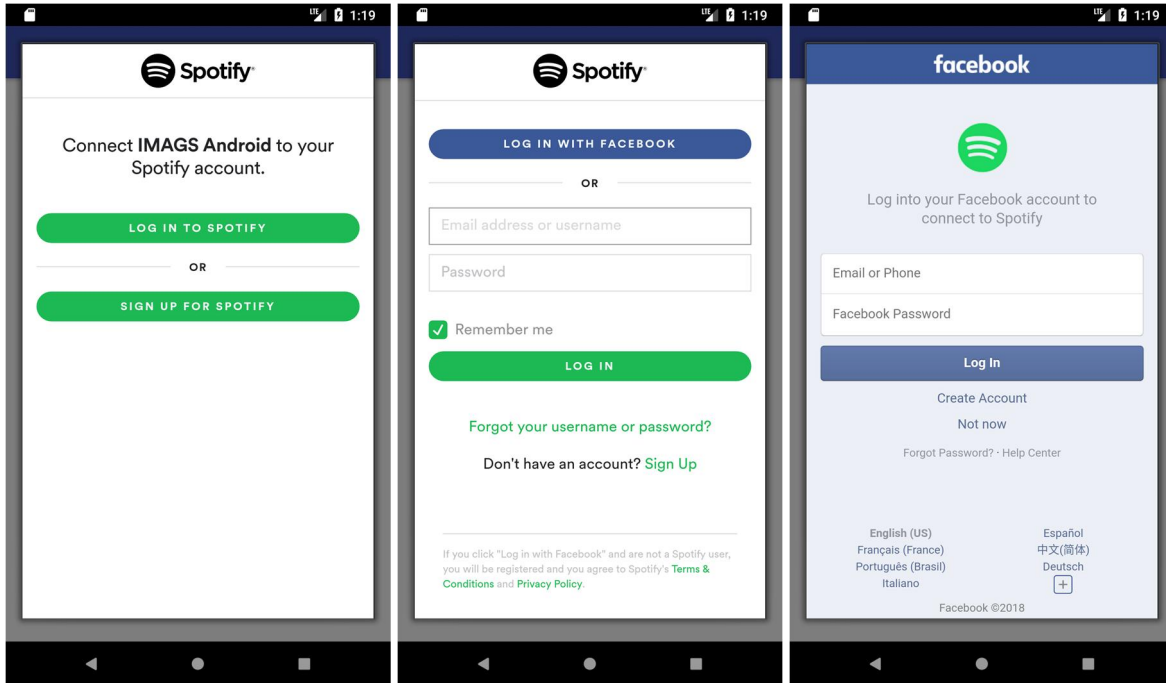
6.4.3 Pre-Session Screen

At this screen, the user is introduced to the pain-rating slider

and asked to set it to the initial intensity before they are able

to proceed.
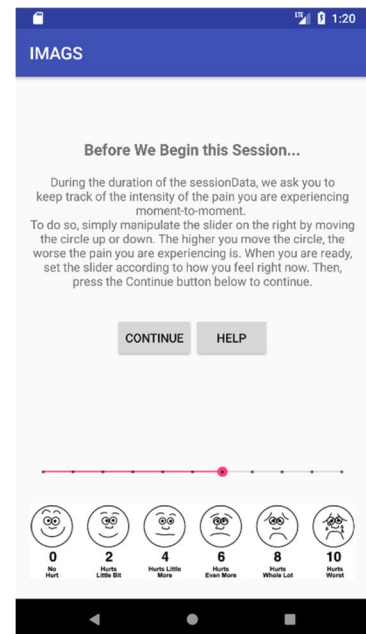
6.4.4 Search Screen

Here, the user is prompted to enter a search query in the search bar. Results from the Spotify catalog are shown below, which the user can scroll through by swiping the list and selects from by tapping a result entry from among them.
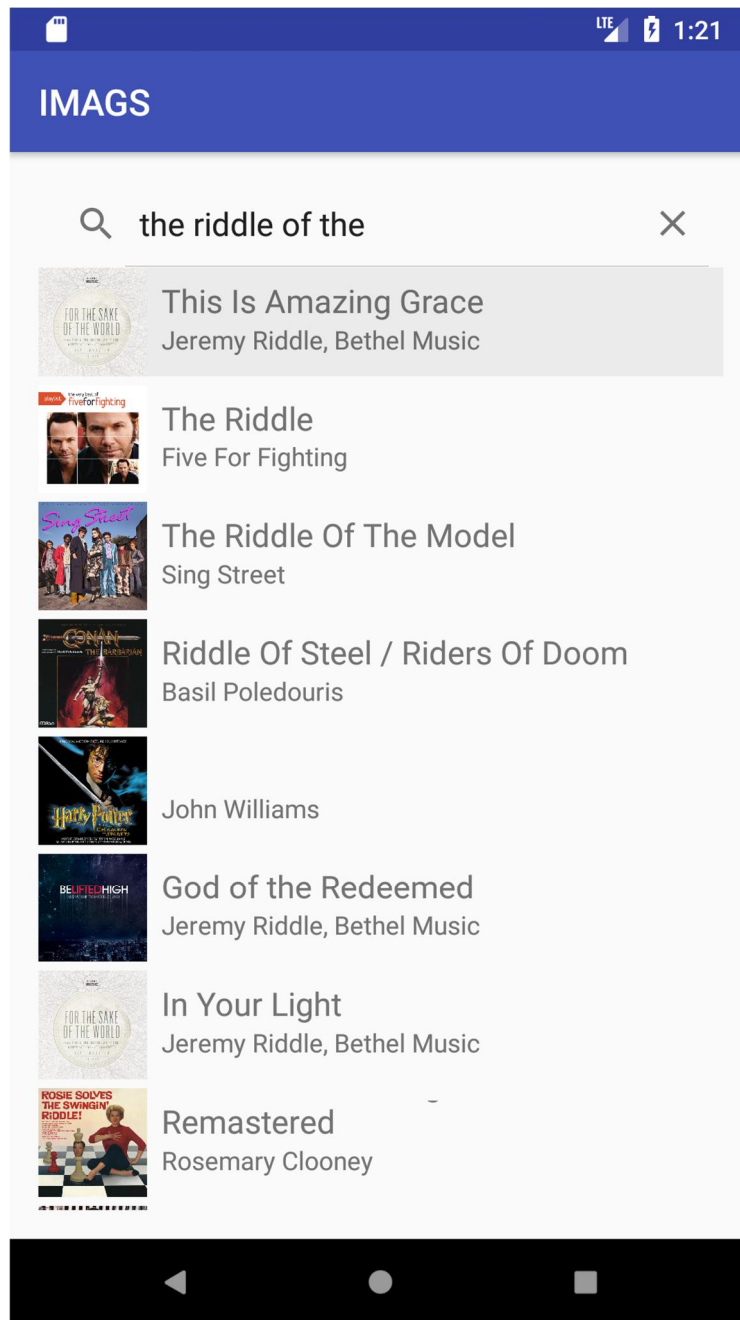
6.4.5 Session Screen

