
# Massachusetts Sober Housing Mobile Application

**Project Team**

Jack Charbonneau: jlcharbonneau@wpi.edu

Alex Hard: adhard@wpi.edu

Kyle Savell: kqsavell@wpi.edu

Tom White: twhite@wpi.edu

Joan Wong: jwong3@wpi.edu

**Project Advisor**

Professor Wilson Wong

Department of Computer Science

The team would like to thank Bruce Feine and Nick Murphy from Rally to Recovery, as well as Troy Clarkson and Donald Flagg from the Massachusetts Alliance for Sober Housing for their support throughout the project.

# Contents

# Tables

# Figures

# Abstract

Facilities such as halfway houses and sober homes provide supportive housing for people undergoing treatment for drug and alcohol addiction. Unlike halfway houses which are modeled after dormitories, sober houses are structured like a private residence and provide many amenities to residents. The Massachusetts Alliance for Sober Housing (MASH) provides a set of standards for sober homes in the state of Massachusetts.

The goal of this project was to develop a mobile application to assist in the administration of MASH certified facilities, and to provide useful features for residents to support their personal recovery.

# Introduction

Opiate addiction has become a problem on a global scale. The opiate crisis began through overprescription of opiate pain killers and this epidemic has resulted in the death of over 33,000 Americans in 2015 alone or approximately 115 per day ("Opioid Overdose Crisis", 2018). In 2017, Massachusetts faced over 2000 deaths directly related to opiate overdose ("Data Brief", 2018).

In the early 1990s, opiate medication manufacturers informed the medical community that opiates were non addictive and a safe medication to prescribe as pain killers. This misinformation led to an increase in prescription of the highly addictive drugs. After the patients' prescriptions ran out, the opiates resulted in numerous patients becoming addicted but no longer having access to the medication. Although these patients were victim to false information about the safety of opiates, their addiction often forced them into situations that quickly spiral out of control. According to the National Institute of Drug Abuse, 80% of heroin users were prescribed painkillers originally, and there has been a 30% increase in opiate related overdoses between 2016 and 2017.

Individuals suffering from opiate addiction can get help recovering through a halfway house or a sober home; i.e. a living spaces where recovering individuals live together and promote recovery. Whereas halfway houses tend to be government-run and act like dormitories, halfway houses act as a private residence and tend to be run by sober-living experts. While all homes operate under the 'sober housing' umbrella, many of sober houses run under differing standards, such as MASH standards, independent standards, or no standards at all. Because there is a lack of universal standard, the homes face different levels of success. While the individuals

living in the sober homes come from a wide variety of backgrounds, situations, and locations, one thing is almost universal; ownership of a cell phone.

In order to better understand the effectiveness of the sober homes in a person's recovery as well as assist the person in staying sober, a mobile app was created to track the progress of recovery and give residents a place to send feedback regarding the sober homes.

To create this application, the team identified a set of goals to accomplish. First, the team investigated the needs of the residents and the sober home associations in regards to what functions they wished the application to include. Second, the team implemented those features through a series of Agile sprints. Finally, the team deployed a completed application for use by the residents and sober home association that will hopefully have a positive impact on their recovery.

# Background

The purpose of the application was to improve the quality of data collected from sober homes in Massachusetts while providing additional support for sober home residents.
A sober home is a residence for people recovering from alcohol and drug addiction where their recovery is promoted and the residents are expected not to use drugs or alcohol. These houses reside in family neighborhoods to promote a healthy environment (Gorman, Marinaccio, & Cardinale, 2010). Even though each sober home falls under this definition, every residence runs differently. Sober homes are run by an owner who makes the rules for that specific building. Thus, sober homes can range from places for peers to support one another in their recoveries to having more strict rules such as weekly meetings and curfews ("What is a Sober Home," 2016).

When discussing sober living environments, it is important to distinguish between sober homes and halfway houses. Halfway houses can be run by government agencies and sober homes cannot. Sober housing models a private residence and offers more privacy and comfort than most halfway houses, which model dormitory living and few amenities. However, because halfway houses are less structured, they tend to be less expensive than sober housing (Real Recovery 2017).

Sober living environments have existed for a while, and have come in many different forms. Sober housing originated in the mid-19th century and were houses run by organizations such as The Salvation Army and YMCA. In the 20th century, recovery methodologies such as the “12-step” program that emerged post-WWII were introduced to these housings. Today, in addition to recovery goals, many houses encourage peer-to-peer recovery and self-sustainability among residents (Maldonado 2018).

Despite the effort of local and federal governments to create these sober living spaces, there have has been much opposition. Landlords can label any residence as a “sober home”, and some have abused that fact; a known problem is that some owners convert single-family homes into residences for 20-30 people without providing recovery support in order to maximize rent. Additionally, people in these neighborhoods have been known to raise public safety concerns due to the background of the residents. These issues required government agencies to differentiate between legitimate and illegitimate sober homes and to protect sober home residents (Gorman, Marinaccio, & Cardinale, 2010).

To aid legitimate sober homes, Massachusetts has made a certification process for sober housing. Residences are able to get a certification stating they comply to national standards, ensuring they are operating fairly and legally ("MA Sober Home Laws," 2016). In Massachusetts, the Massachusetts Alliance for Sober Housing (MASH) sets the certification standards for sober housing and helps existing sober homes uphold their standards so that they can be certified. Although sober houses can operate without being certified, only certified sober homes can receive referrals from state-funded institutions like courts, treatment centers and other facilities ("Standards," 2016).

MASH has 35 standards which it uses for certification, targeting a variety of different fields such as administration, recovery and property. Accuracy of data is of paramount importance to MASH; certified sober homes are required to provide accurate financial information, keep accurate resident information, and all information about the home must be substantiated. Sober homes are required to follow fair housing laws and ensure a resident’s personal information remains private. For recovery, residents are required to have enough resources at their disposal to support them. House owners and leaders are trained in managing a

sober home and make plans tailored for the individuals in their home, where accountability and a safe respectful environment is expected. In terms of the physical house, it is meant to be as home-like as possible and act as an actual household ("Certification Standards," 2016).

Many of these certification standards are derived from Massachusetts laws regarding sober homes. Proposed Bill H.1828 states that sober homes should adhere to national standards, which are the basis for the standards set by MASH. Many of the standards require fair treatment of tenants and focus on the priority of recovery. Regarding certification, the bill makes it clear that certification can be both given and revoked by the director of substance abuse services at any time, which is why MASH continually makes sure sober homes are following their extensive guidelines which can be found in Appendix D ("MA Sober Home Laws," 2016).

# Methodology

The goal of this project was to create a mobile application that would help validate and optimize the effectiveness of sober homes. In this chapter of the report, the team discussed the setup of the team dynamic, the various development tools used to create the mobile application, and the methods employed to perform requirement gathering and designing the application.

## Software Development Approach

Software methodology is crucial to the collaboration and success of a team and the quality of the project overall. Therefore, the team considered many approaches, including the Waterfall Model, the Kanban-Agile methodology, and the Scrum-Agile framework. After discussing each of these options, the team decided to adopt the Scrum-Agile approach in developing the mobile application. In the following sections, the team gave a brief overview on each of the approaches mentioned previously.

### The Waterfall Model

The Waterfall Model consists of six stages in the software development: requirements analysis, design, implementation, testing, deployment, and maintenance. Each phase is dependent upon the completion of the previous phase.

In the requirements analysis phase, all of the features and components for the lifetime of the project are captured. Developers need to ensure that the requirements are feasible, testable, and measurable. For the design phase, the development team creates the architecture for the project, including class diagrams, activity diagrams, and any other useful reference tool to

understand the behavior or construction of the application. Any hardware requirements are documented in this phase as well.

The creation of the product happens in the implementation phase. The way that the product is created will vary between teams, but almost all will also create unit tests to assist in testing individual features of the product.

Verifying the features from the implementation phase happens during the testing phase. All of the unit tested code is integrated into the project, and additional non-unit testing is done in this phase. If there are a large number of issues discovered during the testing phase, it is common to have the project regress back into the implementation phase to redo portions of the product. After the product has been tested and the requirements have been satisfied, the team moves onto the deployment phase. Testing aspects of the program such as operations under stress, exit conditions, and backup restoration are included in this phase.

The deployment phase consists of the initial setup of the code in production machines. Typically, a user manual is produced during this step to inform the user on how to properly use the product. The final phase of the Waterfall Model is the maintenance phase.

The maintenance phase is entered after the product has been deployed. Here, software developers issue patches for the software to address lingering bugs as well as to match the changing need of the contracted organization. This phase is important for the developers because it allows them to address problems that may arise that do not warrant the expense of the project regressing back to the development or even requirement gathering phase. Bugs are filed and fixed, small improvements are made and additional features are added to satisfy user requirements, and usability is maintained. If this is a web application, developers would ensure that the servers stay running; for desktop applications, that operating system patches do not break

the product (What is SDLC Waterfall Model, 2018). For professional or contracted work, the length of developer support in this phase is normally specified in the contract during the requirement analysis phase. In open source and less traditional contract development, the time spent in maintenance is not always well defined.

Because of this "strictly sequenced" model, requiring one phase to be completed before the next, it is costly to respond to users' changing requirements. For example, if a new feature is to be added, or an existing feature needs to be changed, the development process will regress to an earlier stage, the requirement analysis phase, and then be carried through all the other following phases one at a time before actually returning to the phase prior to implementing this change in design. During the initial meeting with the team's sponsor, there were numerous ideas exchanged between both parties. This indicated to the team that the requirements for this application were highly likely to change as the project progresses to both fulfill the sponsor's needs as well as the requirement as an Major Qualifying Project (MQP) project at Worcester Polytechnic Institute (WPI). Due to the inflexibility of the Waterfall Model, the lack of communication this approach offers between the team and the sponsor, and the inability to add features and changes to the mobile application after beginning development, the team did not find this approach feasible for this project.

## The Agile Methodology

The Agile framework has gained popularity over the years due to its promotion of a flexible, transparent, adjustable-to-change model for software development. In a 2015 study, 25% of software companies indicated that they use and apply Agile techniques on a daily basis, which was a 10% increase compared to a study completed in the prior year, and 60% of the respondents indicated that they were exposed to Agile (Farvin Packeer Mohamed, 2014; APMG International, 2017). The flexibility and transparency that this model offers is based on four values: *individuals and interactions* over processes and tools, *working software* over comprehensive documentation, *customer collaboration* over contract negotiation, and *responding to change* over following a plan (Beck, 2001). These values form the basis of the twelve principles abided in an Agile community listed below:

1. "Our highest priority is to satisfy the customer through early and continuous delivery of valuable software."
2. "We welcome changing requirements, even late in development. Agile processes harness change for customer's competitive advantage."
3. "We deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale."
4. "Business people and developers must work together daily throughout the project."
5. "We build projects around motivated individuals, giving them the environment and support they need, and trust them to get the job done."
6. "The most efficient and effective method of conveying information to and within a development team is face-to-face conversation."
7. "Working software is the primary measure of progress."

8. “Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.”
9. “Continuous attention to technical excellence and good design enhances agility.”
10. “Simplicity - the art of maximizing the amount of work not done - is essential.”
11. “The best architectures, requirements, and designs emerge from self-organizing teams.”
12. “At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly” (Beck, 2001).

## The Agile Development Life Cycle

As for any software development methodology, the Agile development cycle involves the following stages: requirements gathering, planning, product design, development, release, and tracking and monitoring (“What’s the Difference”, 2018). However, it is important to note that in the Agile framework, it is not necessary for these phases to take place in sequential order; that is, multiple phases may be happening simultaneously, or in parallel, with one another. There are several methods for implementing Agile. In the next section, the team discussed two of these methods: Kanban and Scrum.

## Kanban Software Development

Kanban is the second project management approach the team examined. Kanban supports a “continuous workflow,” where the application is developed incrementally on a daily basis. Kanban is rooted in a “Just In Time” (JIT) philosophy, where the needed product is resupplied just as it runs out. This approach was traditionally utilized in supermarket and industrial businesses where the food or materials were restocked just as the current stock was about to run

out. For Kanban, this approach has been converted for software teams, where the amount of work in progress (WIP) is matched to the team's capacity (Radigan, 2018).

One of the key features of Kanban is its visualization of displaying the states of current tasks on Kanban Boards to identify any bottleneck issues. This also has been adapted from a factory-floor approach, where physical cards describing the state of a process were passed between people. On a Kanban Board, a virtual card (or a physical one for physical Kanban Boards) represents one work item and describes what the item is and who it is assigned to. Each card is placed under a section on the board such as "To Do," "In Progress," or "Done," and the task is moved to the appropriate section when the state changes. WIP limits can be applied to specific sections so that only a certain number of cards at a time can be there. Having too many cards in a section indicates there is a bottleneck in the subsequent section. Overall, the Kanban Board gives the entire team a visual representation of the flow of work, who is working on specific tasks, and where any blockages in the workflow are (Radigan, 2018).

The backlog for Kanban is the "To Do" section of the board, with each card in this section being an item in the backlog. Although there are not many predefined team roles in the Kanban methodology, there is the role of product owner. Product owners prioritize the work in the backlog however they see fit while also trying to minimize interruptions for other team members. When team members finish the item they are working on, they pull the task with the highest priority from the backlog to work on (Radigan, 2018).

As stated before, Kanban has the potential to create bottlenecks for the workflow. A bottleneck happens when one of the sections gets more work than the maximum throughput of that section. For example, if an "In Progress" section has five current work requests, but a WIP limit of four, this would cause a bottleneck. In other methodologies, it can be difficult to identify

these bottlenecks in a timely fashion. For the Kanban methodology, there are also predefined ways to handle a bottleneck. If the WIP limit of the bottlenecked section is too high, the limit can be reduced so there are fewer context switches to that section. Alternatively, more resources could be applied to the section, or the number of work items could be limited or grouped together based on similarity, to reduce the risk of a bottleneck occurring ("What is a Bottleneck", 2018).

The Kanban Board directs the team's focus to active tasks and eliminates overproduction by saving, conserving and focusing team resources. As a result, this type of project management is more responsive to changes in workload than the Waterfall Model. When the priorities for a project change frequently, this methodology performs well because of how it weighs demand against throughput. Additionally, the Kanban approach also fosters rapid feedback and shorter cycles with continuous delivery ("What is Kanban?", 2018). In the Waterfall and Agile-Scrum methodologies, cycles are determined by a preset time frame. In Kanban, a cycle is the amount of time it takes for a single task card to travel through the team's workflow and be deployed (Radigan, 2018). Because cycles are based on a single-piece workflow, they are much more flexible and changes can be made at any time, whereas in Agile-Scrum, changes should not be made once a development iteration has begun ("What is Kanban?", 2018).

