Mobile Application Development for Mindfulness Based Stress Reduction

An Interactive Qualifying Project
submitted to the Faculty of
Worcester Polytechnic Institute
in partial fulfilment of the requirements for the
degree of Bachelor of Science

by

Bryce Corbitt

Date

January 29, 2020

Report Submitted To:

Professor Eleanor Loiacono
Worcester Polytechinc Institute

Contents

Abstract

The goal of this Interactive Qualifying Project (IQP) was to implement the first prototype of "DoYouMindful", a mobile application to assist college students in the practice of Mindfulness Based Stress Reduction (MBSR). This prototype serves as a proof-of-concept for previous designs, as well as a logistics test for future features. This IQP was a continuation of two previous Major Qualifying Projects (MQP) that laid out initial designs and deliverables for the app. From these projects and input from staff at UMass Medical School, the prototype application was developed for iOS and Android. Additionally, a back-end system was designed and implemented to manage and serve dynamic content to users.

Acknowledgments

I would like to extend my sincere thanks to all those who made this project possible by contributing their time and resources, including:

- Professor Eleanor Loiacono for advising the project and providing feedback and organizing project tasks and meetings.

- Dr. Daniel Amante and fellow staff at UMass Medical School for providing insight into the MBSR program and recommending useful features to implement throughout development.

- Executive Director of Information Technology Siamak Najafi for providing a MacBook computer and iPhone to develop and compile the iOS build of the app, as well as a WPI Virtual Machine to deploy the back-end of the project on.

1 Introduction

Mindfulness is the state of being aware of one's self, one's surroundings, and the present moment. This state is often achieved through silent meditation, allowing one to focus entirely within the present moment. Due to a lack of reliable sources and evidence to prove the effectiveness of mindfulness, its practice receives mixed receptions from the public eye.

To examine the validity of mindfulness practice, various studies were conducted at UMass Medical School on individuals practicing various forms of mindfulness practice. Examined practices with success acquired classification under Mindfulness Based Stress Reduction (MBSR), the clinically proven application of mindfulness to address stress, anxiety, depression, and other mental health issues. UMass Medical School currently instructs MBSR in the form of an eight week course.

With increasing popularity of UMass Medical's evidence based mindfulness curriculum, the notion of finding alternative means to make MBSR resources more accessible was considered. A Major Qualifying Project (MQP) group in 2015 performed various surveys on students to determine the most effective method of accomplishing this. The team concluded that MBSR content would be best received by college students and young adults from a native mobile application (Facchini et al., 2015, 139-140). The team also prepared basic features for an app based on their research and provided initial mock-ups for what such a product would look like.

In 2017, a second MQP team returned to this project and created a more formalized proposal for the app. This included the improvement of envisioned features and the addition of new features. The 2017 team also performed a heavy redesign of the app's mock-ups for each screen, laying out groundwork to how the

app should look and feel with an emphasis on user experience (Falzone et al., 2017, 2).

The goal of this Interactive Qualifying Project (IQP) was to take the designs and research provided by the previous two teams and create the first iteration of a prototype app. A mobile app for MBSR will provide a gateway to effective and reliable resources for mindfulness practice to aid college students combat stress, anxiety and depression.

2 Background and Research

2.1 Review of Prior Research

2.1.1 2015 Major Qualifying Project

Figure 2.1: Mockup screens for the home screen (left), events screen (middle), and breathing exercises screen (right) by the 2015 MQP team.

Though MBSR has been proven effective, training has been primarily focused on adults. The goal of the 2015 MQP team's work was to explore what technologies that could be used to aid MBSR involvement with college students and young adults. To accomplish this goal, the team conducted various surveys and interviews on students and young adults. Questions focused primarily on using different technologies

and preferences for receiving aid for mindfulness on one or another. Technologies included online exercises, Youtube resources, a mobile app with guided exercises, Facebook groups, Twitter, and a mobile app with a timer. The team was able to conclude that a mobile app capable of streaming mindfulness exercises was the most popular amongst the majority of individuals studied. From this conclusion, the group created the first designs for a mobile app to be created (Facchini et al., 2015, 12-14). This included requests for many of the features planned in the current prototype, as well as the first mock-ups of what the app should look like.

2.1.2 2017 Major Qualifying Project

Redacted

Figure 2.2: Mockup screens for the home screen (left), events screen (middle), and breathing exercises screen (right) by the 2017 MQP team.

In 2017, a second MQP team returned to the project to further refine the app design and revise feature requests in detail (Falzone et al., 2017, 7). This was achieved by conducting similar surveys and interviews with students to obtain feedback on the designs that the prior team had created. The majority of students expressed a more calming appearance for the mock-ups of the app. From this reception, the team prioritized their focus on re-designing the app mock-ups (Falzone et al., 2017, 50-58). This was a critical step for the app, as these updated designs were heavily relied upon for developing the front-end of the prototype app.

2.2 Identify Key Features for Implementation

An essential step in the development of any software prototype is to properly define the deliverables that the software aims to achieve. Many of the goal features of the app were conceived through the previous two MQP groups, but some were envisioned and implemented throughout the development of this project as well.

2.2.1 Features From Prior Research

Countdown Timer

The need for a countdown timer came out of the 2015 MQP team's research defined as analog timer for tracking self-guided meditation practices. It included simple controls to add and remove time, as well as to pause and resume the timer. This concept was improved upon by the 2017 group, adding a digital representation of the remaining time to the screen to make it easier to read.