During the session, information about the current track is displayed be neath the associated album artwork. Above, buttons are available to control playback, select a new song, or end the session. The PainTracker slider remains on-screen below so that the user can adjust it to match any changes in their level of pain experienced while listening.
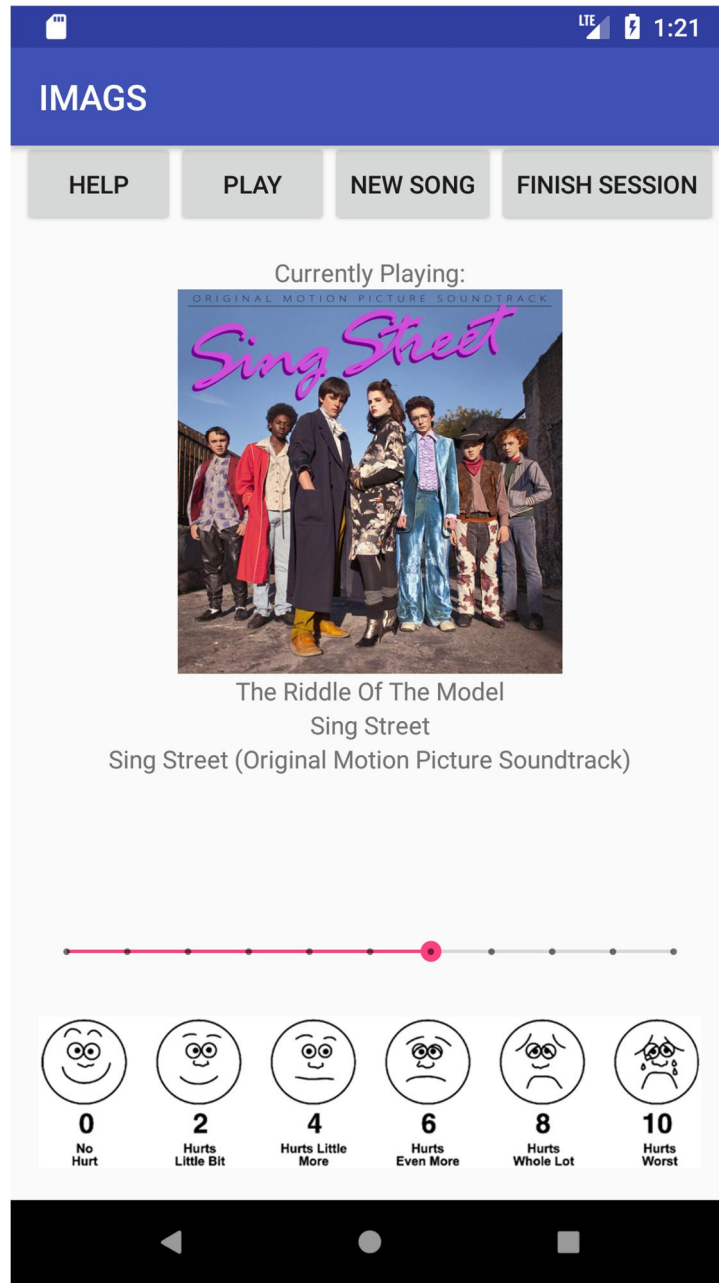
6.4.6 Medication Screen

After they are finished listening to music, the user is asked at the end of the session whether they took their prescribed medication. If they respond in the affirmative, two more simple questions are presented inquiring what the name and dosage of that medication are.
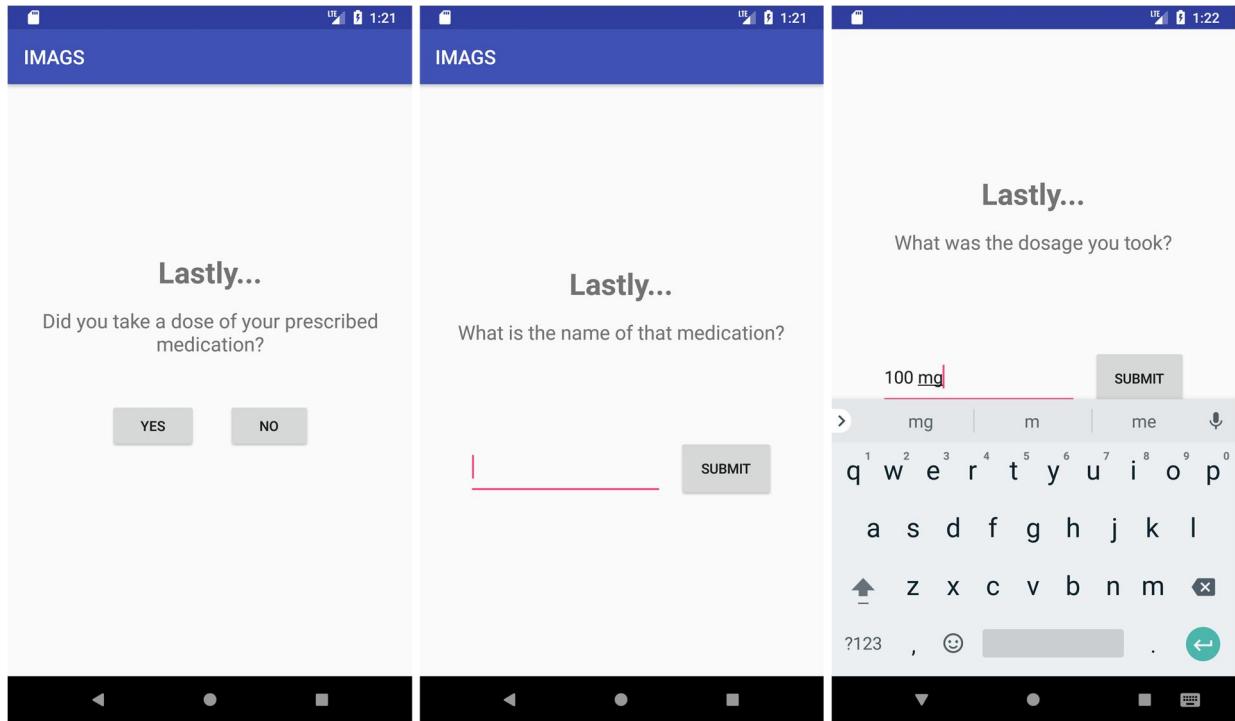
6.4.7 Post-Session Screen

Finally, the user is informed that the session is over and presented with a button which will log them out of their Spotify account.
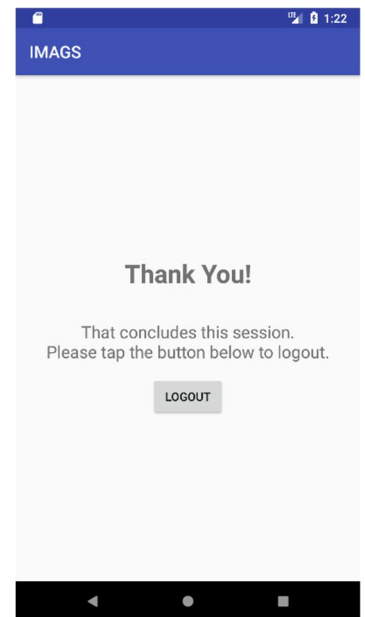
7. Implementation

7. 1 Development

Midway through our second term of the project, the development of our project encountered a major roadblock which necessitated significant deviation from our initial design, especially with regard to the development environment and programming language.

7.1.1 Programming Language

After production of the application was already underway, our team realized that the Spotify Player Widget which we intended to play the music on the mobile device did not behave as we had expected it to. As the widget does not actually play the music, and instead opened the separate Spotify application, this was found to be incompatible with our design and the intended user experience. Therefore, we elected to implement the application as its own standalone mobile app. This necessitated porting our design to the new programming language (Java) and development environment (Android Studio) and reconfiguring assets and layouts, such as creating xml resources in place of html files.

7.1.2 Database

The database component of this project was largely redefined over the course of its development. Though in some ways reduced in the features provided, the structure was expanded significantly. Ultimately, the database was divided into two units: a SQLite database created and stored on the participant's mobile device, and a central MySQL database hosted on a virtual machine on the WPI network.