The Kanban methodology can struggle in projects that span over a long period of time because tasks have no deadline associated with them. The completion and deployment of a single task is determined "at the team's discretion" ("What is Kanban?", 2018). Furthermore, if a team member is spending a lot of time on a particular problem, or work is not delegated to a certain section of the Kanban Board, development could halt and delivery could take longer than it would have under other methodologies.

## Scrum Software Development

Scrum is another variant of Agile software development. Scrum is so widely used that it makes up more than two-thirds of all Agile software methodologies used by software companies in 2013 (Retzlaff, 2013). Scrum is an "iterative software model that follows a set of roles, responsibilities, and meetings that never change," operating in a continuous cycle of delivery ("What's the Difference", 2018). In the following sections, the team discussed each of the crucial components and processes of the Scrum-Agile software development life cycle: logging, the sprint cycle, and roles.

### Logging

Maintaining a log is crucial to keeping the team on track during the development process. There are two types of logs that the team needs to manage: product backlog and sprint backlog. The *product backlog* keeps records of all the requirements and features for the application, which may change over the course of the development process, while the *sprint backlog* stores a subset of these features to be completed within the next iteration (Broggio; Retzlaff, 2013; Taymor). The sprint backlog entries, however, should not change as this may make it difficult to track the progress of the iteration.

The product backlog is initially created in the requirement gathering phase, where developers create user stories to gauge all of the different features for which the users may want to have implemented in the application. The *user stories* are notes used to identify actions and features that accomplish specific goals for specific users of the application (Taymor). The team will then record them in the product backlog and rank these features in order of estimated completion time using a 1-, 2-, and 3-point scale. The sum of the point values associated with

each item in the backlog is referred to as the *sprint velocity* and is used as a measure of the team's productivity.

## The Sprint Cycle

In a Scrum environment, a team develops the software application in fixed-intervals known as *sprints*, which are usually about 1-3 week-long iterations allowing the team to deliver software regularly. Before each sprint, the team will hold an initial meeting to discuss the project and team goals, and populate the *sprint backlog* for the upcoming iteration of the application (Broggio; Retzlaff, 2013). During the sprint, the team is also required to hold short *Scrum meetings* on a daily-basis, where members share what they did yesterday, what they are planning to do today, and what difficulties they are experiencing while completing a specific task from the backlog ("What's the Difference", 2018). After a sprint is completed, a functional deliverable will be available for testing and presentation to users for feedback.

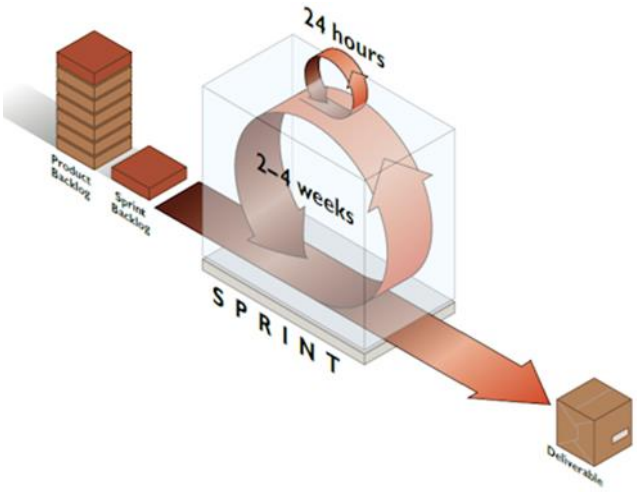



Figure 1. Scrum-Agile Life Cycle (Retzlaff).

Figure 1 above provides a graphical depiction of the interaction between backlogs and the sprint cycle. All of the items that need to be completed are laid out in the product backlog while a subset of those items is contained in the sprint backlog. The sprint backlog is fed into the sprint cycle, at which time the items in this backlog will be completed over a period of a few weeks. When the sprint is finished and the sprint backlog is complete, a new set of deliverables has been made. The cycle will start over again and continues to repeat until all items in the product backlog have been turned into deliverables.

### Roles

For self-organizing teams, additional responsibilities may need to be placed on ensuring that the team is on schedule and is pursuing the project goals. The two positions that play critical role in the Scrum-sprint development cycle for this purpose are Product Owner and Scrum Master. *Product Owner* is the "voice of the customer" (Broggio; Retzlaff, 2013). That is, being the product owner is responsible for prioritizing and assigning user stories to the product backlog, making sure that team is creating a product of user's interest. On the other hand, the *Scrum Master* facilitates Scrum meetings by "removing impediments" or issues that the team may be experiencing to ensure the delivery of sprint goals (Broggio; Retzlaff, 2013; Taymor).

## Comparison of the Waterfall Approach and the Agile Framework

An Agile approach divides the development process into short increments. At the end of each iteration, there is a functional, working deliverable to be presented with new features added. As opposed to the Waterfall Model, this allows the team to receive feedback from users and address any changing requirements they may have at the end of an iteration. This embodies the

Agile value “customer collaboration over contract negotiation” (Beck, 2001). The close interaction and project transparency of the Agile approach, such as holding daily Scrum meetings, ensures better productivity of the team. Although the Kanban-Agile framework does allow the team to view active tasks, the absence of roles, such as the Scrum Master, minimizes supervision and awareness of team status and effectiveness. Because the Scrum-Agile approach is favorable to both the customer and developers, the team chose this approach as the software development method for the project.

# Software Development Environment

## Version Control

Making use of version control ensured that the team could safely develop and test individual features with a greatly reduced risk of introducing bugs or security vulnerabilities to the production code. GitHub is the current industry standard, but other platforms such as Bitbucket and GitLab have enjoyed seen substantial growth in recent years.

All three of these platforms are quite feature-rich and have all of the tools that the team required such as issue tracking and project management tools built-in. However, Bitbucket does not have an integrated project management system and Gitlab has many added tools for business analytics that the team does not require. As a whole, the team was most familiar with GitHub and provided all of the features needed for the project. In order to facilitate and keep track of collaboration in the codebase, the team used GitHub as the version control software.

## Issue Tracking

In order to track issues and bugs within the team's code, the team made use of GitHub Issues. This option was ideal for the team because it is, by default, integrated with the GitHub repository. GitHub Issues also allowed issues to be assigned to specific contributors, which ensures that team members did not waste time fixing bugs that were already being fixed by other team members. Furthermore, the platform allowed issues to be labeled and classified in a variety of different ways. This allowed the team to stay more organized and assisted in focusing on the most critical aspects of development first.

## Project Management

Since the team used Agile, it was of the utmost importance that the team remained on task and on schedule. The team needed to keep track of iteration goals, feature development, quality assurance, bug fixes, and a number of other factors. A popular software for this type of project management is Trello. Trello is multiplatform and allows for the creation of a variety of "Boards" which each pertain to one topic such as UI, Server, Database, etc.. Each board contains a number of "Lists" which generally keep track of the stage of development that a "Card" is in. A card generally represent an individual task or feature that must be completed. Although the team was originally planned on using Trello, the team found that GitHub provided an extremely similar service, GitHub Projects, which is integrated by default with each repository. GitHub Projects has the added benefits of keeping the project management and version control software in one central location, and providing great integration with GitHub Issues that allows an issue to be made directly into a card and placed in its appropriate List and Board. For these reasons, the team decided to use GitHub Projects for project management.

## Communication

### Internal

For communication the team used Slack. All of the team members have used Slack in the past and are familiar with its features. One benefit of Slack is that it allows for the creation of multiple channels which can each be dedicated to an individual topic. This reduces clutter in the team's communications and ensures that conversations will stay organized and cohesive. Additionally, Slack can be integrated with GitHub to send messages corresponding to the repositories activities so that team members are always up to date about the latest developments.

It is also important to note that Slack is multiplatform, meaning that it can be accessed via web browser, desktop application, or mobile application. This helped to ensure that all team members were made aware of any important developments in a timely fashion.

The team also looked at other communication platforms such as Microsoft Teams and GroupMe. GroupMe was not chosen due to its simplistic layout and features. GroupMe also does not support multiple channels or GitHub integration, resulting in less organization and no Github tracking. Microsoft Teams is very similar to Slack in its capabilities. It is also multiplatform and can integrate with GitHub. The team chose not to use Microsoft Teams as it does not offer any advantages over Slack and would require the team to become familiar with a new system and install a new program on personal devices, whereas the team members were already familiar with Slack and had the program set up on their personal devices.

### External

In order for the team to conduct any external communication with sponsors, institutional boards, or research participants, the team created an Outlook Group. This group allowed us to present one email address that ensured all communications could be viewed by all team members. Similar functionality could be provided through a variety of platforms, but Outlook was the best option for the team because the team already had Outlook accounts that were used on a consistent basis.

## Platform

In the current market, there are two primary types of operating systems that cell phones utilize - Android and iOS. Both are widely accepted, used and developed for making it relatively easy for an application to be downloaded and used. While iOS and Android operate very

similarly on the user interface, development differs greatly between the platforms making it difficult to develop one application for both platforms.

The team ultimately decided to develop for Android for a variety of reasons. First, Android phones are much more common that iOS devices, with Android currently holding 88% of the mobile operating system market share as of Q2 2018 (Gartner, 2018). Statistically, developing the application on Android allows it to be available to the most people. Second, developing for iOS is more costly than for Android. In addition to requiring an expensive Apple Developer license, iOS development is centered around MacOS specific technology. To develop efficiently for iOS team members would have needed to purchase MacOS computers or make a significant time investment to set up a proper development environment in another operating system. The team members had previous experience with Android development, but not with development for iOS. This prior experience allowed the team to reduce the time needed to learn new technologies which, in turn, provided more time to focus on development.

It is possible to develop one application that can be utilized by both operating systems by using Facebook's React Native platform. This platform allows developers to write their application using JavaScript to invoke and direct the operating system's native components (Facebook, 2018). In some cases JavaScript cannot be made to implement all desired functionalities, in which case some functionality must be written in native code and then linked in the JavaScript application. In cases where an application must be developed for both Android and iOS, React Native can save development time by allowing much of the applications code to be written once and used on both systems.

In the end, the team decided to develop solely for Android. This decision was heavily influenced by the fact that Android has the majority market share in mobile operating systems

and the team would not need to incur costs from buying developer licenses. Additionally, the team did not want to develop for both operating systems using React Native due to the inexperience with the platform.

## Programming Language

There are three main programming languages used in Android Development: Java, Kotlin, and React Native. Each of the languages have benefits and disadvantages, which made selecting the development language a challenge. After numerous discussions the team decided to use Java.

### Java

Java was chosen as the development language due to its extensive documentation, use within Android's systems, and the team member's development experience. Java was first introduced in may of 1995, over 23 years ago (Binstock, 2015), and has become a popular development language. Due to Java's popularity, the documentation for Java and the knowledge shared between developers is extensive. On a popular programming knowledge website stackoverflow.com, there are currently over 1.4 million posts regarding the Java programming language(Stackoverflow, 2018). This extensive knowledge base was beneficial to the team as it assisted with debugging and implementing new features into the application.

Further the Android SDK is written in Java and makes development and integration with the Android system easy and straightforward (Android Studio, 2018).

Finally, the team members had experience with creating full Java applications through an Agile methodology from completing a software engineering course at Worcester Polytechnic Institute. This previous experience allowed the team to begin development quickly without the

need to learn a new language. The familiarity with the language also allowed the team to implement more advanced features in the application leading to a more robust final product.

### Kotlin

Kotlin, while an official programming language of Android as of 2017, was not chosen because of its relative lack of documentation and the team having no experience with the language (Android Developer, 2018). Kotlin was first released in 2016 making it one of the newest mainstream programming languages available (Breslav, 2016). As of September 2018, stackoverflow.com has only 14 thousand posts tagged to the Kotlin language (Stack Overflow, 2018). The team discussed concern that bugs with kotlin would be more difficult to patch due to the relatively small development community and existing documentation. The team also decided against using Kotlin because they had no previous experience developing in the language and would have had to cut into development time to become familiar.

### React Native

React Native is a framework created by Facebook to allow developers to create Android and iOS applications in JavaScript. React Native is particular useful when an app must be developed to support both Android and iOS since the majority of the code would be cross-compatible between both operating systems (React Native). There are, however, some situations in which React Native does not supply all of the developer's desired functionality. In these situations, native code must be written for each system and linked to the JavaScript code. Additionally, React Native generally performs a worse than native code, which can be important when creating resource intensive applications. The team decided against using React Native in

order to avoid the necessity of linking JavaScript and Native code, and because the majority of team members were inexperienced with regards to React or JavaScript in general.

Because of Java's large development community, extensive documentation, Android support, and the team's familiarity with the programming language, it was chosen as the development language for the project.

## Integrated Development Environment

When programming in Java, it is important to use an Integrated Development Environment (IDE) to maintain organization between numerous files in comparison to a simple text editor. Because of Java's large development base there are many IDEs available. For Android development, two of the most popular IDEs are Android Studio and Eclipse. For this project the team elected to use Android Studio.

Android Studio is an IDE developed by Google and JetBrains and is currently the official development environment for Android (Android Studio, 2018). The team chose to use Android Studio because of its Android specific nature and features. Android Studio receives regular updates and fixes making it the most current IDE for Android development with support for numerous Android versions. Android Studio also comes equipped with an Android Emulator, making testing for different devices easy and quick. Android Studio is based off IntelliJ, another popular Java IDE with which all team members have previous experience (Android Studio, 2018).

Eclipse is an IDE that was formerly the official IDE of Android. Android development for Eclipse was done with the Android Development Tools (ADT) plugin. The ADT was deprecated in 2015 in favor of Android Studio and has not received any updates since then

(Eason, 2015). Because Eclipse is no longer officially supported for Android development, the team elected not to use it.

Because Android Studio is supported by Google as the official IDE for Android, and Android Studio contains Android emulators to streamline testing, the team elected it as the IDE for the project.

# User Interface Mockups

## Adobe XD

The team chose to use the recently released Adobe XD program for user interface (UI) Mockups. XD was chosen due to quick learning curve, easy integration of Android standard UI, and its ability to link mocked up scenes together to create a mock app. XD uses a simple drag and drop interface to create individual app screen mockups. The resources to make the mockups can either be imported from a standard UI library or created by the user. The program also utilizes a simple connection system to link UI elements together, allowing the team to test UI flow without needing a full front end to be created. XD also allowed the team to develop the UI mockups on a computer and load them on mobile devices to test the UI on different size devices. Finally, XD is available at no cost.