Calendar

The request for a calendar screen was created to serve as a place to schedule reminders to use the app and maintain regular mindfulness practice. To properly budget time for development, this feature was simplified to become a simple screen for scheduling notifications. The original request for a calendar screen was also reached from a joint screen with the timer, labeled "Calendar and Timer". For ease of use, this navigation screen was removed and both calendar and timer screens were planned to have dedicated buttons on the home screen instead.

Exercises

The requested exercises screen was envisioned for users to stream various MBSR audio and video materials to aid in mindfulness practice. The original proposal from the 2017 group proposed four main sections: "Yoga Exercises", "Breathing Exercises", "Inspirational Preparation", and "More Guided Exercises on Youtube". Each section was designed to contain both audio and video exercises on the same screen. Because it was unsure at the time of development to organize content in this manner, the exercises screen was revised to contain a dedicated screen for audio exercises, and another for video exercises. The Youtube exercise screen was removed from the design with the mindset that keeping all MBSR streamed content self-hosted would ensure optimal resource reliability and data integrity.

Inspirational Corner

An inspiration section of the app was requested where users could go to view inspirational quotes from various social media accounts. Unfortunately, the practicality of streaming multiple accounts from different social media platforms on the same screen was not valid. Additionally, retrieving content from external sources like

Instagram and Twitter raises the same content reliability as the aforementioned Youtube exercise screen. Because of these reasons, this screen was not planned for implementation in the prototype.

Mindfulness Around the World

The 2017 MQP team also envisioned a section for the app to obtain mindfulness articles, including MBSR studies and various success stories with mindfulness practice. It was unsure whether this feature would extract content from an external source or be hosted internally. Because such a source could not be found for demonstration, this feature was removed from the prototype plan. However, this concept was taken into account when designing the back-end of the platform in hopes that it can be added in the future if content is available.

College Room

The college room screen was planned to serve as a chat section for users to discuss MBSR practice with each other. Unfortunately, the designs to how this section would be implemented was inconsistent: Some mock-ups from the 2017 group contained messages organized similar to a group messaging system, while others instead had individual posts prioritized by a voting system similar to forums and discussion services like Reddit. It was also undetermined whether users would remain anonymous amongst each other, and how such a system would be properly moderated to ensure the safety of users. Needless to say, this feature was removed for the prototype design.

2.2.2 Requested Features Throughout Development

User Account System

A feature request to implement an account system was made during development by the UMass Medical School staff in hopes that it could be used to store user profile information and create custom content feeds to suit the preferences of each user. While no designed infrastructure in the prototype could easily take advantage of a system like this, it was added to the prototype design with the understanding that it will be of use in the future. This basic implementation was designed to have means for creating an account, as well as logging in, viewing profile information, and account recovery. Additionally, a request system in order to register accounts as well was planned as well to properly maintain control over the user base.

Content Serving Features

Many of the features requested from the prior MQP teams relied heavily on the serving of content to users. While static entries for content may be used in place for the purposes of displaying a prototype, it does not accurately depict how the prototype can be translated into production. Questions like "How will the app know what content to stream?" and "How will this app scale with an increasing user base?" are critical considerations that cannot be addressed when static content is used in substitute for a prototype. Because of this, plans for creating a back-end to support the prototype app for dynamic management of self-hosted content was researched, selected, and implemented for the prototype.

3 Methodology

3.1 Cross-Platform Development

During various meetings with UMass Medical throughout the development, the feasibility of providing iOS and Android versions of the app was discussed in depth. UMass Medical instructor Daniel Amante noted the importance to support both mobile platforms as many patients only have access to one or the other. While not necessary for a prototype application, the need for cross-platform support in production was evident. Releasing for both mobile platforms would not only significantly increase usability across the project's target demographic, but also establish a professional presence to stand out on both respective app stores.

To explore the possibility of providing cross-platform comparability in the prototype app, I researched various techniques that mobile developers use to develop iOS and Android apps alongside each other. A common technique that small companies often use to achieve this to build their app with React Native (2019), framework developed by Facebook that takes project code written in JavaScript and compiles it to Android Studio and iOS Xcode projects. Although support to features unique to either platform is limited, the framework's capabilities were suitable for achieving the deliverables of the prototype app, and provided great aid to streamlining development.

3.2 Back-End Design

A back end platform was designed to enable the distribution of dynamic content to the mobile app. The platform is composed of three main services: A database

service, a CMS server, and the MBSR server (see Figure 3.1)2.