7.1.3 Session Screen

Rather than limit the participant to listening to one song per session, the ability to return to the previous screen and search for a new song was added to the Session screen. This allows the participant to search for and play any number of songs during the session until they select the option to end the session.
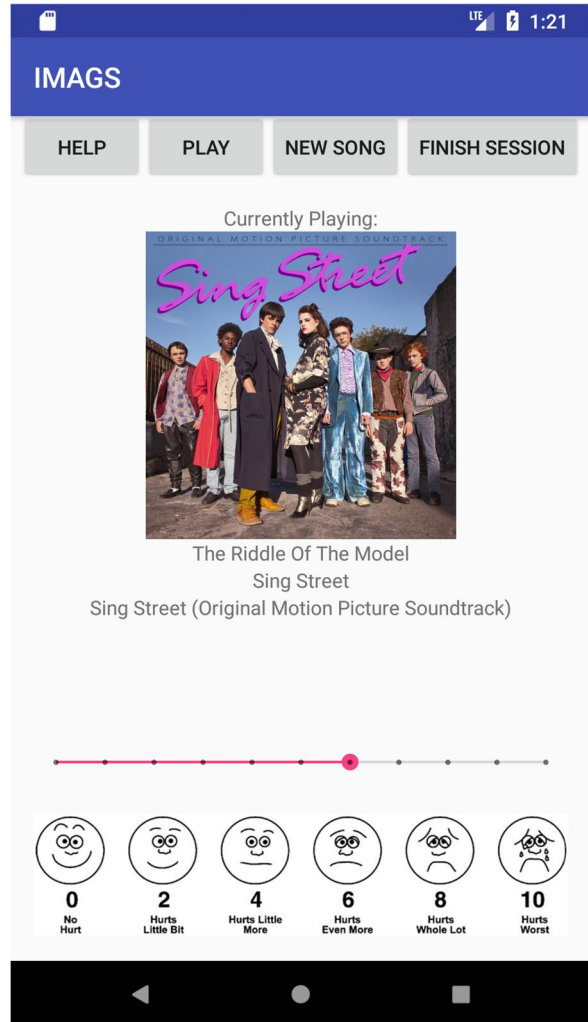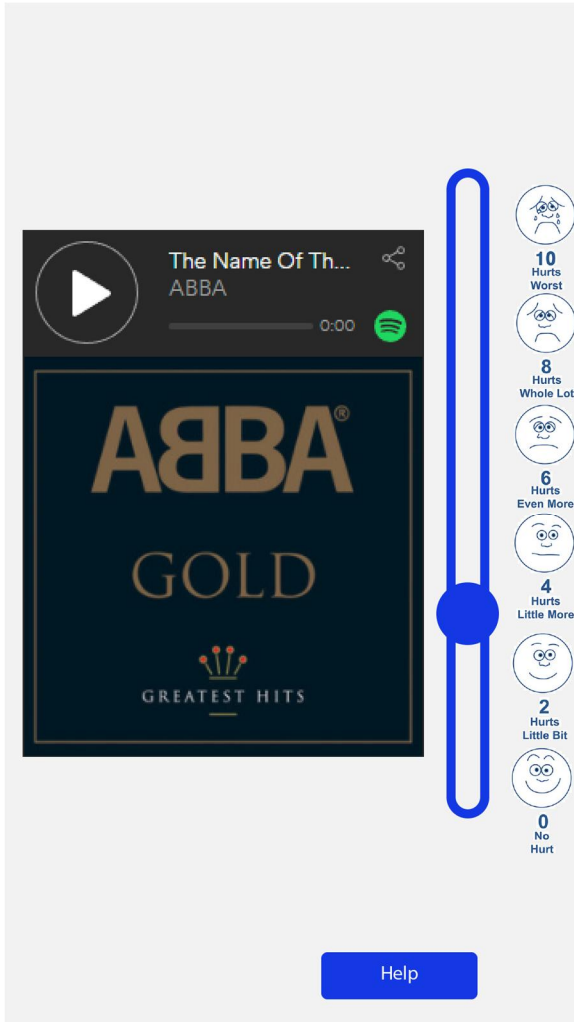
7.1.4 Search Screen

The buttons and PainTracker initially present on the Search screen were removed to allow the search results display to fill the entirety of the screen. This was done to improve readability of the results and to improve the ease of manipulating the results list, which requires the participant to touch and drag or swipe on the list. As the participant was deemed unlikely to experience any significant change in pain during the search step, the omission of the PainTracker was not seen to be disadvantageous here.
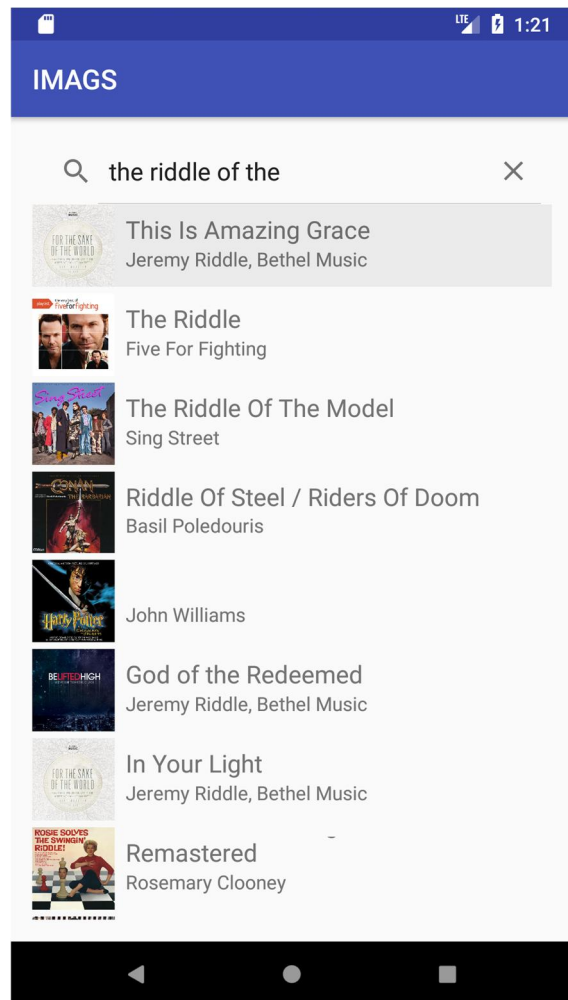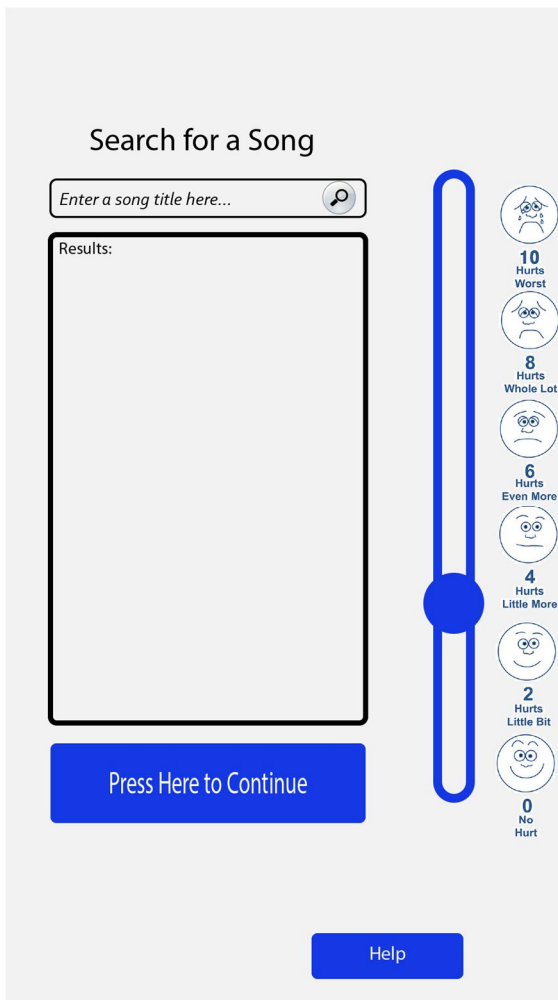
7.1.5 Medication Screen