### LucidChart Android Mockup Tool

LucidChart also offers a UI mockup tool, however the team chose not to use it due to its online nature, limited functionality, and paid subscription. LucidChart has a limited free version, which prevented full development of the UI without paying a subscription fee. Furthermore, with the software being hosted on a website, performance can become slow when a large number of objects are present. Because of these reasons the team chose not to use LucidChart.

Because Adobe XD is free, has greater functionality, and can easily be displayed on a device, the team chose XD as the UI mockup software for the project.

## Database Management System

In the software development field, there are a number of relational database competitors vying for market share. The top ten competitors are Oracle DB, MySQL, Microsoft SQL Server, PostgreSQL, MongoDB, IBM DB2, Redis, Elasticsearch, Microsoft Access, and SQLite ("DB-Engine Ranking", 2018). Competitors, such as Microsoft SQL Server, IBM DB2, and Microsoft Access are relational databases, but were excluded from this analysis because the team was unfamiliar with them. MongoDB, Redis, and Elasticsearch are non-relational databases; instead they store data in key-value pairs, which does not lend itself for how the team modeled the data. Therefore, the relational-databases that the team considered for this project were: MySQL, Oracle DB, SQLite, and PostgreSQL. Though these are all relational database, they were all designed with different paradigms in mind which resulted in certain features being better supported than others in a DBMS. The features that are emphasized or neglected were what ultimately informed the team's decision on which database to use. The team's concerns were with speed, potentially complex queries, and data security. The team utilized a server running Linux, so the only considered database systems had to be compatible with Linux.

### MySQL

MySQL is a common solution for web applications. Its continued popularity over its lifetime has led to a mature tool in the form of MySQL Workbench, which offers graphical interfaces designing, implementing, administering, and migrating databases ("MySQL Workbench & Utilities", n.d). MySQL aims to be very efficient with input/output operations per second(IOPS). MySQL has adopted a high percentage of the SQL standard. This DBMS generally has the highest market share for usage with websites in the Alexa top 1 million visited

websites, and was a strong competitor for use in the project (Database Market Share, 2018). Where MySQL falls short is with security. When MySQL was built, the developers didn't include support for end-to-end encryption with SSL certificates. For the type of data that the team stored and the network setup that the team had, this is a feature that was very important to the project. This feature currently does exist in MySQL, but requires plugins and changing a number of settings, which would have led to difficulty in setting up security correctly.

### Oracle DB

Oracle DB is a closed source DBMS developed by Oracle. It was developed as an enterprise solution for large amounts of business data. Oracle DB has a community/'express' edition (XE) but primarily markets its enterprise solutions. Oracle DB has a low SQL standards compliance and can require specialized knowledge to work with. The time investment required to properly configure the database did not lend itself for use in this project.

### SQLite

SQLite is a lightweight database engine. An attractive feature of SQLite was the ability to create and interact with 'in-place' databases. These databases are stored purely in memory. SQLite also offers a traditional storage mechanism on-disk within a single file. These features and their implementations are useful for local embedded databases, as embedded environments normally have very few resources to spare. SQLite does have a high average limit for how many input/output operations can be performed per second at ~50,000/s ("SQLite Frequently Asked Questions", n.d.), and size of data storage at 150 Terabytes, but it does not have the features that a full fledged DBMS has. Some of the features that would have been useful to the project would have been triggers (stored subroutines run when a rule is matched), a wide array of data types,

and data replication, which could be used for read-only operations during periods of high traffic. Also, SQLite is designed as a system around a single database, unlike the other options mentioned that offer the ability to manage multiple databases under the same server instance which can be useful to further segregate and secure data (“About SQLite”, n.d.).

### PostgreSQL

Postgres offers a similar default start that optimizes queries without much fine tuning (Shaughnessy, 2014). Postgres is commonly used in research and web development environments. The main features that highlight Postgres’s use with web applications include support for SSL certificates, single click synchronous or asynchronous replication, and horizontal table partitioning. Out of the various DBMSs discussed in this section, Postgres has the highest SQL Standard compliance (“About PostgreSQL”, n.d.), which is why the team chose to use PostgreSQL.

## Feature Comparisons

| | MySQL 8.0 | OracleDB XE 11g | SQLite 3.25 | PostgreSQL 10.5 |
|---|---|---|---|---|
| Multiple Databases (DBMS) | ✓ | ✓ | | ✓ |
| Asynchronous Data Replication | ✓ | ✓ | | ✓ |
| Synchronous/Semisynchronous Replication | ✓ | ✓ | | ✓ |
| Maximum DB Size | Unlimited | 4 GB | 140 TB | Unlimited |
| Materialized Views | | ✓ | | ✓ |
| Column Level Actions | | | | ✓ |
| Custom Data Types | | ✓ | Partial | ✓ |

Table 1. Database Feature Comparison

### Data Replication

There are two schemas for data replication: Asynchronous and Synchronous which the team will refer to as Master-Slave and Mirrored respectively. Master-Slave replication is when a single database instance is the ‘master’ that is directly interacted with read/write operations. When a write operation is made on the master database, it sends the update to the ‘slave’ replicated databases but does not wait to confirm that the data gets updated correctly. This frees up the master to perform another action, but it cannot be sure that the slave is at perfect parity with it. The slave database can be used as read only databases in high load situations to not slow down write operations on the master database. There is a danger with these duplicate databases as they may not be perfect clones of the master database. The advantage you get in exchange for this is the potential for a faster read operation under heavy load. This type of replication is asynchronous (TechTarget, 2015). The other type of replication is a Mirrored setup. In this type of replication, all of the databases can be used as read/write clones of the original database. When one instance gets written to, it makes a transaction with the other databases and waits for the transaction to finish before moving on to another task. This setup guarantees that if one database become corrupted, another can replace it seamlessly, at the trade-off of waiting for verification of every write operation (TechTarget, 2010).

### Materialized Views

Views are a mechanism to create pseudo-tables or to join together data from many tables into a concise display. Views can be interacted with for read operations as if they were real tables. The difference between materialized and transient views is when the data is fetched. Materialized views execute the SELECT statement when data from the involved tables is

updated.  Transient views run the SELECT statement every time the view itself is queried (“40.3 Materialized Views”, n.d.).

## Security

The user base consisted almost entirely of sober home residents and recovering addicts. The data that they provide may be extremely sensitive and personal in nature, and as such it was imperative that the team take precautions to ensure that their data was securely transmitted and stored. This requires attention to detail when the user inputs information into the app, what is stored on the local device, what is sent to the application server, how the application server and database server communication, and how the database server audits access to the data store.

### In the App

The team stored as little data as possible in the local application. The team used the standard storage APIs on Android to store preference settings in what Android calls 'internal storage'. Internal storage is app specific and cannot be read by other apps. This effectively sandboxed the application. Sandboxing, in a mobile context, is the process of artificially separating or compartmentalizing apps, which effectively protects the app's data from other, potentially malicious, apps ("Data and File Storage", 2018).

### Over the Wire

When storing data on a remote server, it was imperative to secure the information being transferred. To this end, the team used Transport Layer Security (TLS) 1.3 for end-to-end encryption between an instance of the app and the server that stores the data. End-to-end encryption is an encryption scheme where no central authority is able to decrypt the messages being sent. The only people that can decrypt it are those who are intended to receive the message (Saltzer, J. H., Reed, D. P., & Clark, D. D., 1984). TLS is the protocol HTTPS that websites use

to secure communication with the browser. TLS is a standard that grew out of the Secure Socket Layer (SSL) protocol, which was discontinued due to numerous exploits found since popular adoption of the SSL 2 protocol. Both the SSL and TLS protocol detail a scheme that involves both symmetric and asymmetric encryption in order for the client and server to agree on a shared secret code that will be used to encrypt and decrypt messages securely. TLS is based on SSL 3.0, but was designed differently enough to warrant a new standard, instead of a change to the existing SSL protocol. The team will specifically be using TLS 1.3 because it is the latest released standard of the protocol. TLS 1.3 is described in RFC 8446 (McKinley, 2003).

### Within the Server

Non-application connection sessions were done explicitly through Secure Shell (SSH). SSH is a remote access protocol that encrypts data as it is passed between the server and the client. Password login will be disabled for the server and RSA 3072 keys will be required to authenticate. RSA is an asymmetric cryptographic system that uses pairs of keys to encrypt and decrypt data. The 3072 refers to the length of the key that will be generated; in this case a 3072 bit key will be generated. The team chose to use 3072 as the key length as it was currently regarded by the NSA and National Institute of Standards and Technology to be the minimum length for security (National Security Agency [NSA], 2016). Each project member generated a key pair and used it to prove their identity to the server before starting every session. By taking these measures, making unauthorized access more difficult, and have an audit log of who connects to the server. These steps assisted security during development, but could be impractical when the application is handed over to MASH for continued maintenance. Specifically the use of SSH with a 3072 key length may be difficult to enforce and support, so simple password authentication was enabled at the completion of the project.

## Application Server

The application server was run under a service account without root privileges. Root privileges, commonly referred to as just ‘root’, is the Unix equivalent to having Administrator privileges in Windows. Root is also commonly the name of the superuser account on a Unix-like operating system, essentially allowing the root user to have access to all features and commands on the system. Running the application server with restricted privileges follows the security principle of least privilege (PoLP) where programs or users are only given a level of permissions that allows them to complete the task they are assigned to do (National Institute of Standards and Technology [NIST], n.d.). The application server did not need to modify operating system files or other users’ data, so it did not need root privileges, and therefore was not given any root privileges. This server was the way that the application was able to retrieve data from the database. All statements run by the server were prewritten, known as prepared statements, in order to prevent an SQL injection. An SQL injection (attack) is where a user is able to manipulate a form of input to take an action on the database that was not intended by the developer. Normally this results in data being compromised to the attacker or destruction of data (Open Web Application Security Project [OWASP], 2010).

## SQL Server

The SQL server was run under a service account without root privileges. Communication between application server instances and the SQL server was end-to-end encrypted using UNIX (domain) sockets. UNIX sockets are a mechanism for interprocess communication. This allows messages to be sent between different process running on the system without the need of an intermediate temporary file. By using UNIX sockets, the team avoided binding to a network port,

further reducing the risk of unintended access. The SQL server only served queries originating from the application server to prevent unauthorized access.

# External Resources

For the mobile application, there were two key features that required the use of external resources, including the timeline feature and the Facebook feature. In the following sections, the team discusses the libraries, Android Timeline View,  Android View Animations, Konfetti, and the Facebook SDK used to implement these two features, respectively.

## Licensed Libraries

For the mobile application, the timeline feature allows users to be able to track their own progress on personal goals, which in turn is a way to encourage their recovery. To facilitate the implementation of this feature, the team decided to use an external Android Application Package (APK) named Timeline-View (Asri, 2018). An *APK* is package file used for distributing and installing an application. This file is created from the compilation of resources, code, and the *AndroidManifest.xml* file of the application (Phillips, 2017). The manifest file is an XML file containing metadata of your application to the Android operating system. This library uses RecyclerView, a UI component in Android that allows developers to display a list of objects, in its implementation of timeline events display (Asri, 2018), which the team was already familiar with from previous iterations of development. Therefore, it would be easier to design the application where the team has some kind of knowledge on the setup and functionality of the new library resource.

Furthermore, to add interesting animations to some of the features in the application, the team incorporated two external APKs - Android View Animations and Konfetti (Daimajia, 2017; Segijn, 2018). The Android View Animations library was used to alert the user of any updates from the server, such as notifying the user for any surveys to complete, and any actions taken by

the user, such as changing the status of timeline events. Konfetti is a Kotlin library that offers a confetti particle system (Segijn, 2018), which was used to celebrate any achievements made by the user in the timeline feature, such as meeting a goal. Both of these external libraries were selected for their simplicity and easy-to-use functionality, allowing the team to easily implement the effects needed to improve existing features while making the application more user-friendly.

## External SDK

Another feature involved integrating social media into the application, such as Facebook, allowing users to make posts to their personal page. This functionality was enabled through the use of Facebook's provided Software Developer Kit (SDK). A *SDK* is "a set of development tools required to develop applications for Android platform" ("What is Android SDK", 2018). Facebook SDK contains six other component SDKs for Android development, which developers can download separately to save space (Facebook, 2018). For example, for the purpose of the application, the shared SDK would be sufficient to make a post.

# Research

Many mobile applications have been developed over the years to assist recovery in sober homes. The team researched some existing mobile products, and examined their features and the opinions of their user base as reference for the development of the application. In the following sections, the team discusses the applications found in the areas relating to sober home management, group recovery, and individual recovery.

## Management Applications

The most relevant application on the market for sober home management is the Sober Living App by Behave Health. This application features macro and micro management for all different aspects of managing sober houses. It allows owners to be able to manage multiple houses, keeping track of the overseer, the category of residents, and the overall occupancy. The app also allows owners to keep track of individual information for each resident in a home (Sober Living App, 2017).

There are a few of design features that were used for inspiration for the project application. The color scheme for the app is mostly white with occasional use of desaturated colors, making icons and text very clear. Information for the houses and residents are displayed in rows with tabs at the top of the screen to switch contexts. There is also a dashboard which acts as a homepage, where owners can quickly track all of the overarching views for a house. Lastly, the Sober Living App allows owners to set schedules, drug tests and notes for each individual patient (Sober Living App, 2017). This app has a very clean and efficient design, and is something that was considered for the team's application.

## Group Recovery Applications

SoberGrid is an iOS and Android application that offers users the opportunity to reach out to the people who are also in recovery. Users are able to find, chat with, or meet other nearby users using the GPS capabilities on their phones. SoberGrid's user base has shown a positive attitude towards its on-demand connections with others. This application also allows users to remain anonymous to make them feel comfortable and be more willing to share their experiences, offer support to others, and make posts through a newsfeed feature. Lastly, SoberGrid uses color boxes in users' profiles to indicate a specific need or help to the user (Sober Grid, 2017).

Developer CaredFor has created numerous version of an application for private addiction rehabilitation organizations. These applications contain features such as a progress tracker with motivational messages, as well as a forum for other members of the recovery center to community, discuss, share milestones, and provide support for one other.

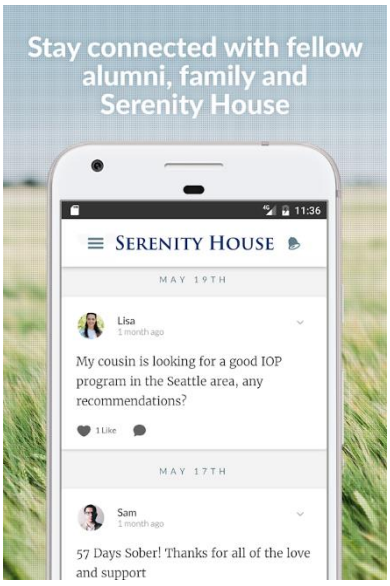



Figure 2. Serenity House Health (Serenity House Health, 2018).