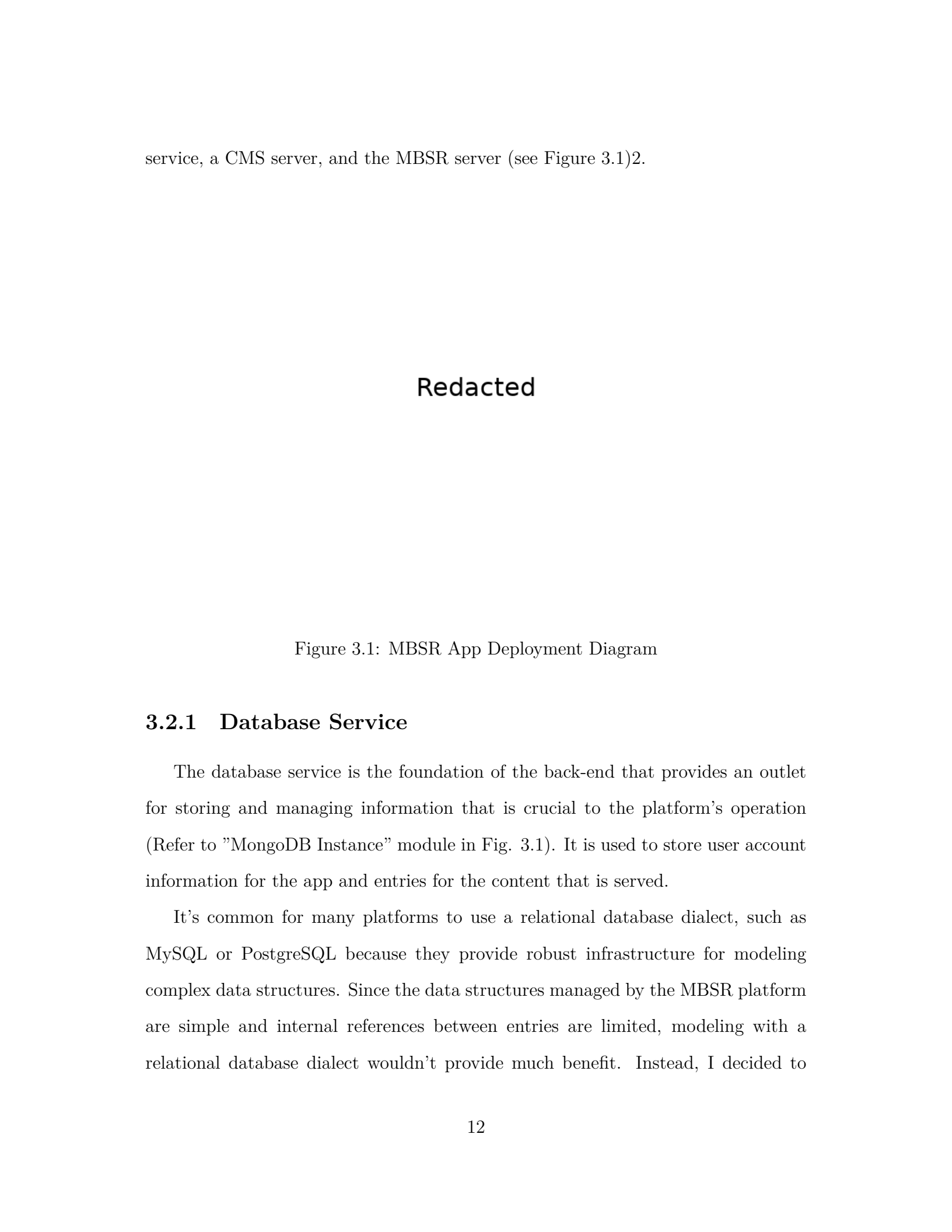

Figure 3.1: MBSR App Deployment Diagram

3.2.1 Database Service

The database service is the foundation of the back-end that provides an outlet for storing and managing information that is crucial to the platform's operation (Refer to "MongoDB Instance" module in Fig. 3.1). It is used to store user account information for the app and entries for the content that is served.

It's common for many platforms to use a relational database dialect, such as MySQL or PostgreSQL because they provide robust infrastructure for modeling complex data structures. Since the data structures managed by the MBSR platform are simple and internal references between entries are limited, modeling with a relational database dialect wouldn't provide much benefit. Instead, I decided to

employ MongoDB (2019), a document-oriented database dialect. Unlike relational dialects, document-oriented databases store data in key-value pairs similar to a hashmap or dictionary. While modeling complex data structures is more challenging, document-oriented dialects provide a performance advantage with respect to managing data in high volumes concurrently. Because the majority of data in the MBSR platform is content that will be sent to multiple users, this advantage is important as the amount of content and users requesting content at a given time scales.

3.2.2 CMS Server

A Content Management System, or CMS, is a utility used to manage content commonly used for storing information on websites. CMS servers typically provide a web portal to manage the content that is then served. This provides an easy to use interface for people who want to publish their own website with minimal web development experience.

Having worked with CMS services in the past, I researched the feasibility of incorporating one into the MBSR platform to provide an easy interface to manage the content that gets streamed to the app. I then discovered Strapi (2019), an open source CMS service with enough versatility to achieve this goal (Refer to "CMS Server" module in Fig. 3.1). Unlike most CMS services, Strapi is a headless CMS. Instead of being bound to rendering HTML pages for a website like most CMS services, Strapi instead serves its content in a JSON format that can be used across multiple applications. This data is requested with a REST API, a set of HTTP requests that Strapi generates to manage each content type. Strapi also supports the ability to define custom content types, a feature that is extremely beneficial for customizing the formatting for content to be served on each screen of the application. Because Strapi is open source, the CMS service is self hosted and modified to best

suit the needs of the MBSR platform.

3.2.3 MBSR Server

The MBSR server is the heart of the platform that bridges the back-end services together and serves content to the user's phone. Any request made from the app is sent to the MBSR server, which then processes the request, gathers data to create an appropriate response, and serves it back to the user. In its current state, each feature in the app could have been implemented without the MBSR server. However, its integration is important for production for three main reasons:

1. **It provides an additional layer of security**. More specifically, it can be used to restrict access to the CMS server. Any request from the app has to go through the MBSR server first, acting as a filter for what users can access. This also allows for additional security measures to be taken on the CMS server. The implementation of Strapi in this platform is configured to require administrator credentials in order to access or modify any content. The MBSR server is equipped with these credentials, allowing it to make authenticated requests to retrieve content and serve it to the user.

2. **It reduces complexity for client requests**. In some situations, multiple requests may need to be made in order to properly render content. Coordinating these requests from the client side can be challenging, and requires more code on the app to be implemented to handle cases when these requests fail or become unreliable. The MBSR server provides a layer in between the client and rest of the back-end so that app only needs to make a single request to retrieve the data it needs.