Initially, the only data that was to be collected here was simply whether the participant had taken any of their prescribed medication or not. As any medication is expected to have a significant impact on the participant's overall pain and experience, it was decided that more information about this element could be very valuable. Therefore, we extended this section to include two very basic follow-up questions if the participant indicated that they had taken some medication. The participant is prompted to enter the name of the medication and then the dosage and are given a field in which they can enter text describing the answers in whatever form they deem appropriate.
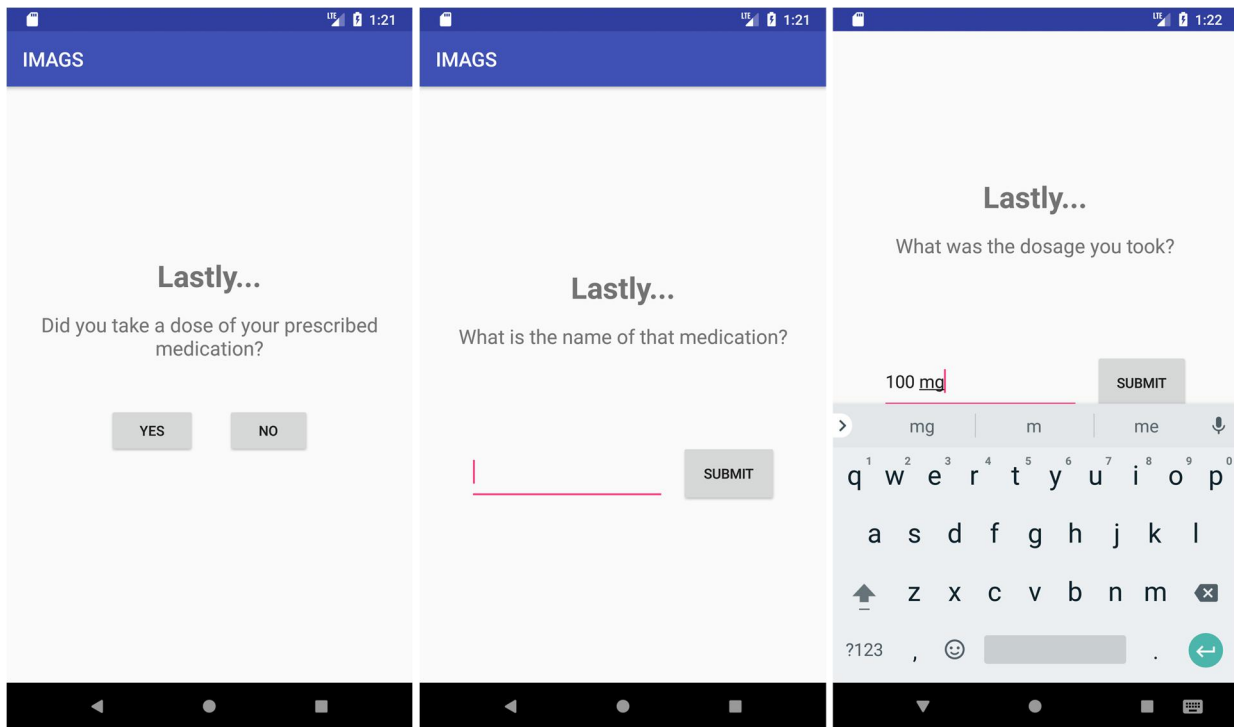
7.1.6 PainTracker Slider

Over the course of the application's development, the PainTracker slider widget which records the participant's pain level was modified in two ways. First, the orientation of the widget was changed from vertical to horizontal. This decision was made in consideration of ease of implementation as well as usability and aesthetics. The first-party widget included in Android Studio called the SeekBar was the most ideal for our purposes, but we found that attempting to modify it from its default horizontal orientation would be a very complicated and difficult process. Furthermore, in discussion between the team members and the advisors, it was decided that a test implementation using the horizontal version appeared much more appealing as it allowed for more adequate width for the other content on the screen and was less visually confusing when placed on its own at the bottom of the screen.

Second, the Pain Faces graphic positioned alongside the slider had functionality added to respond to touch events. This would allow participants to tap a location among the faces to have the thumb on the slider jump to the corresponding horizontal position, which may be easier for some participants such as those with poor dexterity.
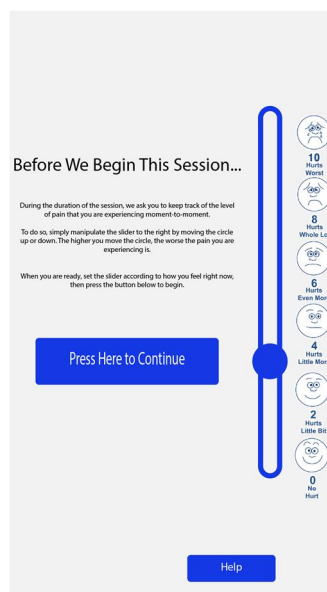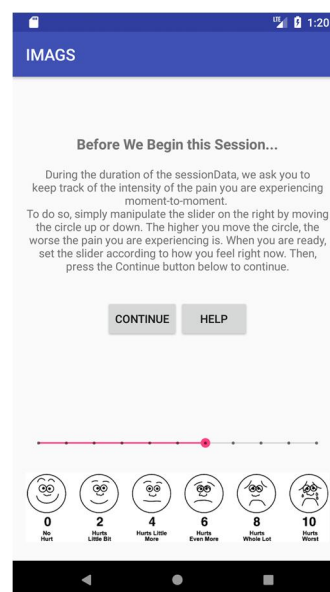
Original Design   Final Design

8. Assessment

8. 1 Requirements

8.1.1 Functional

Referring back to the Software Development subsection (5.1.1), we successfully achieved these functional software requirements:

Timed interface for patients to continually report their pain level through an Android page on your application that is permitting them to register their pain: We met this goal by implementing a slider, as referred to in section 7.1.2, to represent different levels of pain that listened for essentially a difference in the pain level rating (indicated by the position of the movable thumb). Each time the indicator thumb on the scale is moved by the participant rating their pain the state of it is recorded as a 'Pain Log' for the session.

Allow the patients to play selected music: By incorporating the Spotify API we fulfilled this requirement through use of Spotify's music library and additionally provided tools like the Spotify Player. This also made a Spotify Premium account necessary for using this application successfully.

Prompt patients to report if they did or did not take breakthrough medication at the end of the session: We made a class called MedicationActivity that achieved this requirement and would only call the PostsessionActivity class (implemented the participant logging out) once the submit button (viewed at the end of the MedicationActivity) was clicked.

Prompt patients to report an initial pain level at the start of the session and a final pain level at the end of the session: After the participant logs in they are immediately shown a view where the continue button actually starts the session (and shows a view) if they move the thumb on the scale, an indication of their pain level rating. This is implemented in the PresessionActivity class.

Provide secure means of transmitting and storing the application data and reporting it to secure servers: A hosted server with a MySQL database is where the application data is saved. It can only be accessed with the right login credentials. After, it can be easily manipulated and viewed with SQL queries.

8.1.2 Nonfunctional

We successfully achieved most of our nonfunctional requirement goals (5.1.2) through incorporation of the server and designing in a way that could be easily built upon during future work. The requirements are as follows:

Security: Since we used a token-based authorization system as part of the Spotify API, the facilitation of logging in using Spotify account credentials is sound. The application data about the session is identifiable without compromising the information of the patient. The session ID is assigned using the provided Java class UUID, which generated random universally unique identifiers. The participant ID also does not allude to any specific participant, as it just exists to differentiate sessions, and group them, by the participant who provided that data. These choices were also made to preserve anonymity.

Scalability: The mobile application itself will operate independently of other users, with the only shared resources competed for being hosted by Spotify. Spotify is a large company with vast computing and networking resources and the number of users for our application is sure to be relatively insignificant. The app's communication with the central database server is one-way, as data is only stored to that location and not retrieved. This eliminates the concern of bottlenecking, as the app's activity is not blocked by or contingent on fulfilling network requests as they can just be processed on the server side at any time. Furthermore, separate databases could easily be established at different locations to further distribute networking overhead.

Usability: As the participants of this study will be expected to use it on a mobile device, we kept views simple and tried not to overcrowd the screen. Clickable objects are large and distinct enough such that participants with dexterity issues will not encounter difficulty selecting the correct one. The application also directly displays worded information to aid and inform the participant when prompting them to complete an action.