The application allows for alumni of the recovery center as well as staff to post and monitor the open forum. The app is also invite only, providing privacy and security to those who would not like to share their recovery with the public.

## Individual Recovery Applications

There are a wide variety of apps aimed at a solo sobriety program. Some of these include ‘Squirrel Recovery; Addiction’, ‘Quit That’ (Horton, 2015), ‘A-CHESS’, and ‘iPromises’. Each app varies in the medical knowledge that went into them, as well as different features, which results in a wide range of quality. However, there are certain features that seem to be prevalent in many of them.

The most common feature that was found among self help apps is motivational messages. The app will display or give the user an option to display a motivational message, which tries to encourage them and prevent a relapse. Some apps have a daily message that is displayed on first use of the app per day, whereas others give the user a section where they can go to read short motivational messages or a longer motivational article.

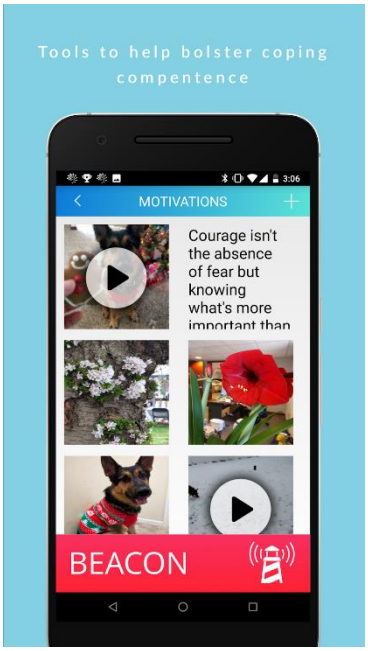



Figure 3. CHESS Health (CHESS Health, 2018).

Another common feature with these apps is the ability to add contacts as a support group. The apps that include this allow for the user to send updates to these contacts on their progress; some even allow for an emergency 'panic button' which alerts the support group that the user needs help because they are having a relapse.

Of the individual recovery apps, there are a couple that stood out in terms of features. The "I Am Sober" application is primarily focused on tracking progress rather than providing support. The app allows users to input goals, milestones, and daily pledges that can be used to track progress one day at a time. The app also offers support for more than one addiction which can be useful since the recovery processes for two different forms of abuse may not be identical. One of the more interesting features of this application is its Sobriety Calculator. This feature attempts to estimate how much money a user has saved by staying clean/sober and uses this information to further encourage positive decision making (I Am Sober, 2018).

recoveryBox is an iOS application that allows users to track their progress throughout their recovery. The app benefits from a simple design that allows users to log their daily activities which are then designated as Green (good), Yellow (Warning), or Red (Bad). Days are also designated as either green, yellow, or red based on what activities were recorded for the day. Long term progress can be viewed in a calendar format in which the calendar square for each day is filled with the green/yellow/red indicator for that day. The recoveryBox app also promotes accountability by implementing a feature that allows daily logs to be sent to friends, spouses, or sponsors. This helps to ensure that good days are properly celebrated and that support is available on bad days (recoveryBox, 2014).

# Software Requirements

The requirement gathering process of this project was completed through surveys and interviews. Surveys were given through an online survey application, and interviews were administered to Rally 2 Recovery and MASH representatives.

## Survey

The surveys were done through Qualtrics, an online tool used to design and analyze surveys. Online surveys were used since they take much less time to create and distribute, are generally more accessible and are better for maintaining privacy (Singh, Taneja, & Mangalaraj, 2009). While traditional paper surveys may be able to target a wider audience since not everyone has access to the internet, the time saved by the quick response time and the built in data analysis tools made an online survey a better option for this project.

Qualtrics also eliminated design problems that online surveys tend to have. With online surveys, it is important to keep track of the data that a participant has submitted and to limit participants to one response each. Qualtrics already does these things automatically, so less time was spent on these facets of the survey. There are also problems that Qualtrics addresses indirectly through the design tools it provides. When creating a survey, for non-skippable questions an error message is needed if the participant attempts to continue the survey without filling in a required field (Singh, Taneja, & Mangalaraj, 2009). Using the question-designing tools in Qualtrics, each individual question could be non-skippable or tailored to any format the team needed.

Qualtrics is able to provide detailed analysis of the questions asked, but the relevance of the data was achieved through careful construction of the survey questions. It was important that questions were only added if they helped get the information that was sought after. The questions focused on the sole topic of the survey, and had clear language that anyone can understand (Dunleavy, 2017)("Surveys 101", 2018). The types of questions also matter; avoiding yes/no answers was optimal as a response scale provides greater depth of answers. The survey was also short, since people are not as likely to participate in longer surveys (Dunleavy, 2017). Most importantly, questions were catered to the target audience and the knowledge of this population was kept in mind ("Surveys 101", 2018).

Since the project involved human subjects, Institutional Review Board (IRB) guidelines were followed. This meant that a user could not be forced to answer a question if they did not want to which is especially important given the sensitive nature of the project's topic. In the survey, allowing a participant to skip questions while still providing a message describing which questions were skipped in case the participant accidentally missed one was implemented through Qualtrics (Singh, Taneja, & Mangalaraj, 2009). Alternatives for questions, such as a "choose not to answer" option, was another way this issue was solved (Dunleavy, 2017). Safeguards were also required to ensure the confidentiality of the participant's information (Singh, Taneja, & Mangalaraj, 2009). While the online data on Qualtrics provides a degree of privacy, little personal information about the participants were gathered and the information that was gathered was categorized by data ranges to ensure anonymity.

In terms of design, online surveys should be attractive so participants are not put off and the color scheme should be simple while still making important elements clear. Qualtrics has a clean and attractive design by default, but can be edited manually to fit the theme of the survey.

The team used the default Qualtrics design since it was suitable enough for the survey. Layout can also be an important factor in terms of the success of the survey; single-page surveys gives the user a greater sense of how much they have completed of it, but due to the scrolling requires more effort from the user. Alternatively multi-page surveys do not require scrolling but generally take longer to complete and there is not an indicator of progress (Singh, Taneja, & Mangalaraj, 2009). Due to needing participant consent at the beginning of the survey, the survey for this project was multi-page, starting with the consent script. Qualtrics has the ability to add a progress bar for multi-page surveys so the respondent can track their progress.

In order to distribute the survey there are multiple strategies that can be used. Solicitation can be done through email, either through individual distribution or through mass-emails. If done correctly this method can be relevant and simple if the email is not oversold. However, mass-emails tend to come across as impersonal and garner low response rates. Another option is to do solicitation via associations. Because the project is sponsored by MASH and Rally 2 Recovery, surveys were distributed to both organizations to distribute to the target audience. While this was more likely to get the target population of respondents the team are looking for, solicitation through this method has a history of lower response rates than emails. Since web surveys themselves tend to have lower response rates than traditional surveys, a combination of both solicitation methods could have been used to maximize the response rate (Singh, Taneja, & Mangalaraj, 2009).

## Interview

Another data collection method that was implemented was an interview. This method is useful for gathering more personalized data and information about the underlying factors of an

issue (Madziwa, 2016). They are helpful when a point needs to be clarified for the participant, as this can be done while in conversation (Alshenqeeti, 2014). Interviews are also helpful for a limited number of respondents, and they typically provide more meaningful information than surveys (Madziwa, 2016). Most of the personal data collection that was conducted was with a small subset of staff at MASH and Rally 2 Recover, making interviews the optimal method in this case.

There are many types of interviews that could be used. One type is a structured interview, where questions are similar to that of a questionnaire and little freedom is given to the interviewee's response. Another type is an unstructured interview, in which the interview is more like a conversation where key topics are discussed and the interviewee is encouraged to elaborate on answers. A third type is the semi-structured interview, where a degree of freedom is given to the interviewee and the interviewer's goal is to touch on specific topics to get the required information (Alshenqeeti, 2014). This project used a semi-structured approach since the team needed information for specific areas to understand the clients' requirements and depth to the answers.

Unfortunately, there are a lot of negatives to using interviews for collecting data. Due to the social nature of this method, the interview and the resultant data is only ever as good as the interviewer themselves. Thus, interviews tend to to be subjective, and the interviewer's bias and worldview affect the data. Most relevant to this project is the fact that interviews take a long time to conduct; it can take a long time to set up interview appointments as well as to transcribe and analyze the data (Madziwa, 2016). Given the small sample size the team targeted for social data and the need to understand the underlying issues, interviews were the best option for this kind of data.

Like surveys, interviews are also subject to IRB guidelines. Thus, although interviews gather more personalized data, the data also needed to be anonymous. Due to this, no questions regarding personal information or that may allude to a participant's identity were asked. To identify the data that is collected, anumeric identifiers were used instead of names.

## Epics & User Stories

**Epic - Accounts**

As a RESIDENT I want to [create an account] so that I may [save my data in the app].

As a RESIDENT I want to [login to an account] so that I may [access my profile, timeline, and chat functionality].

As a RESIDENT I want to [keep my account private] so that I may [prevent others from viewing sensitive information].

As a RESIDENT I want to [add a profile picture] so that [others may know who I am].

As a RESIDENT I want to [publicize my account] so that I may [share my progress with others].

As a RESIDENT I want to [revise my account information if it is rejected by a sober home] so that I may [fix any errors].

As a HOUSE OWNER I want to [have a separate tier of account] so that I may [perform supervisory functions for my houses].

**Epic - Timeline**

As a RESIDENT I want to [view my timeline] so that I may [see my recovery progress].

As a RESIDENT I want to [create a custom event] so that I may [share personal progress].

As a RESIDENT I want to [set a goal] so that I may [remind myself what I am aiming for].

As a RESIDENT I want to [make goals and events public] so that I may [show others how I am progressing].

As a RESIDENT I want to [have a count showing how long I have been sober] so that I may [be motivated by the progress that I have made].

As a RESIDENT I want to [comment on other residents' timelines] so that I may [interact with fellow residents and give them encouragement].

As a HOUSE OWNER I want to [approve of resident events] so that I may [verify or deny whether an event actually took place].

**Epic - Social**

As a RESIDENT I want to [chat with other people who are recovering] so that I may [encourage others and be encourage by others].

As a HOUSE OWNER I want to [chat with other house owners] so that I may [share and get helpful tips for running a house].

As a HOUSE OWNER I want to [send announcements to the residents of my house] so that I may [update residents with events].

**Epic - Data Collection**

As a HOUSE OWNER I want to [explain why someone was kicked out of a sober home] so that I may [justify why a resident was removed].

As a HOUSE OWNER I want to [create a sober home in the application] so that I may [add residents to it].

As a HOUSE OWNER I want to [approve new residents] so that I may [regulate who joins the application].

As a APP OVERSEER I want to [collect basic data from residents] so that I may [know how best to support them].

As an APP OVERSEER I want to [collect information through periodic polls] so that I may [know the state of the residents across houses].

As a RESIDENT I want to [opt out of sensitive questions] so that I may [be comfortable with the information that I am sharing with the application].

As an APP OVERSEER I want to [compare data between sober homes] so that I may [know which sober homes are doing better and why].

As an APP OVERSEER I want to [access data in usable formats] so that I may [perform further analysis].

As an APP OVERSEER I want to [see visualizations of the data] so that I may [get a quick overview of the data].

As an APP OVERSEER I want to [display percentages of goals met] so that I may [get an overview of the data].

As an APP OVERSEER I want to [see why residents were kicked out of a sober home] so that I may [know if the sober home owner is acting fairly].

As a RESIDENT I want to [rate the quality of my sober home] so that I may [let other people looking to be residents know the quality of the sober home].

**Epic - Resident Support:**

As a HOUSE OWNER I want to [remove residents from a sober home] so that I may [clear space if a resident has graduated or remove people that are disruptive to the sober home].

As a RESIDENT I want to [see where local sober homes are located] so that I may [know which sober homes I can go to].

As a RESIDENT I want to [access emergency info] so that I may [call for help if needed].

As a RESIDENT I want to [find local Narcotics Anonymous meetings] so that I may [know where to find local support groups].

As a RESIDENT I want to [sort local sober homes by certain criteria such as gender] so that I may [know which sober homes I am eligible for].

As a RESIDENT I want to [report incidents to a higher authority] so that I may [prevent negligence or injustice].

## Functional Requirements

Functional requirements are requirements based on what the client wants. These can be conscious requirements, which are features that the stakeholders of the application deem necessary. There are also unconscious requirements, which are features not specifically stated by the client but will be needed by them later on. These requirements can also be features that the client did not know could be possible but that would be a benefit to the application (Futrell, 2002). For the application, the functional requirements were:

- An account system with multiple tiers for residents, sober home owners and MASH representatives
- A timeline that allows users to document their journey through recovery
- A comment/like system to allow residents to contribute to each other's timelines
- Ability for sober home owners to approve resident information for their sober home
- Ability for residents to rate and file complaints for sober homes
- An emergency button that can anonymously be used if the sober home is not safe
- Feature that allows users to see and filter local sober homes
- Geographically match residents with local NA meetings

## Nonfunctional Requirements

Nonfunctional requirements are requirements that are not requested by the client but are necessary for the operation of the application. These requirements relate to performance, scalability, security, maintainability among other non-user features. For the application, the nonfunctional requirements were:

- Database should be able to handle up to 1000 users.