3. **It keeps the endpoints in control on the server side**. With HTTP based

APIs, the endpoint is portion of the request URL that determines what code on the server is executed to process the request. For example, the current endpoint URL to access account information on the photo sharing service Instagram is as follows:

`https://api.instagram.com/v1/users/self/?access_token=ACCESS-TOKEN`

Instagram (2019)

Hypothetically, this request could be integrated into a mobile app to access account information. However, if Instagram changes the URL that is to be used at any point in the future, the URL in the app code then needs to be updated as well. Additionally, any user with the current version of the app that makes the call will not work until the updated app is released and installed.

Integrating the MBSR server into this scenario, the mobile app would make instead make a request to the MBSR server, which would then receive a forwarded response. If the Instagram endpoint URL changes at any point in the future, the URL can be updated in the MBSR server code without any change to the app.

3.2.4 Portability With Containerization

Deploying a back-end that relies on multiple services often involves multiple steps that can vary depending on the deployment environment. The process to deploying a service on one operating system may be different than another, and usually depends on different libraries being installed on the host machine.

To make the deployment process for the MBSR platform as simple as possible, I integrated containerization for each of the back-end services with Docker. Docker (2019) is a virtualization tool used to run services isolated from the host, creating an

environment similar to that of a virtual machine for each service to run in. Docker also also enables the setup for these environments to be scripted. With each service's installation scripted, a multi-service back-end with a complex deployment process can be executed with a single command. The isolation of the service from the host machine also ensure that the deployment process will work on any machine with Docker installed, regardless of operating system or environment.

3.3 Source Control

For large software projects, keeping records of changes that are made in the source code is crucial. This is especially important when multiple developers work on the same project concurrently. I have created a source repository using Git (2019), a commonly used source control utility, to track changes to the prototype codebase. This repository is stored on the Git hosting site Github (2019), where it is publicly available with instructions on how to deploy the back-end and development environment for the prototype app.

4 Results

4.1 The Content Manager Experience

4.1.1 Managing User Accounts

Infrastructure for an account system has been developed for users with hopes that it can be utilized in the future for tracking progress with MBSR practice and other features to engage users with MBSR. To create an account, entry for a user is specified on a CSV file in the code repository. Upon starting the MBSR server, the user's name and email will be logged with a unique URL that can be used to fill out a registration form to create a username and password. Upon valid submission, the user will be redirected to a success screen and be able to use the entered credentials to log in on the app.

4.1.2 Managing Content

Separate from user accounts, the persons responsible for managing the content that is streamed to the app have their own accounts to be used with the CMS web portal.

Navigating Content Types

Upon logging into the portal, content managers are greeted with a homepage that contains each of the platform's defined content types listed on the sidebar. Clicking on any of them will redirect to a table view that can be used to search and filter each entry belonging to the selected content type (see Fig. 4.1).

Redacted

Figure 4.1: The Table View for "Audio Exercise" entries in the CMS web portal.

Creating New Content Entries

From the table view of any content type, a blue "+ Add New [Type]" button is present that can be used to create a new entry under the selected type. Upon clicking this button, the content manager will be redirected to a form page that contains inputs for each fields that define the content (see Fig. 4.2). If the content type relies on a photo, video, or audio file, the input for the field will allow the content manager to upload a file from their computer. Clicking the blue "Save" button in the top of the creation screen will persist the entry and allow it to be served in the app.

Redacted

Figure 4.2: The creation form for a new ”Video Exercise” entry.

Managing Existing Content Entries

The process to modifying existing content entries is similar to the creation process. From the table view of a given content type, the content may click on an entry in the table to get a form of the entry with inputs that are automatically filled with the existing values for the entry (see Fig. 4.3). These fields can be changed and then updated by clicking the blue ”Save” button. Additionally, an entry may be deleted by clicking the red ”Delete” button. Any changes in the content in the CMS server will immediately be reflected upon how it is served in the app.

Redacted

Figure 4.3: Edit page for an MBSR "Event" entry and its render in the app.

4.2 The User Experience

4.2.1 Account Management

Requesting Account Registration

In order to register for an account, the user's information must be entered in the account registration CSV that is stored with the MBSR server code. This should not be used in practice, but serves the prototype to represent a system that could be developed to manage account registrations in the future. In order to request account registration, users may go to the MBSR server's account registration request page, where a form may be filled out to file a request (see top of Fig. 4.4).

On submission, an email will be sent to the account manager's email notifying

the request. If the request is accepted, the account manager may add the user's information to the CSV file. The next time the MBSR server is ran, a unique URL for the user will be generated in the log, which the user can then reach to complete the registration process by entering a username and password (see bottom of Fig. 4.4). If successful, the user may use these credentials to log into the mobile app.