Performance: The IMAGS app has the ability to perform all the necessary components to make it work. It is efficient, with little to no input or resource-loading lag, and was designed with the users (participants and researcher) in mind.

8. 2 Regression Testing

8.2.1 User Test

	Feb 13 2018	Feb 27 2018	March 25 2018
As a patient, I can rate my level of pain while listening to the song so that I can record how it has changed.	yes	yes	yes
As a patient, I can register my Spotify account with the study so that it will be identifiable in the study	not yet	yes	yes
As a researcher, I can relate users' account to patient IDs via a cryptographic function so that patient anonymity is preserved?	not yet	yes	yes
As a patient, I can rate my level of pain at the beginning of the session so that I can record its initial severity.	yes	yes	yes
As a patient, I can use my existing Spotify account to identify myself when using the app	yes	yes	yes
As a patient, I can report if I took a breakthrough medication at the end of the session.	yes	yes	yes
As a researcher, I can see the patients breakthrough medication status	not yet	not yet	yes
As a researcher, I can see the initial and in-progress pain levels of the patients in each session so that I can determine how they changed	not yet	not yet	yes

As a researcher, I can see what songs were listened to in each session so that I can determine their respective effectivenesses.	not yet	not yet	yes

9. Future Work

9.1 Registration System

With the current implementation, each participant is required to login with a Spotify Premium account in order to use the application. However, in some situations that account might not necessarily be their own individual account, such as if the participant does not have a subscription to the service. With this possibility in mind, it is expected that use of the application will sometimes, if not always, involve the creation and retention of some number of reusable or shared accounts by the administrators, the credentials of which would be subsequently distributed to participants who need them. A system which would be utilized to set up an account for a participant to use could be created, which could provide a number of possible services. One possible feature would be a way in which to track which of these reusable accounts is currently assigned and to whom. This would assist in identifying which data within the database corresponds to which particular participant based on the intersections of the time at which the session occurred and the period in which a participant was assigned that account. Secondly, this system could maintain a registry of both personal and reusable accounts that are validated as part of the study. Such a registry could be used by the application as a source to check against to authenticate users as they log in at the beginning of each session, which would restrict access only to approved individuals. Additionally, users could be added to different lists or given attributes to allow the app to distinguish them for purposes such as control groups and investigating other specific variables.

9.2 Administrator Console

Another major addition that could be expected to follow this project would be the development of an application designed specifically for desktop use by administrators to manage various aspects of the study. Perhaps the most obvious features of this application would be the ability to view, organize, and transfer data collected by the mobile application that has been added to the central database; to add or remove individuals and/or accounts from the study; or to manipulate certain elements of the mobile app's behavior in order to control for certain variables. This would ideally make use of a custom graphical user interface and could be designed in such a way as to be easy to use and configure important components of the study without requiring advanced technical skills.

9.3 Administrator-Defined Playlists

A prospective approach to this study would include experimentation controlling variables relating to certain aspects of the music listened to by participants. To facilitate this, it could be useful to implement a feature in which participants are able to listen to specific playlists that are created by the administrators of the study. The app would perhaps detect whether the participant has been assigned to a playlist group by checking their user id on login or by random assignment within the app itself.

9.4 Control group

For research purposes, it may be useful for administrators to include a control group in their studies. Such a control group could be designated as being comprised of participants who do not listen to music at all during the session but would otherwise use the rest of the application in the same manner as the experimental group. To this end, the app could be configured to detect

whether the participant is a part of the control group as they log in and, if so, skip presenting the

participant with the search function and player while still allowing them to use the pain tracker to

record their pain during either a predefined or arbitrary duration of time.

9.5 Progress Bar and Seeking

Another small but useful feature to consider for future implementation could be a

progress bar linked to the playback position of the current track. Furthermore, this could be made

into a seek bar, which would allow the participant to manipulate a draggable "thumb" to seek

within the track.

10. Conclusion

This MQP successfully delivered an application for playing music from the Spotify music library, that presented a pain scale that participants could interact with and gathered the subsequent participant data (rated pain levels, song information, and the participants medication status). It met most of the requirements that the research proposal draft outlined. Though adapted to our small team size, the experience working on the application was a valuable exercise in utilizing the software development methodologies described in the paper, such as Agile and Scrum. We overcame considerable challenges over the course of the project, especially in regard to discovering the inadequacies of our initial design and assumptions concerning the use of the Spotify widget in a web application as originally proposed. This taught us the importance of a thorough design process, as we were able to apply the majority of the underlying application architecture to the new software development environment. Furthermore, we gained experience with how to handle unexpected issues and how to devise and execute alternate solutions in a timely manner in order to minimize delays in productivity and the end product delivery.

References

American Chronic Pain Association. *eMTCP Music App*. Available at: https://theacpa.org/music-app (Accessed: 14 September, 2017)

The Echo Nest (2014) Spotify Acquires The Echo Nest. Available at: http://the.echonest.com/pressreleases/spotify-acquires-echo-nest/ (Accessed: 21 March 2018).

Eeles, Peter. "Capturing Architectural Requirements." IBM., last modified -11-15, accessed Oct 16, 2017, http://www.ibm.com/developerworks/rational/library/4706.html.

Garcia, Rebecca L. and Christopher J. Hand. 2016. "Analgesic Effects of Self-Chosen Music Type on Cold Pressor-Induced Pain: Motivating Vs. Relaxing Music." *Psychology of Music* 44 (5): 967-983. doi:10.1177/0305735615602144. http://journals.sagepub.com/doi/full/10.1177/0305735615602144.

Git. Git. Available at: https://git-scm.com (Accessed: 12 March 2018).

GitHub (2017) Built for developers. Available at: http://github.com (Accessed: 12 October 2017).

Institute of Medicine (2011). *Relieving Pain in America: A Blueprint for Transforming Prevention, Care, Education, and Research*. National Academies Press, Washington, DC;

Institute of Medicine Report from the Committee on Advancing Pain Research, Care, and Education: *Relieving Pain in America, A Blueprint for Transforming Prevention, Care, Education and Research*. The National Academies Press, 2011

Jafari, H., Zeydi, A. E., Khani, S., Esmaeili, R., & Soleimani, A. (2012). *The effects of listening to preferred music on pain intensity after open heart surgery*. Iranian Journal of Nursing and Midwifery Research, 17(1), 1–6.

Jamison, Robert N. et al. (2015). *Opioid Analgesics*. Mayo Clinic Proceedings, Volume 90, Issue 7, 957 - 968

McIntyre, Hugh. *Apple Music Can Now Claim 27 Million Subscribers, Showing Enormous Growth*. Available at: https://www.forbes.com/sites/hughmcintyre/2017/06/05/apple-music-can-now-claim-27-subscribers-showing-enormous-growth/ (Accessed: 10 September, 2017)

National Institute on Drug Abuse. "Overdose Death Rates." Available at https://www.drugabuse.gov/related-topics/trends-statistics/overdose-death-rates (Accessed: 19 September, 2017)

Parker, Nick. *Heartbeat App Fights Pain with Music*. Available at: http://evolver.fm/2011/03/09/heartbeat-fights-pain-with-music/ (Accessed: 14 September, 2017)

PCMag Digital Group. "Trello." PCMag Business Software Index., last modified -05-04T20:11:23+00:00, accessed Oct 12, 2017, https://www.pcmag.com/business/directory/project-management/951-trello.

Pew Research Center (2017). *The evolution of technology adoption and usage*. Available at http://www.pewresearch.org/fact-tank/2017/01/12/evolution-of-technology/ft_17-01-10_internetfactsheets/ (Accessed: 10 September, 2017)

The PHP Group (2018) PHP: Hypertext Preprocessor. Available at: http://www.php.net (Accessed: 12 March 2018).

Official Google Blog (2014) Save More with Google Drive. Available at: https://googleblog.blogspot.com/2014/03/save-more-with-google-drive.html (Accessed: 12 October 2017).