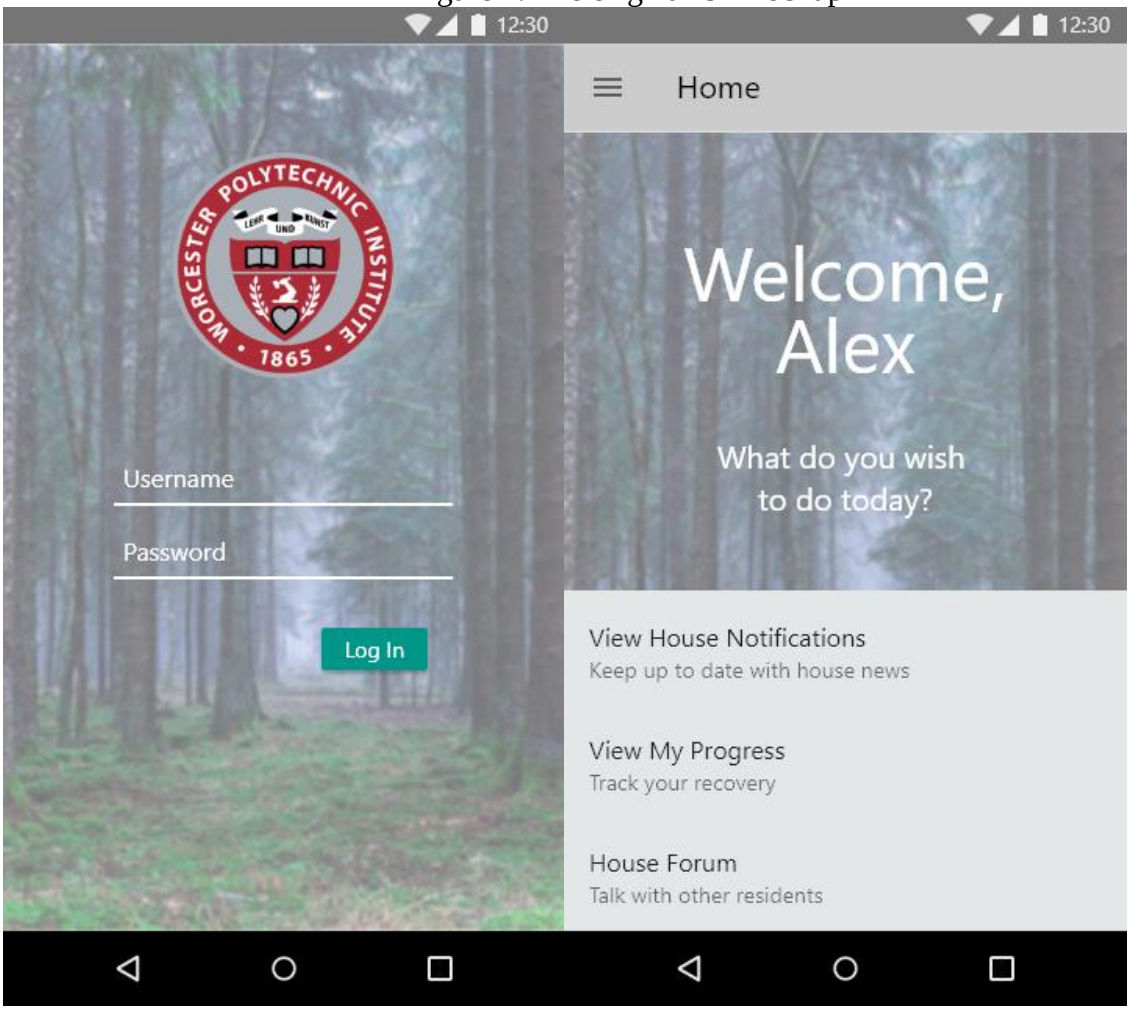
- Database should be secure and not be susceptible to injection attacks.
- Requests to the database should be executed in a reasonable time (10 seconds maximum)
- Network connection to the server should be secure using SSL.
- The application should work for all Android devices.
- The application should be tested using the Android emulator.

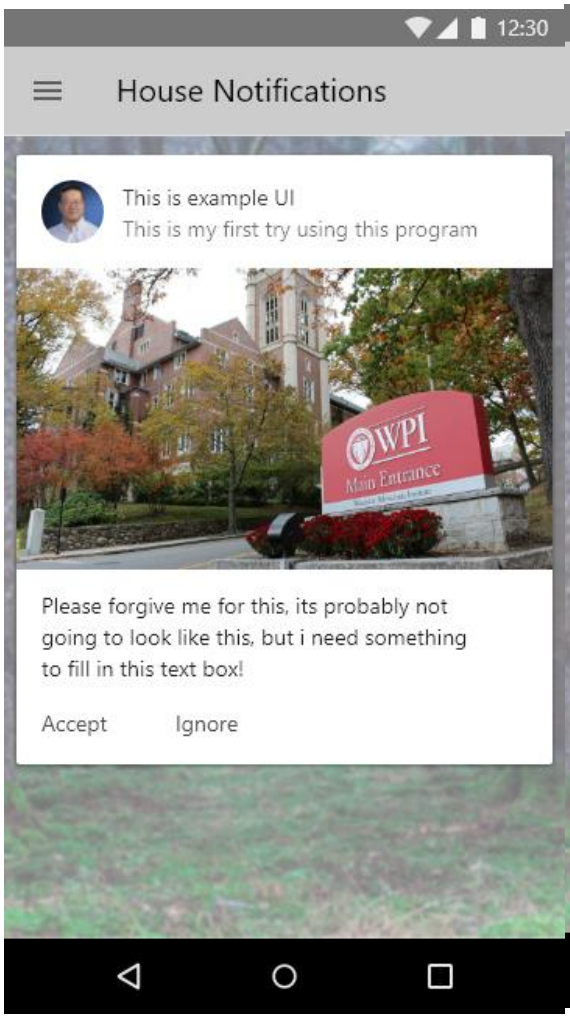
# Design

As mentioned previously, the User Interface (UI) of the application was first mocked up in Adobe XD before being created in Android Studio. Initially the UI was designed with no guidelines and was strictly a mockup to demonstrate the potential design of the application. It created a vision for the team to follow. The initial UI in the application was basic and purely for functionality. It lacked proper constraints to work on all phone screen sizes, appropriate color palettes, and Material design recommendations. The final version of the application UI followed Material color design, reformatted registration for a more user friendly experience, and overall had a better feel. The following sections show the changes in the application UI.

# UI Mockups

Figure 4. The original UI mockup




12:30
House Notifications
This is example UI
This is my first try using this program
WPI
Main Entrance
Please forgive me for this, its probably not going to look like this, but i need something to fill in this text box!
Accept
Ignore

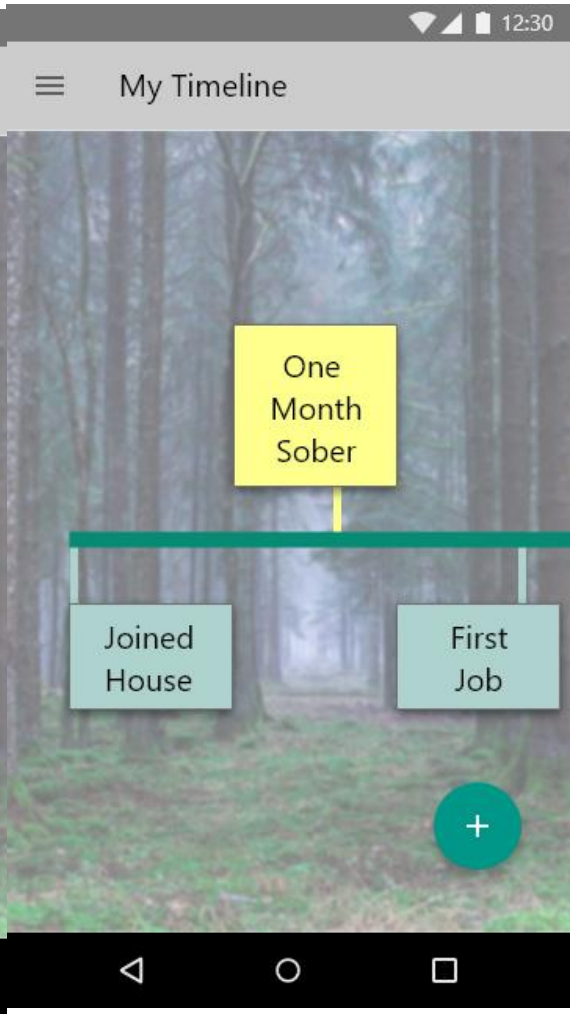
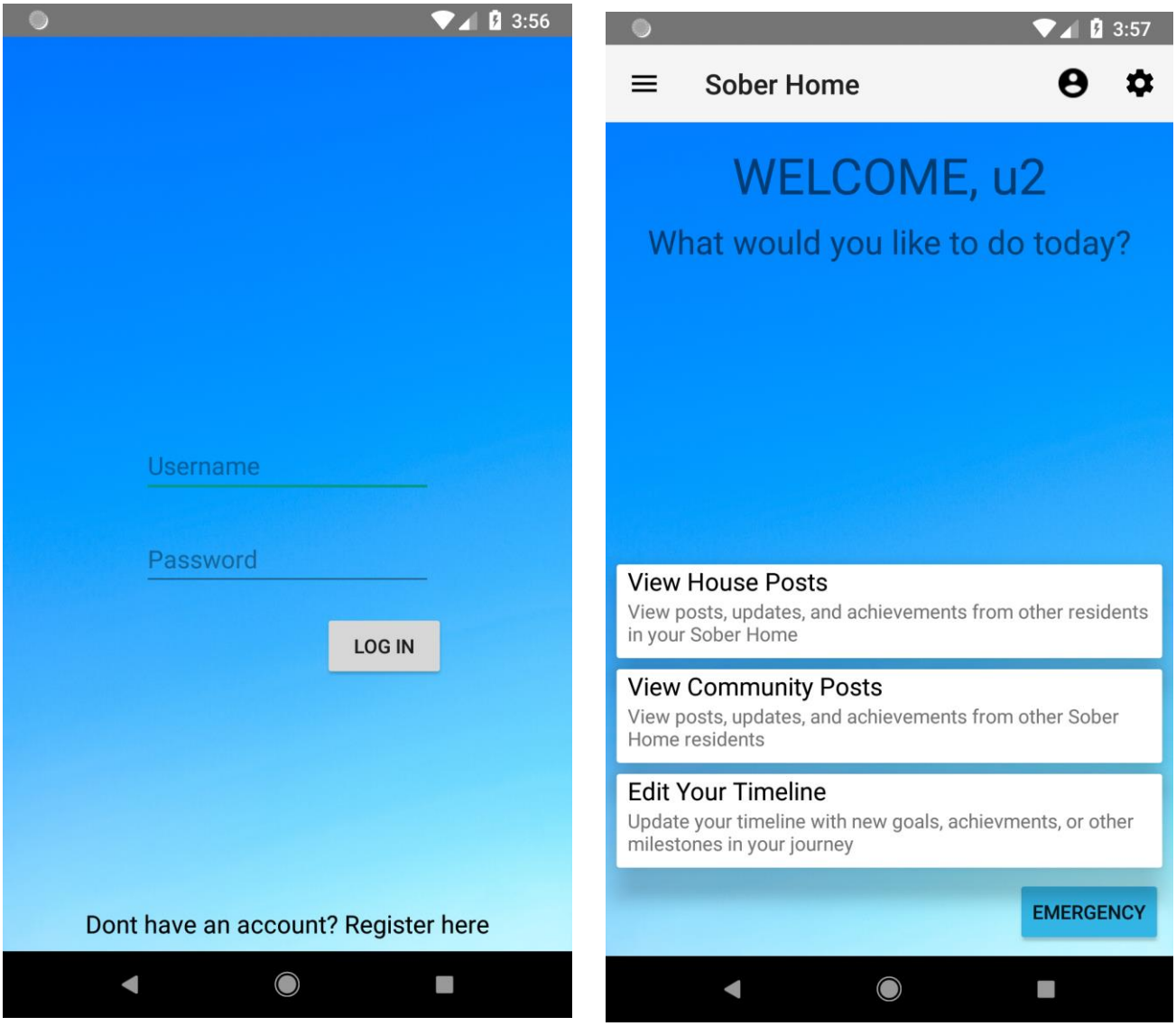
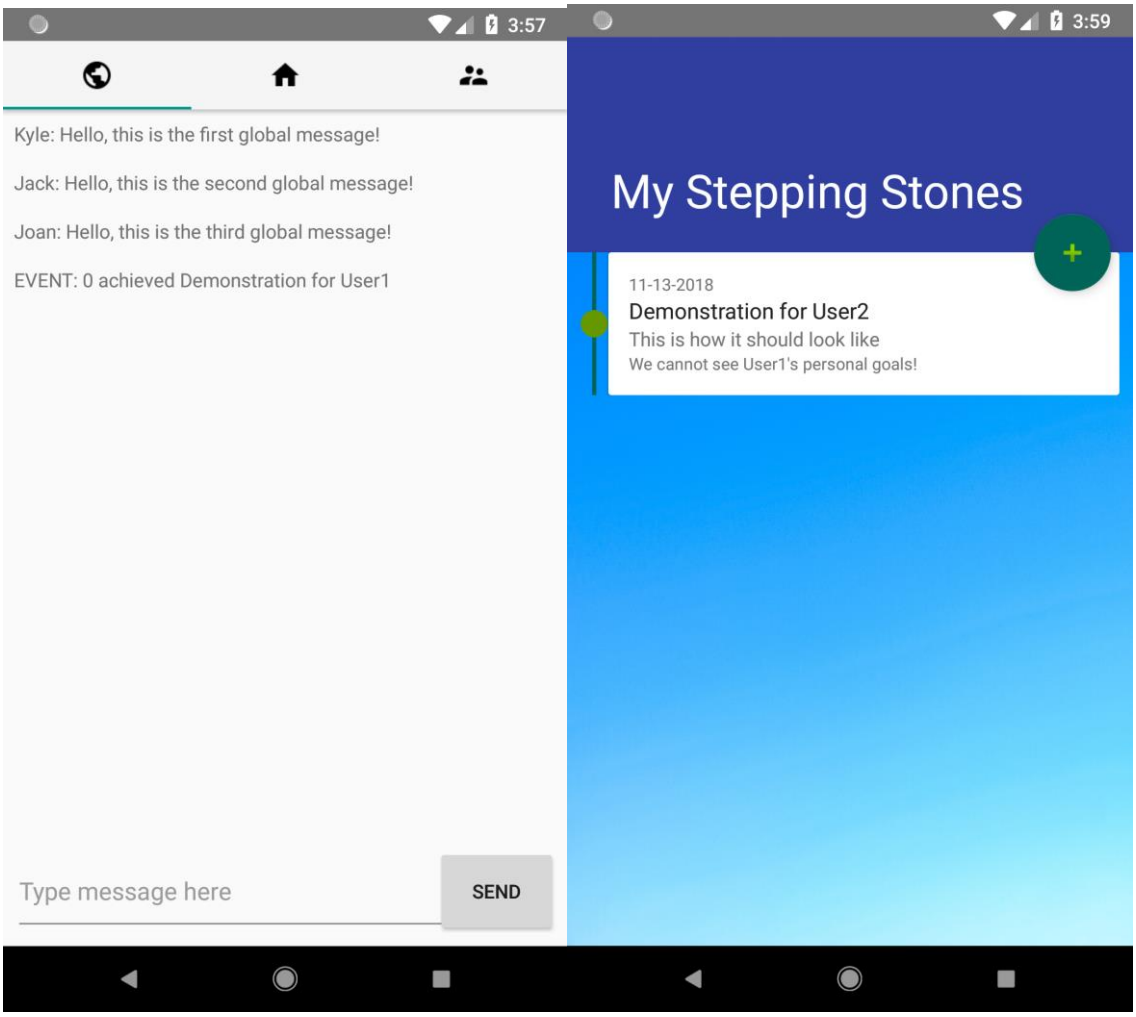

12:30
My Timeline
One Month Sober
Joined House
First Job
+

Figure 5. The Functional UI Mockup




3:57
Kyle: Hello, this is the first global message!
Jack: Hello, this is the second global message!
Joan: Hello, this is the third global message!
EVENT: 0 achieved Demonstration for User1
Type message here
SEND
3:59
My Stepping Stones
11-13-2018
Demonstration for User2
This is how it should look like
We cannot see User1's personal goals!