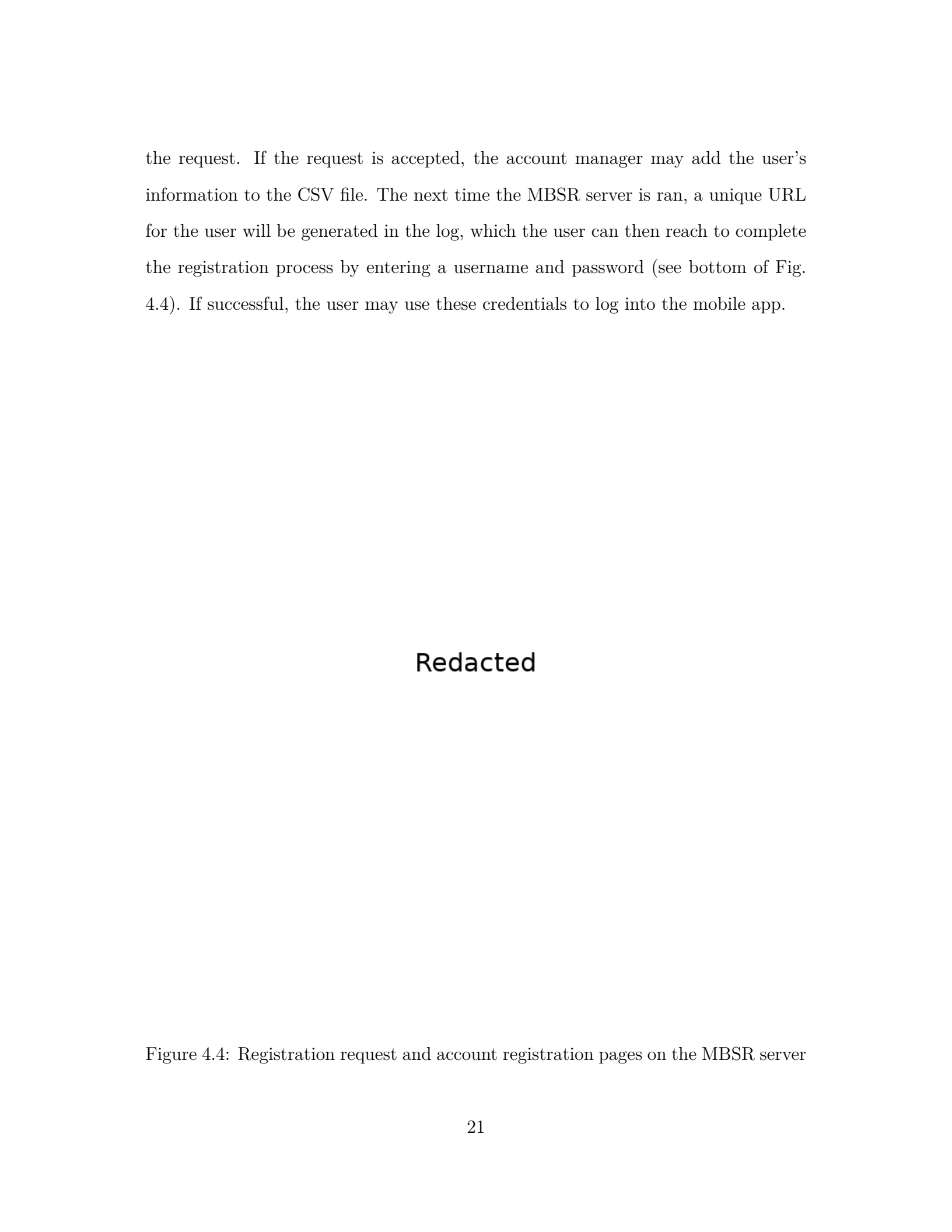

Figure 4.4: Registration request and account registration pages on the MBSR server

Signing Into Account in App

From the home screen of the app, users can log into their accounts by tapping the gear icon in the top right of the screen and then tapping "My Account". If already logged in, the user's account will be displayed. Otherwise, the user will be directed to the login screen where the credentials created during registration can be entered (see Fig. 4.5). Once logged in, the user's account information will be displayed on screen. Additionally, the user's username will be displayed in the upper left-hand corner of the home screen.