"Slack: Where Work Happens." Slack., accessed Mar 10, 2018,
https://slack.com/lp/two?cvosrc=ppc.google.slack%20messenger&cvo_campaign=&cvo_campai
gn=&cvo_crid=232283764453&Matchtype=e&utm_source=google&utm_medium=ppc&utm_c
ampaign=brand_mv&c3api=5542,232283764453,slack%20messenger&cvosrc=ppc.google.slack
%20messenger&cvo_campaign=&cvo_crid=232283764453&Matchtype=e&utm_source=google
&utm_medium=ppc&c3api=5523,232283764453,slack%20messenger&gclid=Cj0KCQiAoY7V
BRDtARIsAHWoO-INUF3zzuhIMPNpnJLCp7jhphf3Q4q6o-
Idw6uJqWLdPHj8eg7RPvUaAkX5EALw_wcB&gclsrc=aw.ds&dclid=CLS465fT4tkCFaXJ4wc
dMe0O1g.

Spotify. Spotify Developer. Available at: https://developer.spotify.com/ (Accessed: 12
September, 2017)

Sprouse-Blum, Adam S., Greg Smith, Daniel Sugai, and F. Don Parsa. 2010. "Understanding
Endorphins and their Importance in Pain Management." Hawaii Medical Journal 69 (3): 70.
http://www.ncbi.nlm.nih.gov/pubmed/20397507.

"Task Management Software for Work: Keep Your Team on Track · Asana." Asana., accessed
Mar 11, 2018, https://asana.com/uses/task-management.

"Try Canvas | Free LMS Trial | Canvas Learning Management System." Instructure., accessed
Oct 12, 2017, https://www.canvaslms.com/try-canvas.

"What is Agile? What is Scrum?" CPrime. February 13, 2018. Accessed Fall 2017.
https://www.cprime.com/resources/what-is-agile-what-is-scrum/.

Wong-Baker FACES Foundation (2016). Wong-Baker FACES® Pain Rating Scale. Accessed
September, 2017 with permission from http://www.WongBakerFACES.org. Originally published
in Whaley & Wong's Nursing Care of Infants and Children.   © Elsevier Inc.

World Wide Web Consortium. Introduction to XML. Available at:

https://www.w3schools.com/xml/xml_whatis.asp (Accessed: 12 March, 2018)

Appendix

User Manual

1. **Welcome Screen**

　　This is the screen which is displayed when the application is initially opened on the device. There are only two options available at this screen. To proceed, tap the "Continue" button. Otherwise, tapping the "Help" button will open a dialog window that gives more information about the app, such as who is intended to participate and how to contact support.
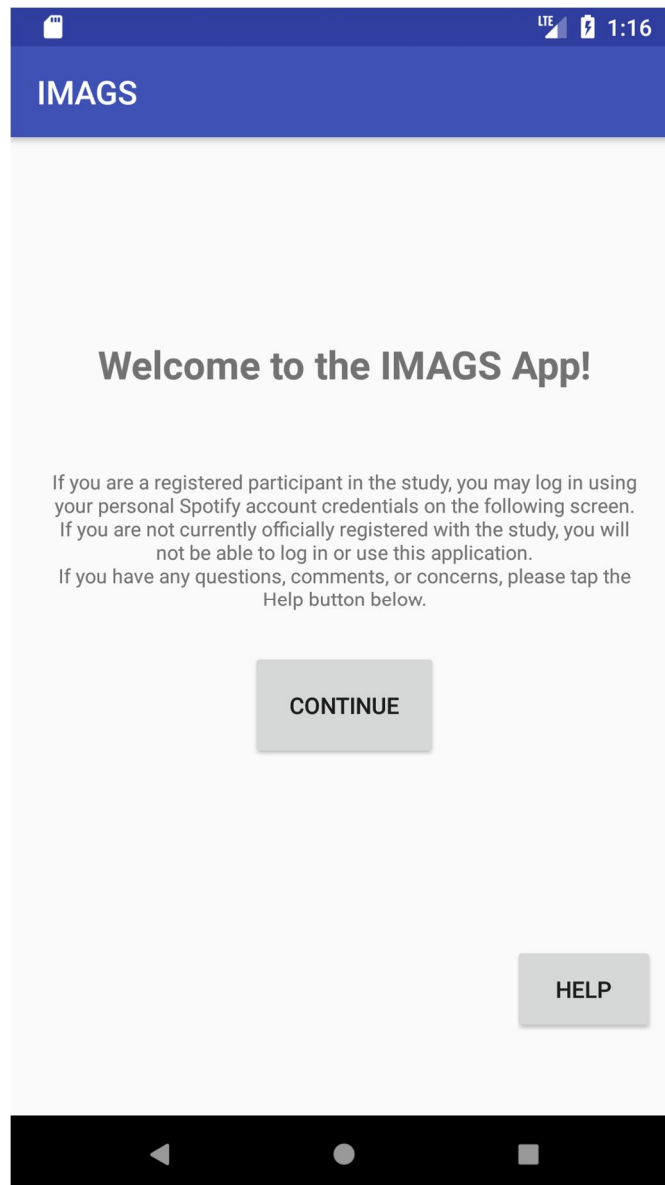
2. Login Screen