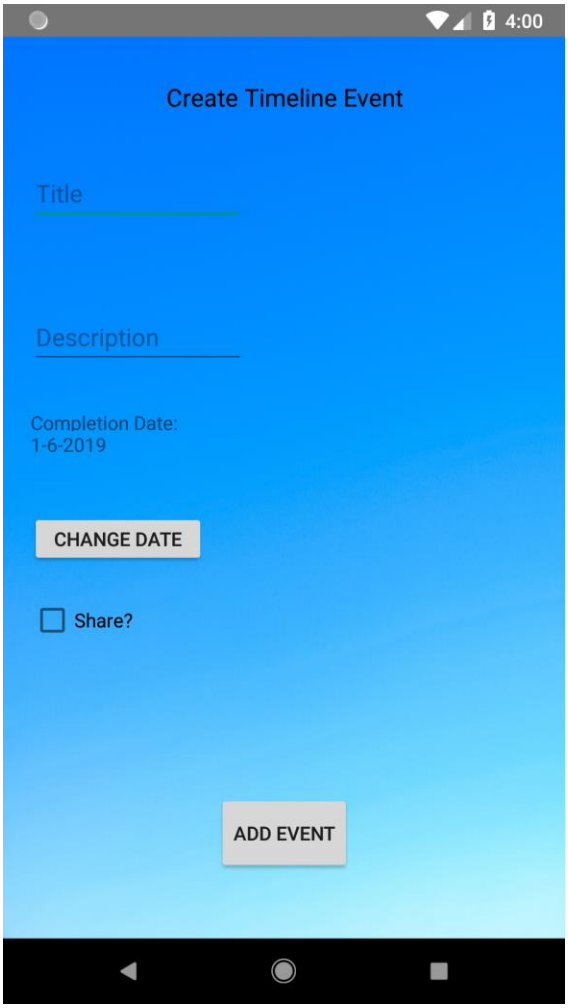
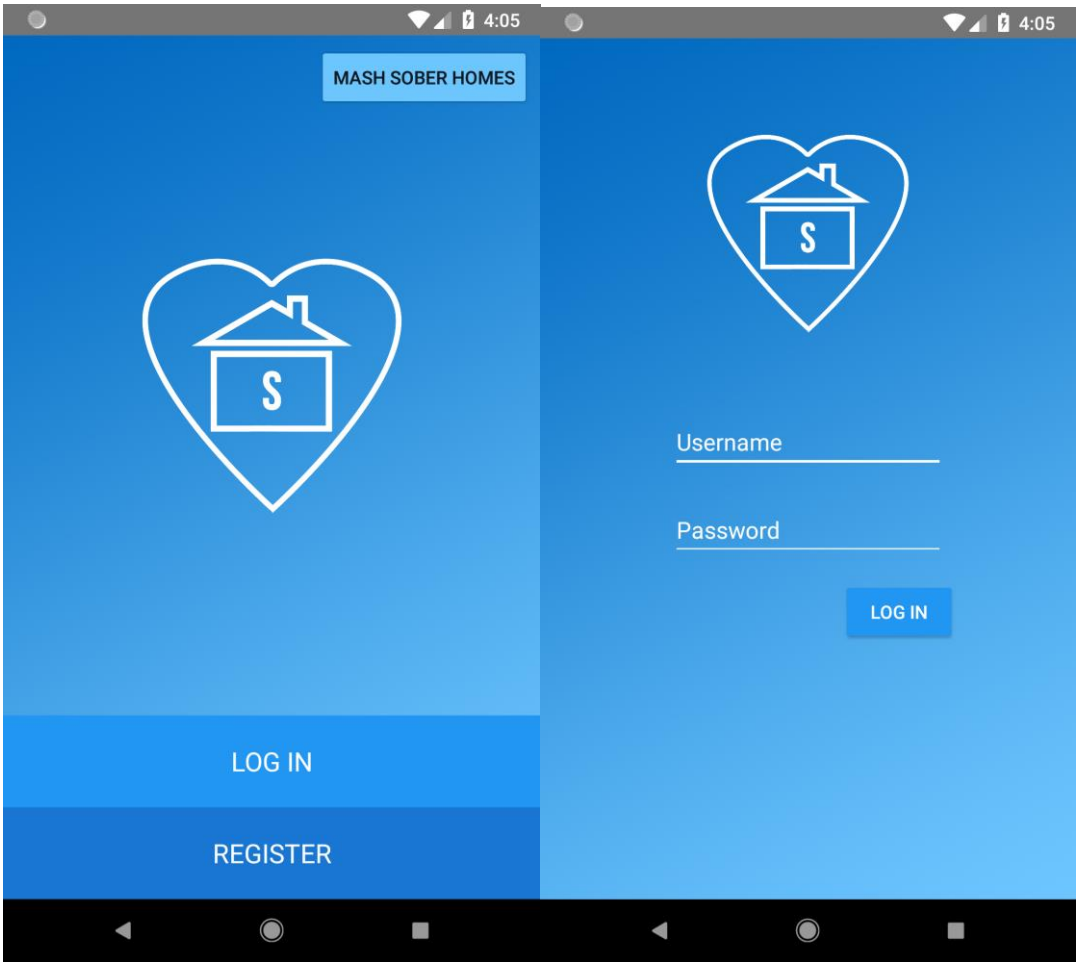
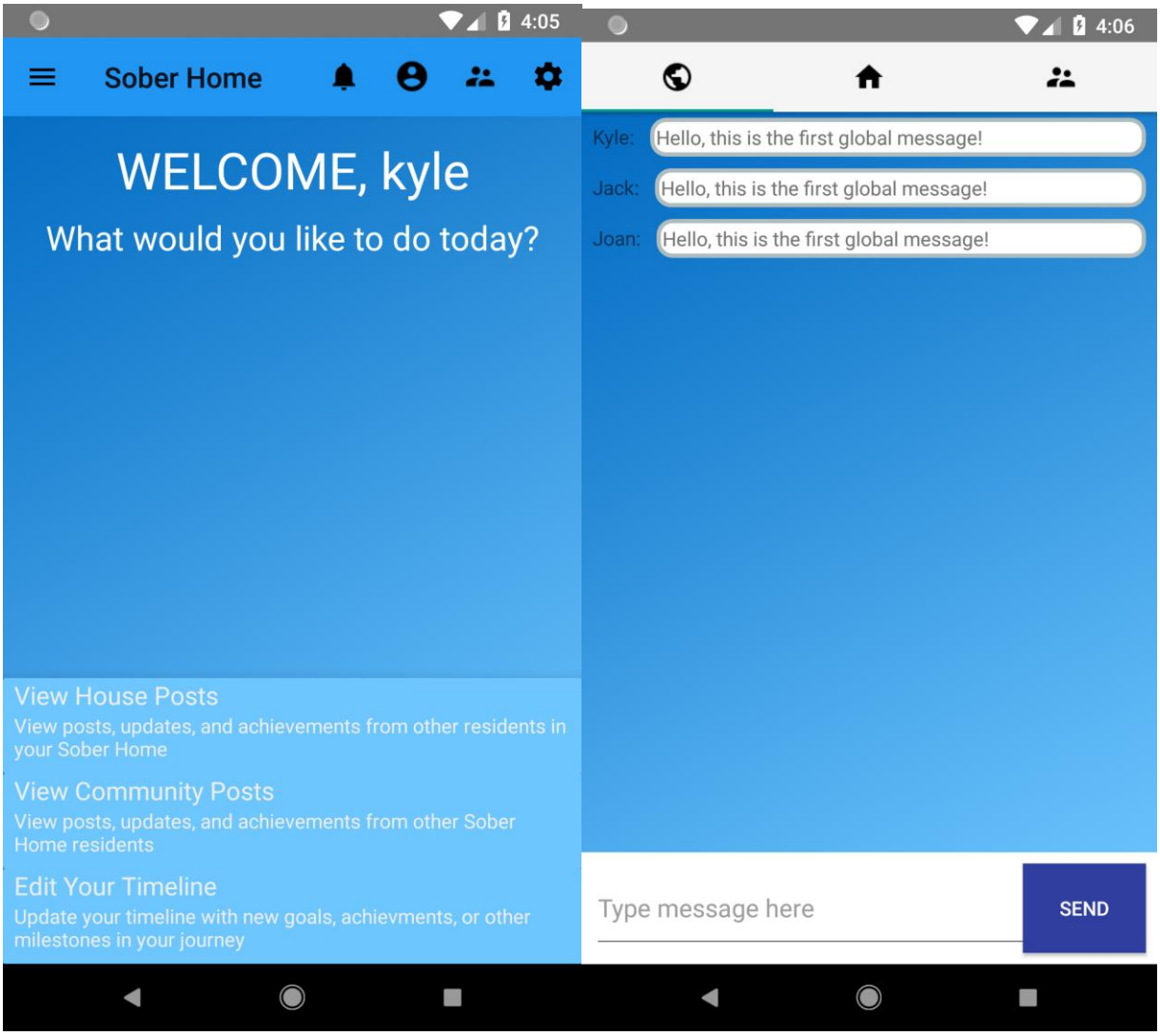
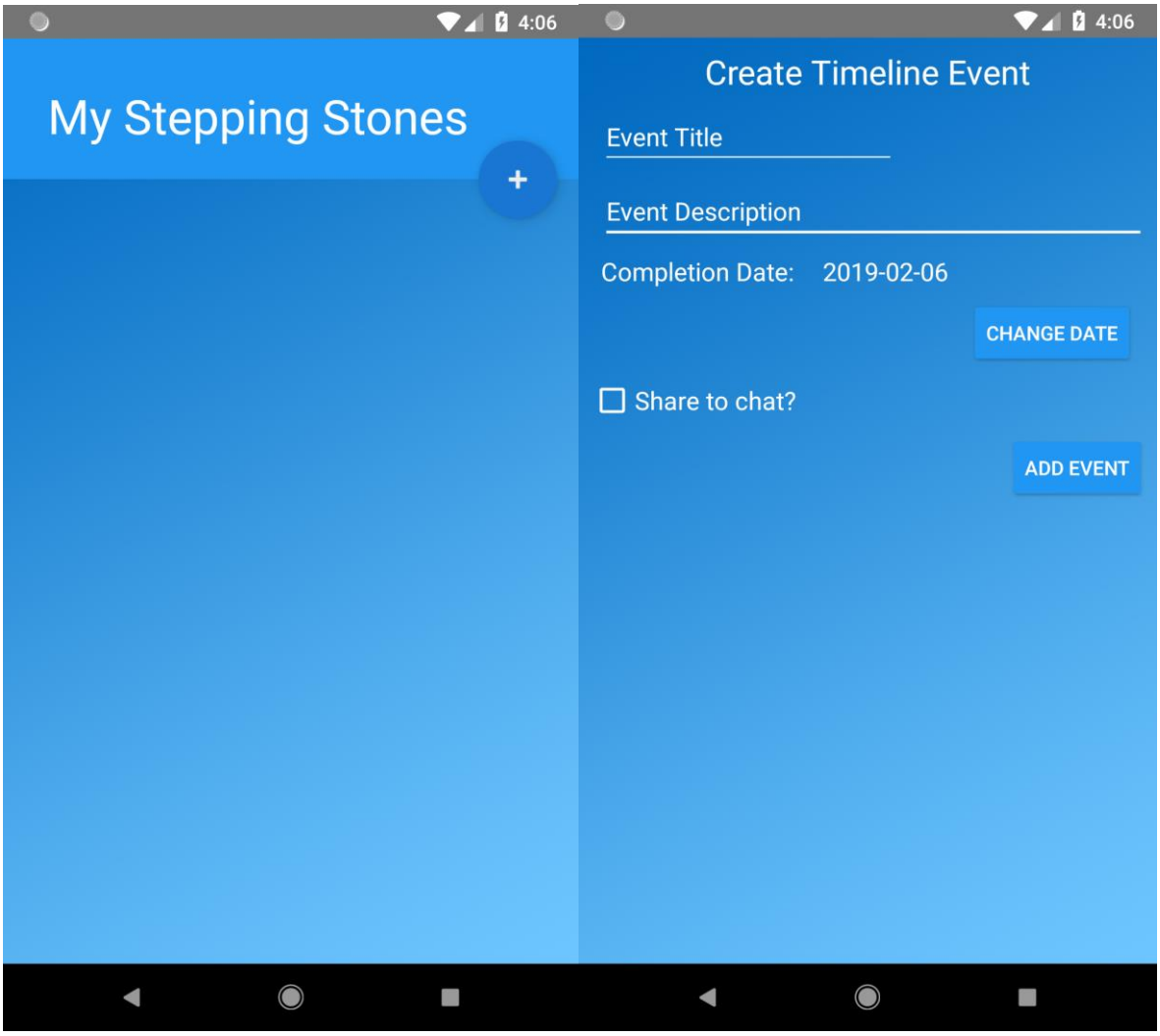
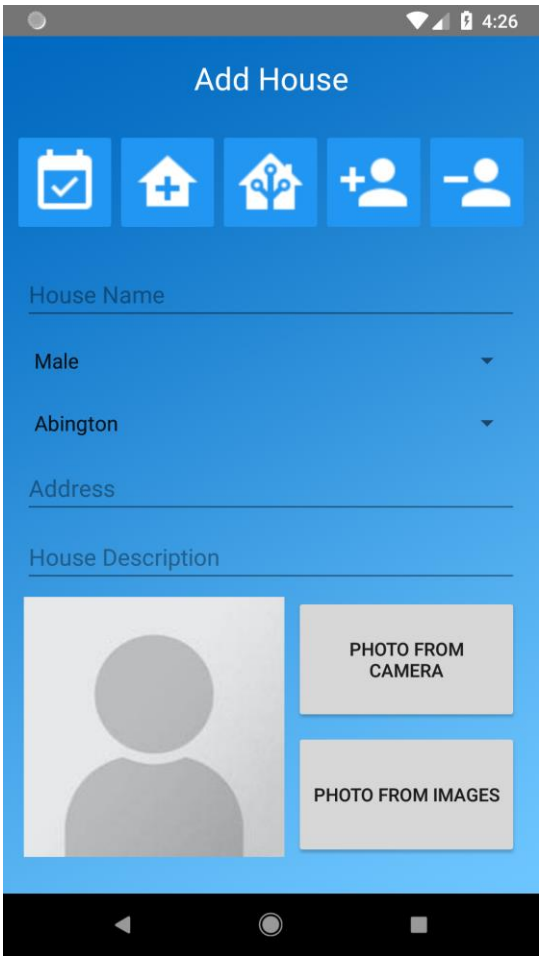