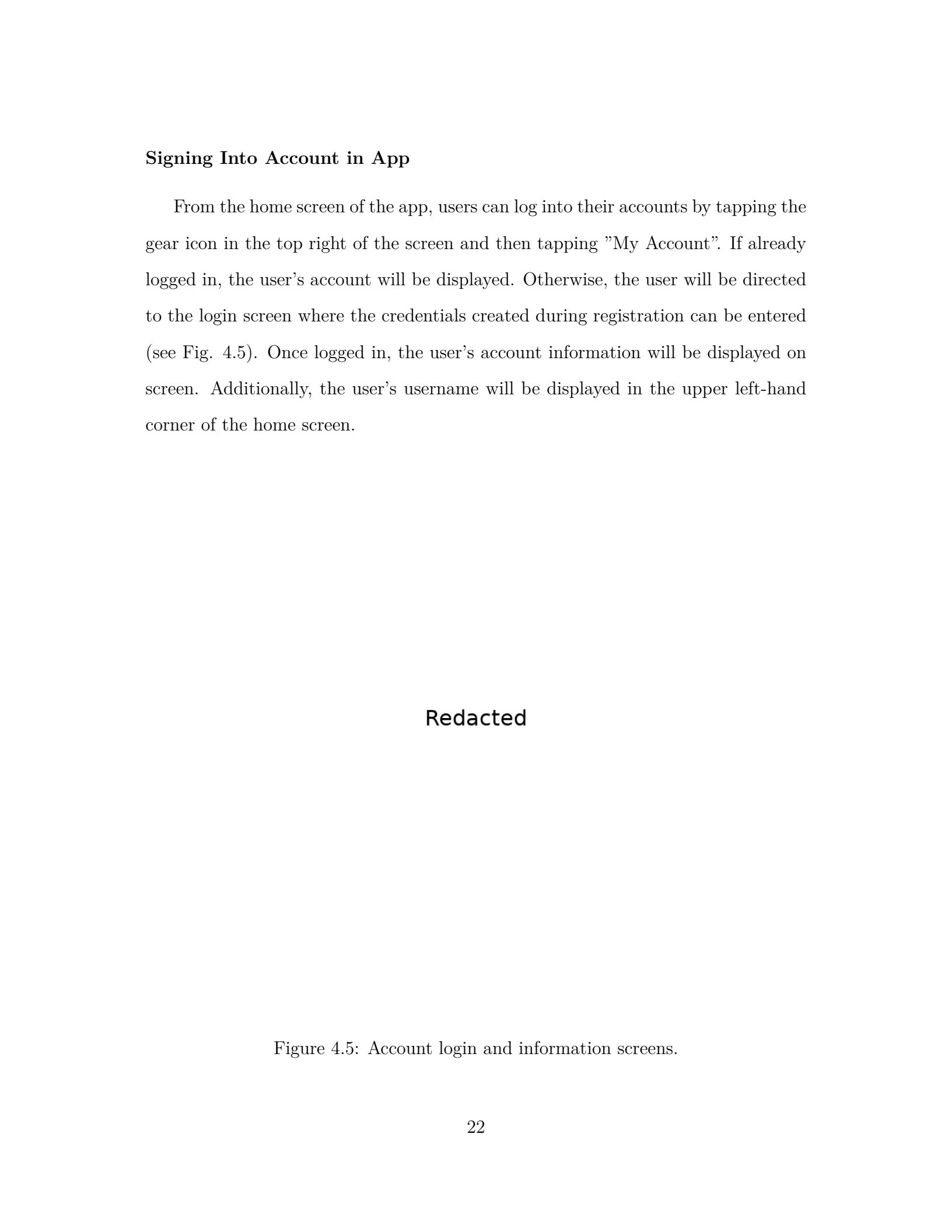

Figure 4.5: Account login and information screens.

Account Recovery

A basic account recovery system has been developed to regain access to accounts in the event that the password is lost (see Fig. 4.6). In the account login screen of the app, the user may tap the ”Forgot Password?” text directly under the password input to be redirected to the password recovery screen. After entering the email address associated with the account, the user will receive and email with a unique generated URL to set a new password. If successful, the user will be redirected to a confirmation screen, and will be able to login with the submitted credentials.

Redacted

Figure 4.6: The account recovery process.

4.2.2 Guided Meditations

Guided meditations and exercises are essential to become acquainted with mindfulness practice. To ensure that users have access to effective and reliable MBSR resources, support for streaming content server has been implemented.

Audio Exercises

From the Audio Exercises Screen, users may stream any audio recordings hosted on the CMS server. In addition to play/pause control, users may also scrub through clips to skip to specific parts of the stream (See left side of Fig. 4.7).

Video Exercises

While practice materials are more commonly audio focused, support for streaming video exercises has been implemented to help further engage users with MBSR resources (See right of Fig. 4.7). Similarly to audio streams, videos have play/pause and scrubbing controls. Additionally, users may put any video stream in full-screen mode to get a better view of the content.