If not already logged into a Spotify account on the device, you will be prompted to do so now. Once successfully logged in, a confirmation message will be displayed. Tap the "Continue" button to proceed to the next screen. The "Help" button is again present, which will display the same information as it did on the previous screen.
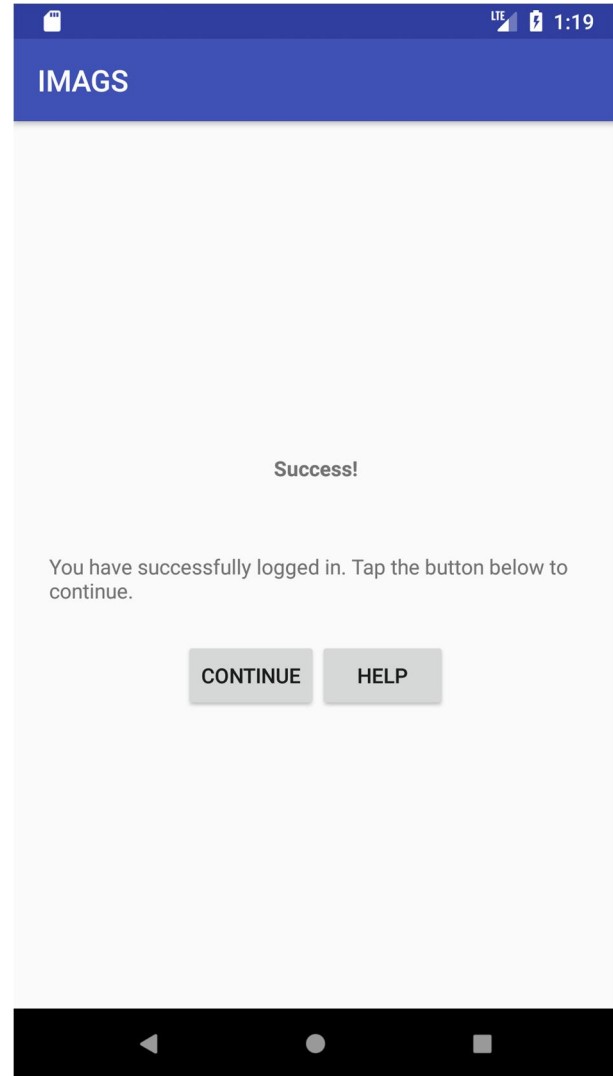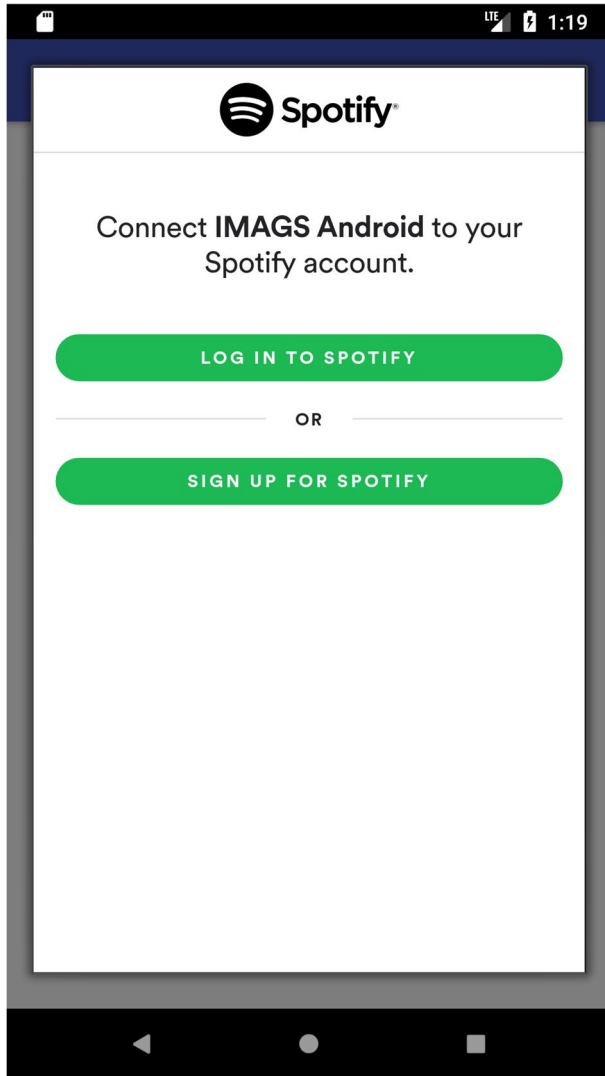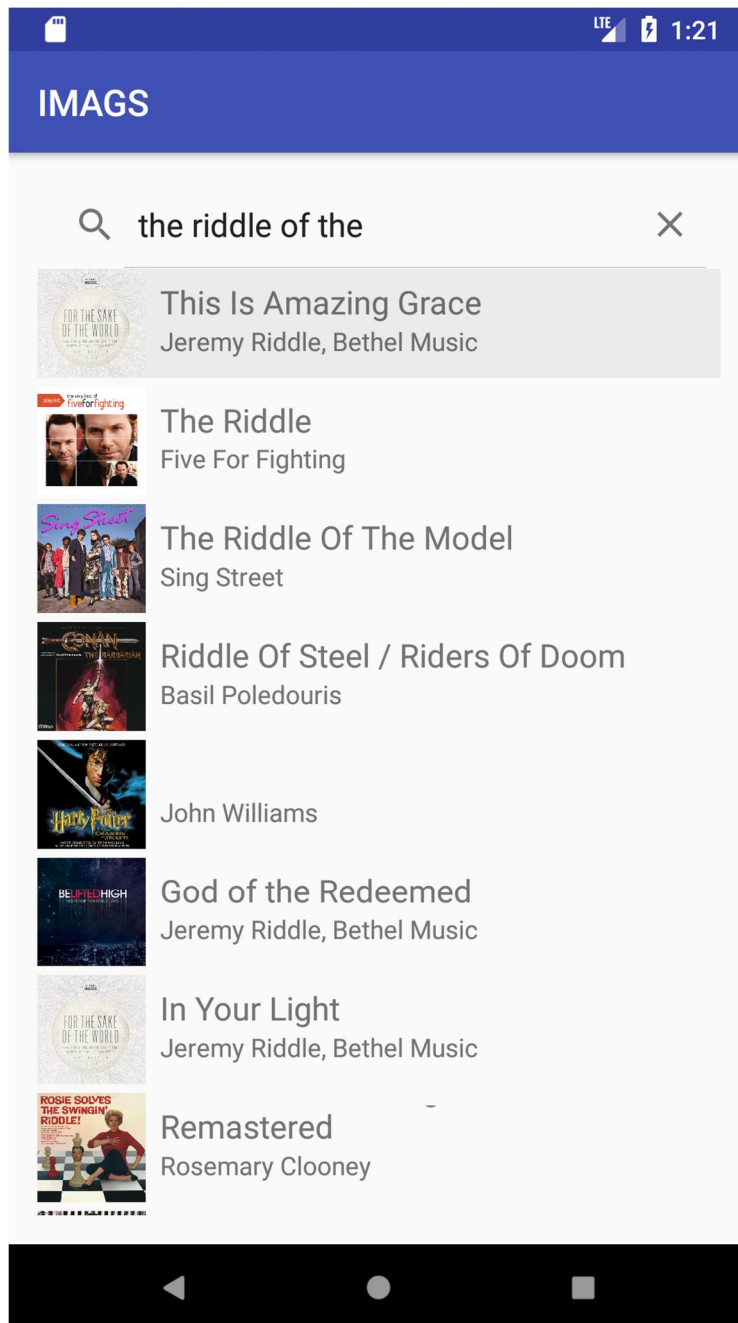
3. Presession Screen

Presented here is information about what will be done during the session, such as manipulating the pain-tracker to match your level of pain. Before beginning the session, the pain-tracker must be set according to your current state. Tapping the "Help" button here will explain more information about the pain-tracker and its usage.
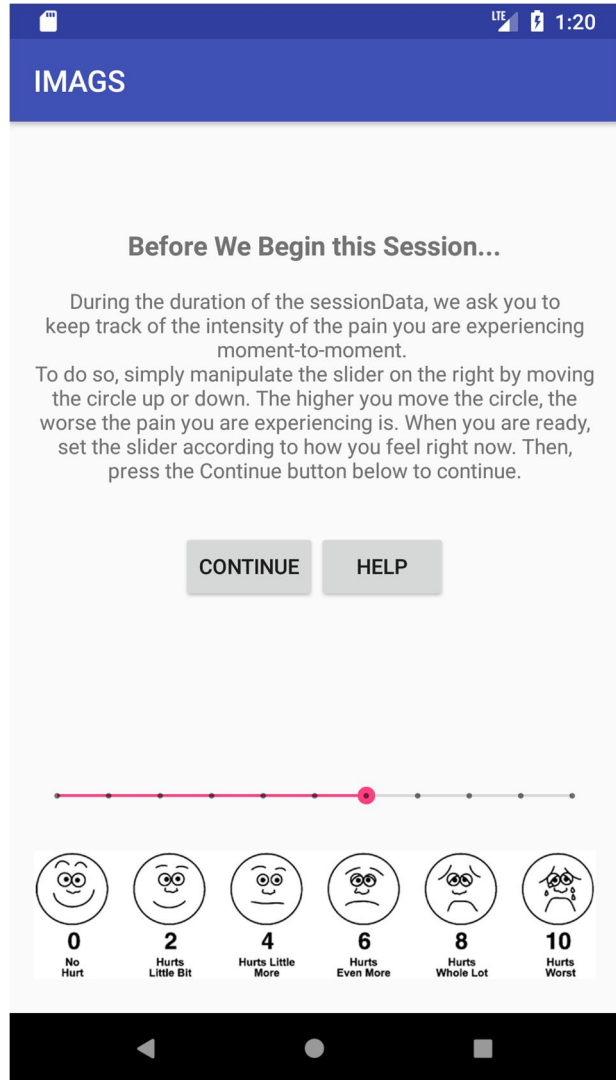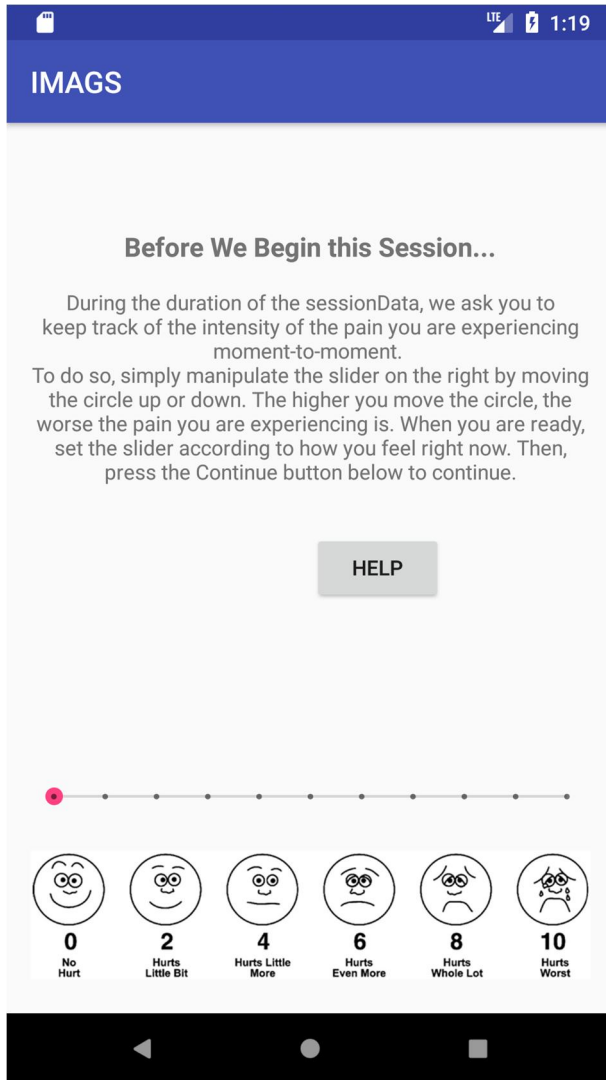
4. Search Screen