4:00
Create Timeline Event
Title
Description
Completion Date:
1-6-2019
CHANGE DATE
Share?
ADD EVENT

Figure 6. The Final UI Mockup




4:05
Sober Home
WELCOME, kyle
What would you like to do today?
View House Posts
View posts, updates, and achievements from other residents in your Sober Home
View Community Posts
View posts, updates, and achievements from other Sober Home residents
Edit Your Timeline
Update your timeline with new goals, achievments, or other milestones in your journey
4:06
Kyle:
Hello, this is the first global message!
Jack:
Hello, this is the first global message!
Joan:
Hello, this is the first global message!
Type message here
SEND


4:06
My Stepping Stones
+
4:06
Create Timeline Event
Event Title
Event Description
Completion Date: 2019-02-06
CHANGE DATE
Share to chat?
ADD EVENT


4:26
Add House
House Name
Male
Abington
Address
House Description
PHOTO FROM CAMERA
PHOTO FROM IMAGES

# Activity Diagram

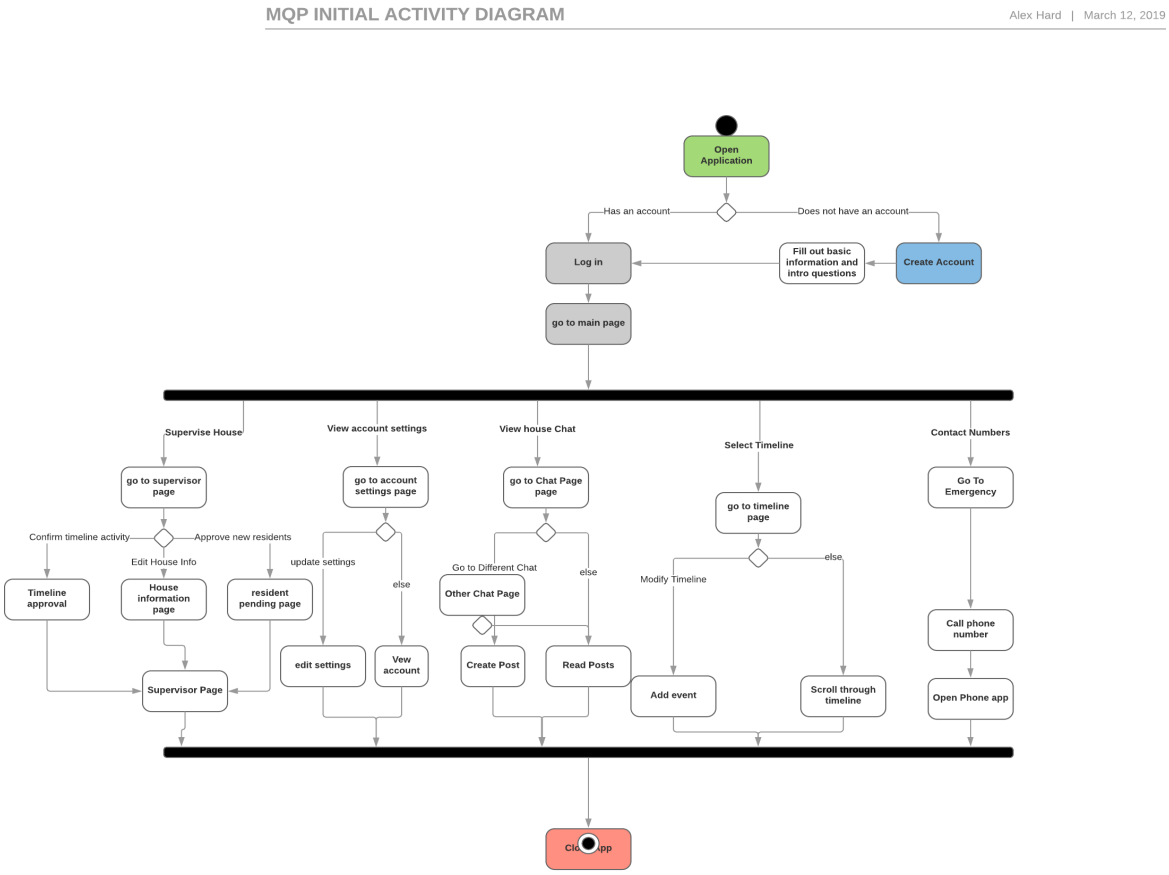



Figure 7. Activity Diagram

# Context Diagram

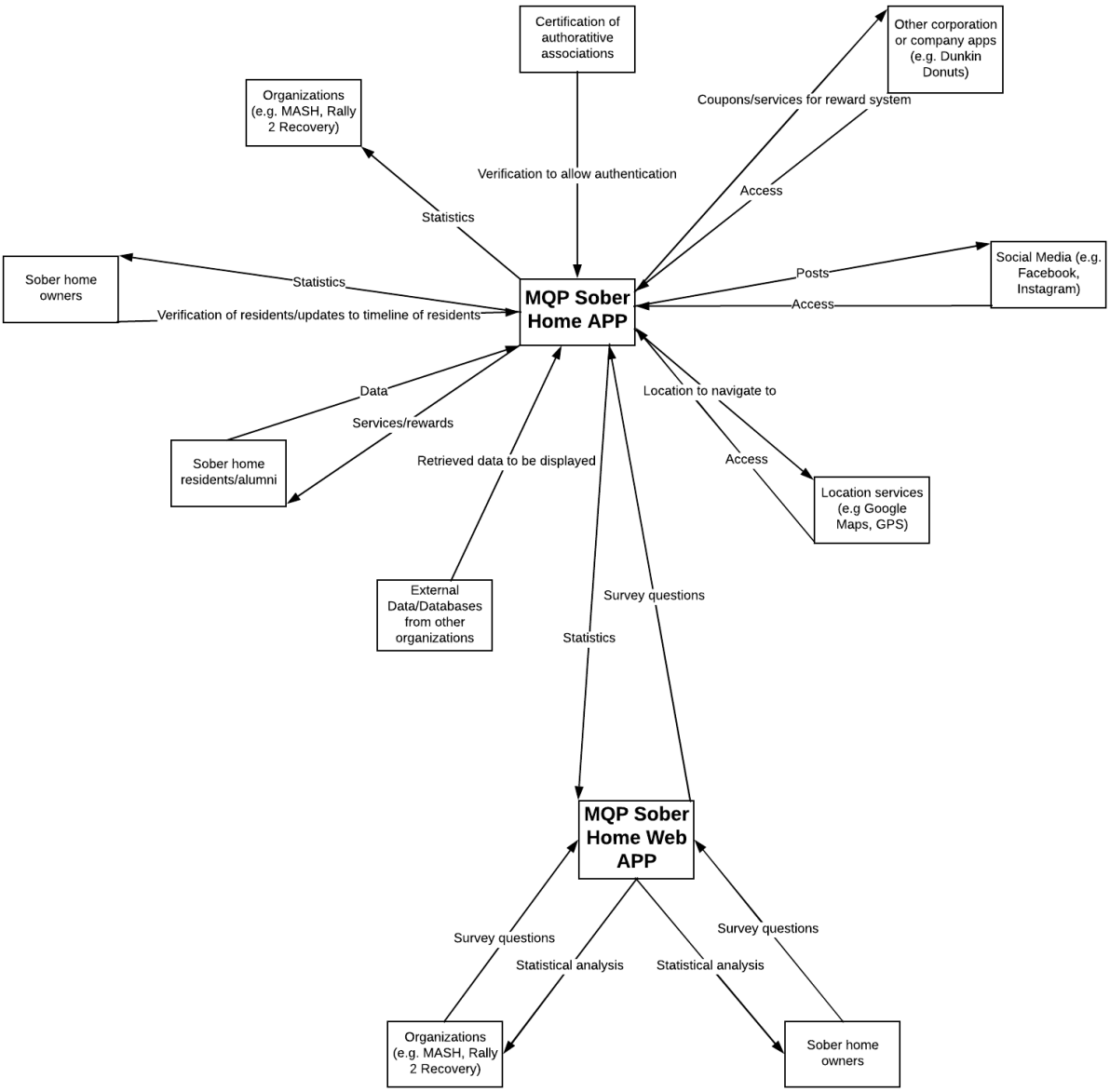



Figure 8. Context Diagram

# Class Diagram

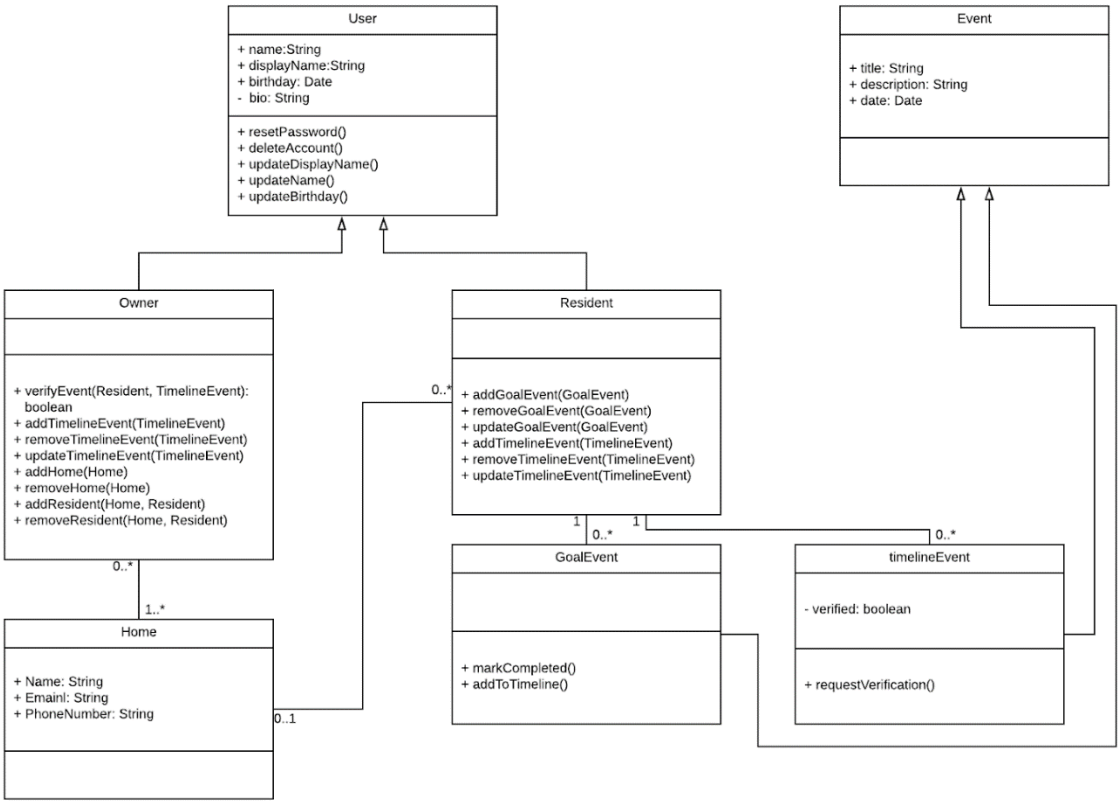



Figure 9. Class Diagram

# Entity-Relation Diagram

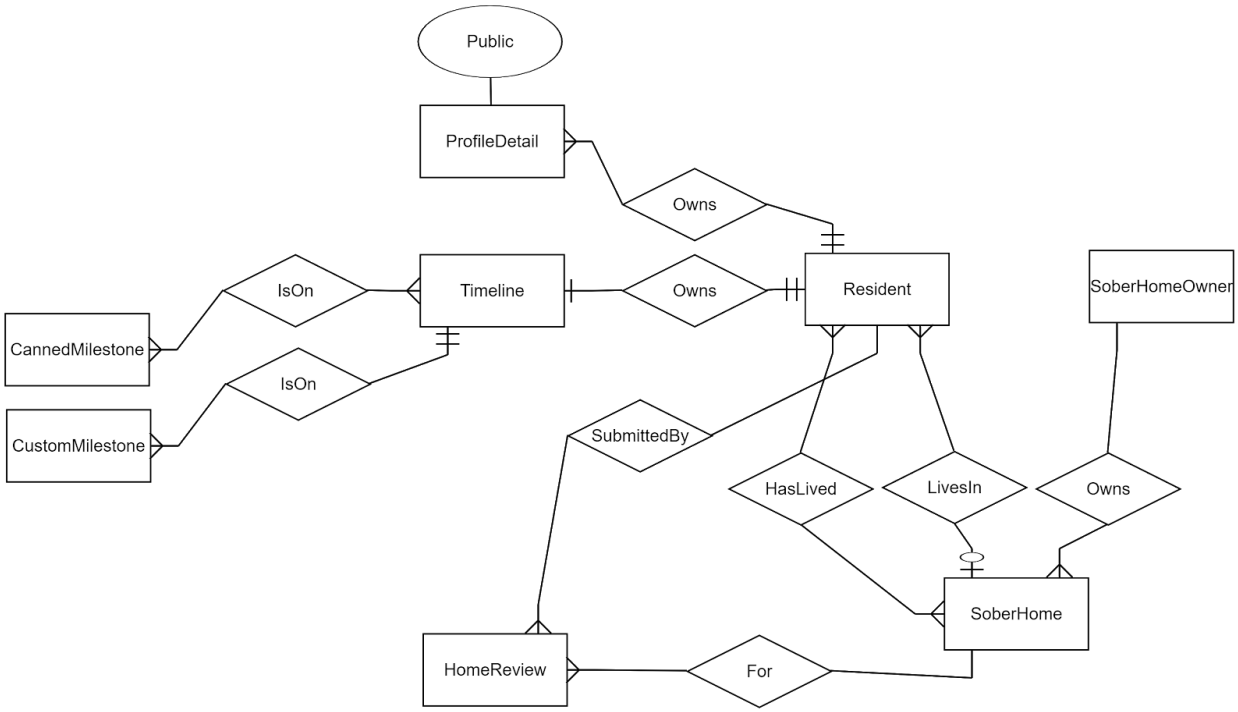



Figure 10. Entity-Relation Diagram

## Design Patterns

Android development includes a few design patterns that are enforced through encapsulation of the SDK. For instance, there is a strict enforcement of the Model-View-Controller (MVC) pattern, as well as a singleton for the 'Context' object of the application which is used to access information about the application. The team used these throughout the application, but because they are standard to Android, the team will focus on design patterns that the team chose to use beyond these in this section.