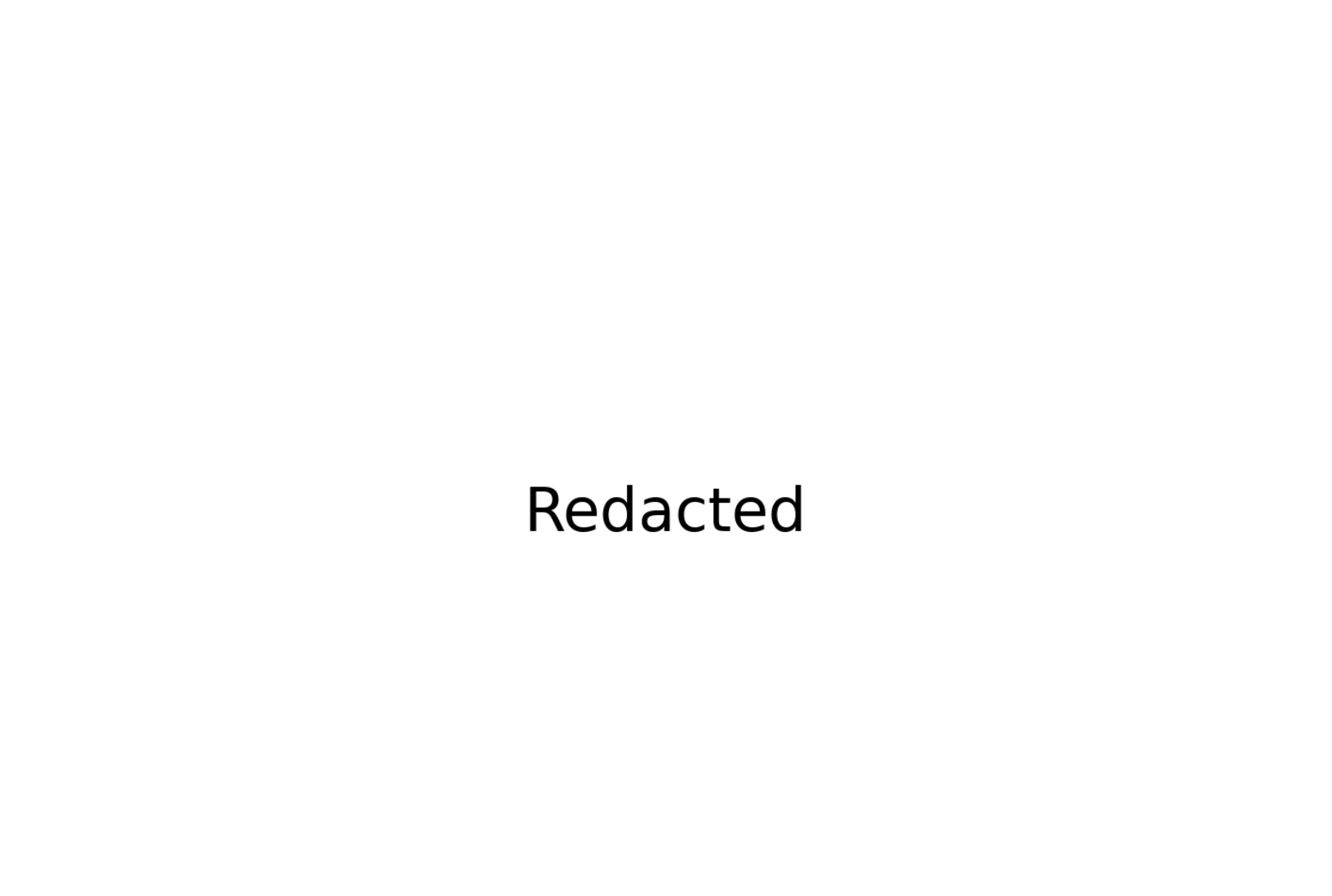

Figure 4.7: Screens for streaming audio and video exercises.

4.2.3 Self Guided Practice

When users become familiar with the fundamentals of MBSR practice, they may choose to begin meditation individually. To assist with this, a countdown timer has been developed for timing self guided meditations. Users may pause/resume the timer at any time during practice, as well as add or remove time from the timer. When the timer completes, a soothing tone will sound to indicate that the session is over. The timer may also be toggled to a "vibrate only" mode if the user does not wish to hear the tone after the session ends.

4.2.4 Local Involvement

To engage users with upcoming mindfulness events in their area, an events screen has been created that retrieves event entries from the CMS server. Each event entry contains the title of the event, as well as a description, location, and date. Additionally, photos may also be added to accommodate event entries (see Fig. 4.3). Events are sorted ascending date, so that the soonest events appear first on screen. The next day after an event has occurred, its entry will be automatically filtered out when the app requests for the event feed. While it is no longer visible to the user, the entry remains stored on the CMS server and can be made visible again by editing the entry to a future date.

4.2.5 Remembering to Practice

Performing MBSR practice regularly is important to fully benefit from it. From the reminders screen, users may schedule future dates to be reminded to practice mindfulness (see Fig. 4.8). After tapping the "Schedule a Reminder" button, the user will be prompted to select a date and time to be reminded to practice again. This will create a button on the screen that displays the scheduled date and time. Scheduled reminders may be canceled at any time by pressing this button. Upon reaching the specified date and time, the user's phone will deliver a notification as a reminder to use the app.

Redacted

Figure 4.8: Scheduling a reminder and receiving the scheduled notification.

4.3 Issues and Limitations

Throughout the prototype's development, I managed to achieve many of the deliverables defined by the previous MQP projects, as well as features requested throughout the development process. However, these accomplishments cannot be accepted without noting the flaws that remain in the MBSR platform's current state.

4.3.1 Development Limitations

Despite the contributions I have made to this project, it is evident that more could have been achieved if I had developed the prototype with other students. Because I developed this project alone, time became a large limiting factor. While the process making design decisions was efficient, having other members could have

provided important insight and provide a better solution. Additionally, I noticed multiple bugs throughout testing the application that I was unable to fix due to time constraints. Working with multiple people on this project would've allowed for more time to be taken during on developing features, and would likely prevent many of these bugs from existing.

4.3.2 User Interface Inconsistencies

Some of the most present issues while using the app on various devices are with respect to the user interface (UI). The each phone renders certain components like buttons and text can vary, and something that renders properly on one device may not for another. While I managed to fix some of these issues, extensive testing on various platforms should be performed in any iteration of this project in the future before going into production.

One of the largest UI issues was being able to scale elements properly depending on screen dimensions. While most of the phones with 16:9 aspect ratios rendered the app consistently with each other, longer phones with 18:9 aspect ratios varied. For example, testing on the Samsung Galaxy S8 rendered elements far more spread out than expected. While other phones of the same aspect ratio rendered the same way, some smart phones (such as the Google Pixel 3 and iPhone XR tested in emulation) obstruct rendered content if a notch is present on the device.

Another device-specific issue was noted while testing on the OnePlus5 smart phone. Specifically on Oneplus devices running React Native application, the text rendered can occaisonally be cut off of certain elements if the text is bold. While this issue was resolved by explicitly specifying the font in the text element, it also created another compatibility issue on the iOS version of the app. iOS does not have the same native fonts present on the OnePlus5 and other Android devices, so

it fails to render the screen. This secondary issue was fixed by writing platform specific conditional code for both iOS and Android.

4.3.3 HTML5 Streaming

Because of the virtualization from running each back-end service with Docker, there are very few operating system related issues with deploying the MBSR platform. However, a few large issues exist with streaming audio and video that I was not able to resolve. Currently, the Strapi CMS service does not have full support to stream HTML5 audio or video. While the files are capable of being transferred, the CMS lacks the ability to properly stream specific sections of an audio or video clip by default. This prevents users from being able to properly scrub through videos in the app.

In attempt to resolve this issue to the best of my ability, I wrote code for the CMS server to properly handle serving HTML5 streams. While my fix worked on all Android phones, iOS devices remain without proper audio streaming capabilities, and refuse to stream video entirely. This is noted as an ongoing issue by the Strapi Development team, and will hopefully be resolved in the future.