At this stage, a search bar can be used to search the Spotify library for the first track to be played. This is done by entering a search term into the field at the top of the screen and then pressing the enter button or tapping the search icon on the right side to submit. A list of results will be displayed below, which can be scrolled through by dragging a finger up or down on the list. To select a track and begin the session, the desired result from the list is simply tapped.
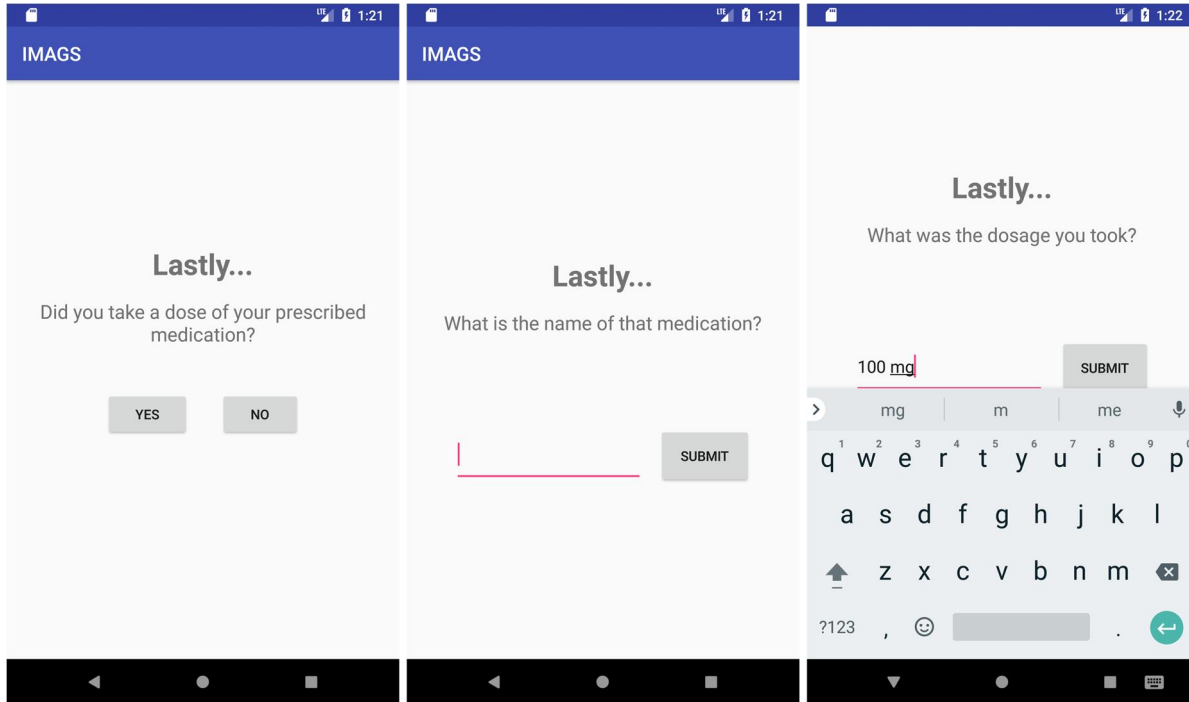
5. Session Screen

This screen is where the majority of time will be spent. The album art and basic information of the track that was selected at the previous screen will be displayed at the center of the screen here. At the top, there is a row of four buttons. The first of these, the "Help" button, opens a dialog window with directions to the user about what they are doing, as well as a reminder of how to use the pain-tracker. The next button is a toggle, displaying either "Play" or "Pause" according to whether or not the track is currently playing. Tapping this button will perform the action that the button is showing at that moment. Following that button is the "New Song" button. This button will return to the previous screen in order to search for and select a new track to be played. Finally, the last button, "Finish Session", ends the session. This button is used when done listening to music and completes this session of the study.
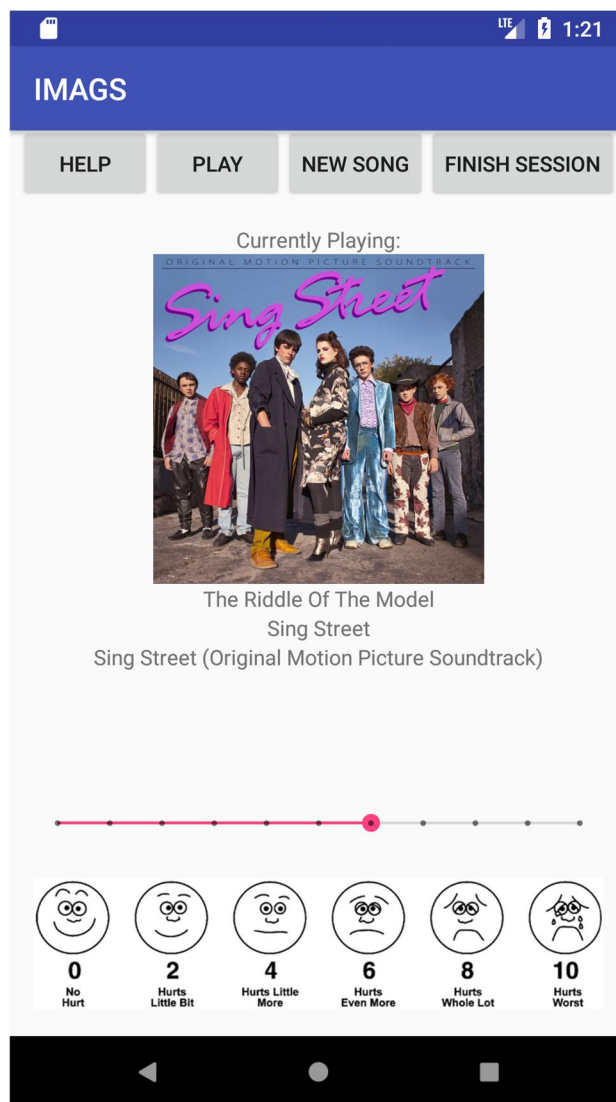
6. Medication Questionnaire Screen

At this screen, information is gathered about any medication which may have been taken. There are questions regarding whether any medications have been prescribed to treat chronic pain, what that medication is, and if it has been taken.

7. Logout Screen

Reaching this last screen indicates completion of participation of the current session of the study and are the option is given option to logout of Spotify before tapping the "Logout" button to close the app.