4.3.4 Persistence of App State

A problem that was not discovered until late in development is the inability for the mobile app to retain the state of a given screen after navigating to another screen. This results in many elements like the countdown timer and scheduled notifications to be lost if the user navigates to a separate screen and returns back. React Native libraries exist that can maintain a ”Global” app state and resolve this issue, but I was unable to integrate such a library into the prototype due to time constraints.

Similarly to state persistence across multiple screens, the prototype is also currently unable to retain app state after the app is closed. While this does not affect any currently implemented features, it is worth noting in case it is necessary for future development.

4.3.5 iOS Integration

The vast majority of issues exist specifically on the iOS version of the app. The iOS operating system is far more strict than Android with respect to network and permission related activity. Aside from the previously stated issues with streaming audio and video to iPhones, SSL encryption is required for all network connections of the app if the app is to be published on the iOS App Store. While SSL encryption is an essential for any application in production, I made the decision not to incorporate it into the prototype in its current state because it complicates the back-end deployment process. If testing on the iOS version of the app is ever to occur that would require registering the app on the iOS App Store, SSL encryption for all requests the MBSR server makes with users must be implemented.

Despite coding the app with React Native, the iOS version of the app still relies heavily on configurations through Xcode, the iOS integrated development environment (IDE). This means that a computer running Mac OS is required in order to test and compile the app for iOS. While I was able to borrow a MacBook and iPhone for the duration of this project to work on the iOS version of the application, this requirement is important to note for future development.

In addition to the requirement of a computer with Mac OS to compile the iOS version of the app, it is also worth noting that the computer will likely also require the most recent version of Mac OS in order to properly compile the app to run on an iPhone. One week after compiling the first successful iOS build for the iPhone 6S

used in testing, the phone automatically updated from iOS version 12 to 13. While this appears to be a small issue, I was unable to compile the app after the update because the version of Xcode in use did not support compiling apps for iOS13. Also, the latest version of Xcode could not be installed initially because it requires that the latest version of Mac OS is installed on the host machine. Once I was able to update the Mac computer in use, I was able to update to the latest version of Xcode and successfully compile for iOS 13.

5 Future Research and Development

There are multiple steps that can be taken in the future to progress development of the platform. While not immediately necessary, funding will play an important role in the rate at which future milestones can be reached. Therefore, taking the necessary steps to prepare for a grant application should be prioritized.

To prove the effectiveness of the DoYouMindful app, testing will need to be performed to examine how users interact with the app and get feedback its features. This will require a future project group to conduct a study on students with the app and conclude if the app has the potential to be used as a supplement to MBSR and mindfulness practice. Steps will also need to be taken to prepare the prototype app for such a study. Improving current features for a better user experience will more accurately reflect the capabilities of the platform in production. The ability to log user activity from the app should also be implemented to use for further improvement of the app experience and to examine trends in the content that is most commonly accessed. Additionally, demo content should be properly populated in the CMS server and organized in the app to better present the resources utilized throughout learning MBSR practice. Improved organization of content require refined content type definitions on the CMS server and updated app code to account for the revisions made.

Once funding is acquired, progress can be made in obtaining content that could be used in production. With more finalized content, the improvements that should be made to the app will become more evident. It would also be beneficial to have two teams for maintaining dedicated iOS and Android codebases of the app in the future to provide the best possible support for both operating systems.

6 Conclusion

This project provided the first working prototype for an MBSR mobile platform four years in the making. The result of this prototype attests to the feasibility of implementing the platform's envisioned features and provides a strong foundation for future expansion and development. Development of this prototype also addresses many of the challenges that can occur throughout the software development life cycle, and serves as a reference for continued development for the future.

As a personal reflection, this project has been an invaluable learning experience as I continue to study Computer Science at WPI. This has been my first exposure to mobile app development, as well as the first opportunity I've had to work with useful technologies like Docker and Strapi. The steps taken throughout the development process have challenged my ability to utilize what I have learned in the various Computer Science courses I have taken, and further instate their applicability to solving real world problems. I am grateful to have had the opportunity to work on a project with such potential to make a positive difference people's lives, and hope that I will be able to further contribute to its development in the future.

The findings of this project were presented to WPI students and faculty on November 25th, 2019 at WPI's first annual Works in Progress: Undergraduate Research Symposium. In addition to a poster summarizing the project, an interactive demonstration was held at the event to showcase the app running on an iPhone 6S and Samsung Galaxy S8. A laptop was also used to showcase the inner-workings of the back-end and the process of managing content through the CMS web portal.

Redacted

Figure 6.1: MBSR App Development poster design used at the WPI Works in Progress: Undergraduate Research Symposium

.

Bibliography

Docker (2019). The Industry-Leading Container Runtime. https://www.docker.com/products/container-runtime.

Facchini, A., Lutz, M., and Welton, H. (2015). MBSR & College Students. Technical report, Worcester Polytechnic Institute.

Falzone, R., Mahn, A., Pataky, N., and Rodriguez, J. (2017). Enhancing Mindfulness-Based Stress Reduction Through a Mobile Application. Technical report, Worcester Polytechnic Institute.

Git (2019). https://git-scm.com/.

Github (2019). https://github.com/.

Instagram (2019). Instagram User Endpoints. https://www.instagram.com/developer/endpoints/users/.

MongoDB (2019). https://www.mongodb.com/.

React Native (2019). Create native apps for Android and iOS using React. https://facebook.github.io/react-native/.

Strapi (2019). https://strapi.io/.