While switching the application over to use a networked data store, the team chose to incorporate a library, Volley, to handle making requests. The requests were going to simply be interacting with an a RESTful API, so they were all going to follow a similar structure. In order to save time writing complex Volley calls, the team used a facade pattern to only expose the variable parameters such as the target endpoint, arguments, a success callback, and a failure/error callback. In the facade, the team set up a method for each HTTP method needed in that given part of the program (such as authentication or registration). Because Android enforces that network requests must be asynchronous, the team also included some way to keep track of all active requests. For this a NetworkManagerSingleton was used which handled the array of active requests, sent the new requests, and translated between HashMap arguments and JSON strings.

Even in the case of an HTTP error code or a networking error, Volley will return a JSON object as a response. Passing around JSON objects in the program can get complex, due to the unpacking method declaring that it can throw a JSONParsing error that needs to be in a try/catch block which is why the team decided to use the a variation of the Null Object pattern. An 'error state' JSON object was defined in the NetworkManagerSingleton and returned to the caller of the

HTTP request (in order to fulfill the method contract). The caller then checked if the returned object was in fact the ‘error state’ object and then decided what to do based on that. This is a variation because the team did not re-implement any methods of the JSON class to deal with being null, the team just took advantage of an instance of an object standing in for a Null value.

A design implementation that the team are particularly happy with is the generation and response recording of Surveys. As seen in the figure below, the application allows for multiple surveys to reuse questions, in case the administrators of the application would like to see if responses vary over time. This design is also aimed at keeping the data collected from the surveys useful while still keeping the individual user responses anonymous. This is accomplished by tracking which surveys users have taken in a many-to-many relationship so users are not sent the same survey twice and keeping the result of the survey in a different relationship, without including the id of the user who submitted the response.

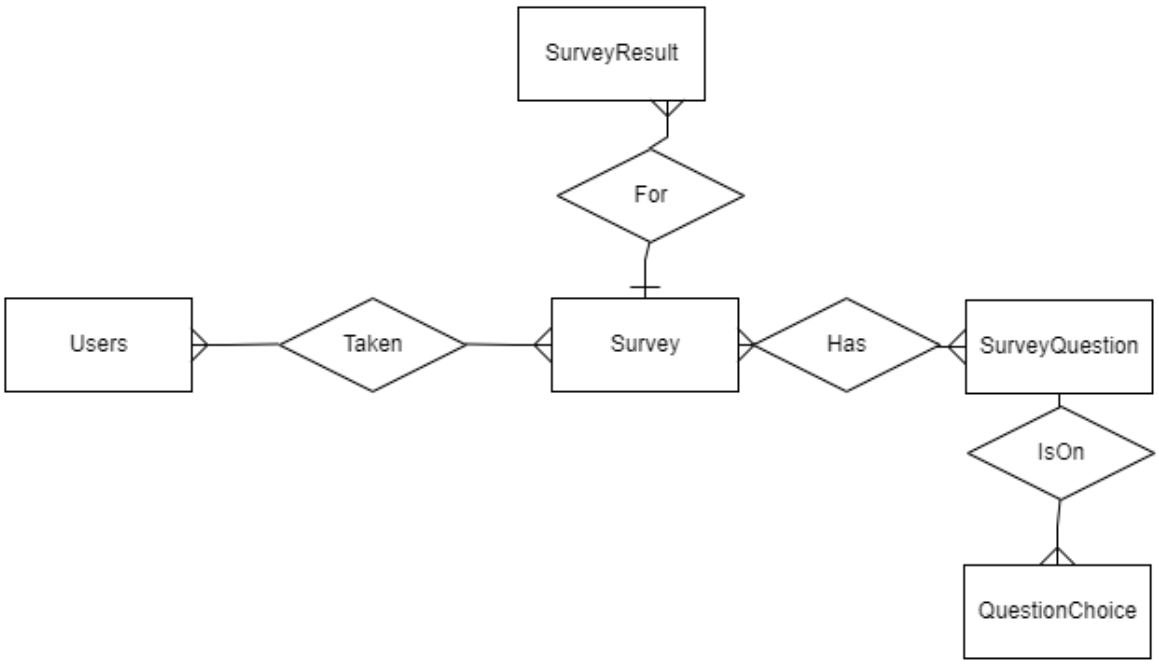



Figure 11. Survey Question ERD Snip-it

# Software Development

## Iteration 0

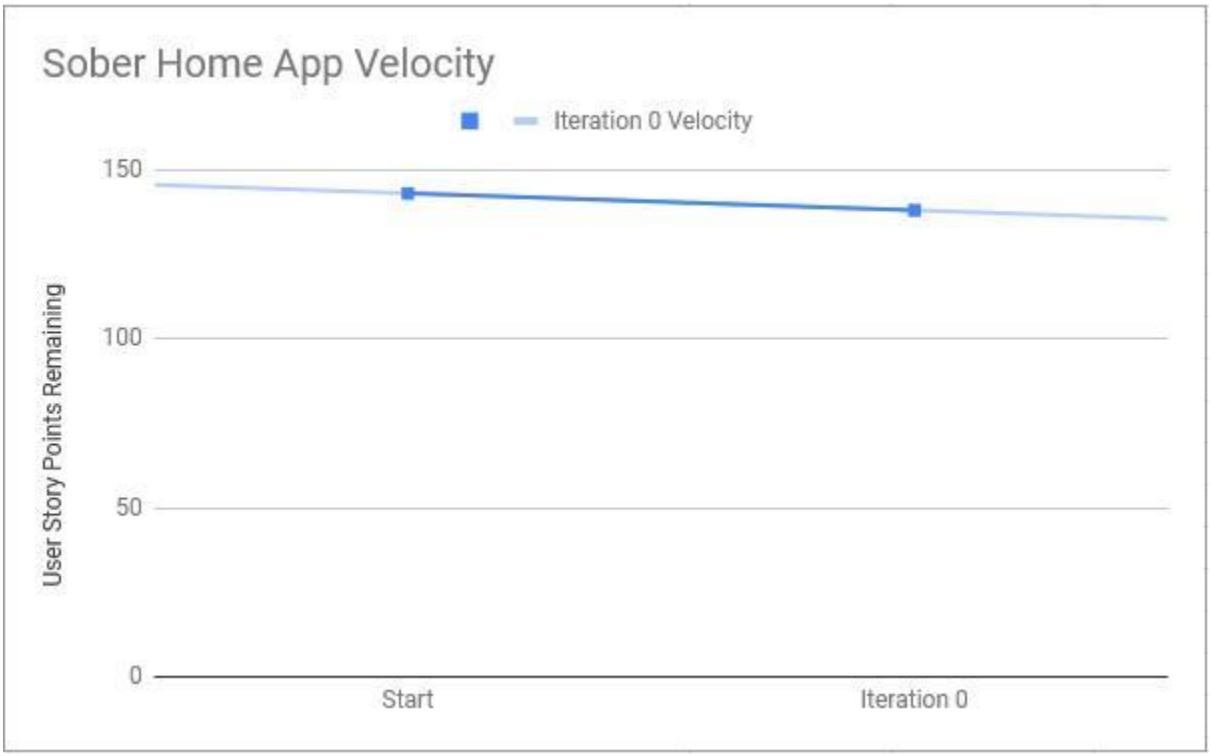



Figure 12. Iteration 0 Velocity Chart

During this iteration two user stories were completed - the account-creation form and the login screen.

- As a RESIDENT I want to [create an account] so that I may [save my data in the app]: 4 points
- As a RESIDENT I want to [login to an account] so that I may [access my profile, timeline, and chat functionality]: 1 point

The velocity for this sprint was 5 points. Since this is the first iteration, the velocity for the project was also 5 points. The database that the group planned to setup was unable to be created

due to the lack of a server. Despite this set-back, this initial iteration went smoothly; all of the functionality that the team intended to create was completed with the help of a stub database connection.

To create a foundation for the application and to get acclimated to programming for Android, the team decided to do an initial iteration to prepare for the actual application. The goal of this "iteration zero" was to create a minimal Android application that collected user input as data and switched between multiple activities. Additionally, the team wanted to create an initial version of the PostgreSQL database so that it would be ready to store the data for the application.

The application that resulted from this iteration had three interconnected activities that were utilized in subsequent iterations; a login page that was the entry activity, a registration page and a homepage that greeted the user. To switch between activities and retain basic information from user input, the group used intents. Intents are the way Android starts new activities and shares information between those activities. These intents are used to launch the registration page and the homepage from the login screen, and the username and password from the login fields were passed into these intents as extras.

The login activity had username and password fields, a login button, and a prompt at the bottom of the screen to take the user to the registration page. If the user pressed the login button it would take them to the homepage, which for this iteration was left blank except for a message that displays the user's name. Basic input validation was implemented to check that the username and password fields were not empty, and if they were, an error message would pop up.

The registration activity contained basic demographic information that the team thought would be helpful for both creating a user account and collecting the data. In this iteration the registration page asked for the user's name, age, birthday, gender, email and the current sober

house they were a part of. The user was also prompted to choose a password using a password field and a confirmation field. As with the login activity, all of the registration fields contained input validation so that if a field  that was required was empty an error would pop up. At the bottom of the page was a submit button that took the user back to the login screen, and was later used to add the new user to the database.

Due to not having access to a database server during this iteration, the group decided to create a database stub which would emulate data transfer. Using this stub connection, data was temporarily stored in a CSV file for this iteration. All of the database calls in this minimal application were done using this stub connection; it stored the registration information and checked that the login credentials were valid.

In order to prepare for data transfer with the actual database the team would be using, the group started to create an interface for the database. This interface handled all of the SQL queries to the database, so the application could retrieve data just through function calls built into the interface. The plan was to later move this interface into the database server so that it could also perform validation on the data being sent to the database.

## Iteration 1

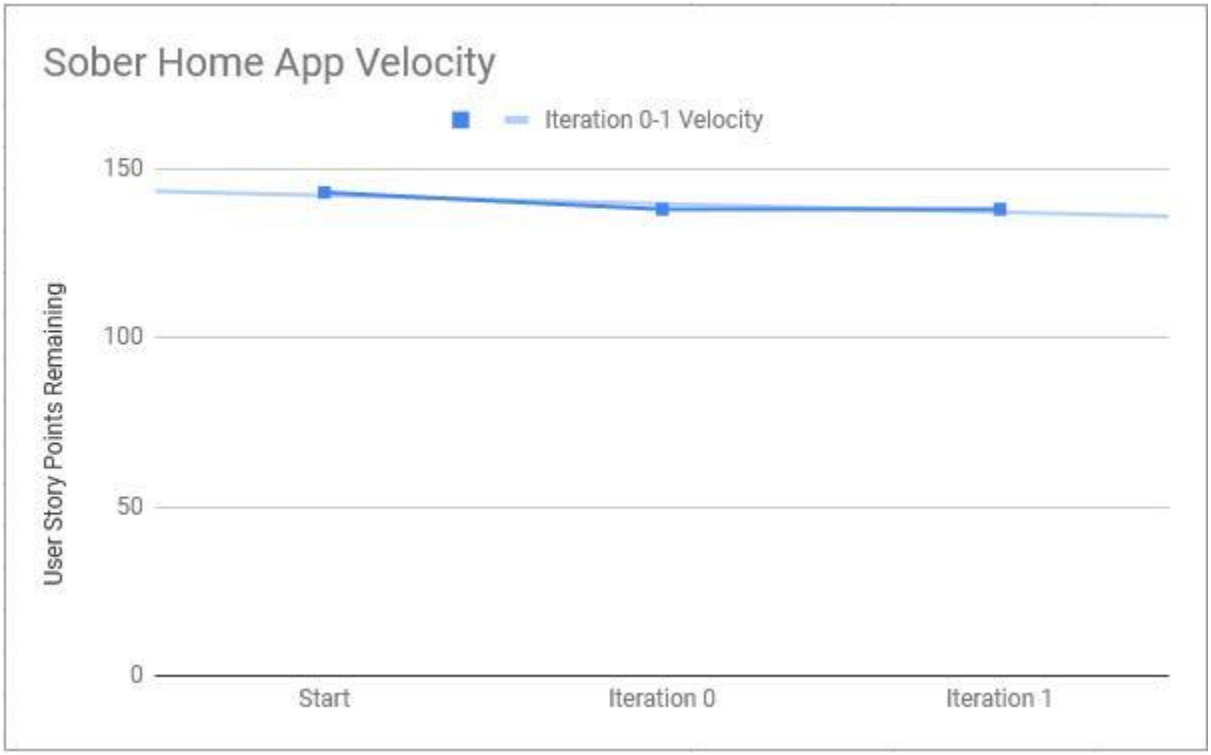



Figure 13. Iteration 1 Velocity Chart

During this iteration no user stories were completed. Thus, the sprint velocity was 0 points, and the project velocity for this iteration was 2.5 points. No user stories were completed because a couple of features added during this iteration such as an emergency contacts page were later removed due to their unforeseen complexity. Despite the setbacks, the group learned widgets and systems that would be used throughout the entire application and had a plan for the next iteration.

For the first official iteration of the application, the team decided to focus on a few smaller activities that would be integral to the rest of the application, but that would also allow the group to continue to learn the Android system. These activities included an activity to change the user information created in the registration page from iteration zero, a profile activity and an

emergency contacts activity. As part of the profile activity, the team planned to integrate other applications such as the camera and photos apps to get images to use for the user's profile picture. The group did not know if the database would be ready, so they planned to implement the database functionality for these activities in a later iteration.

At the end of the iteration, the deliverable did not meet the expectations for the iteration. The emergency contacts activity was finished, however the profile page and settings pages were not fully completed. The group also discovered limitations of Android that hindered the backend. Therefore, a portion of this iteration was dedicated to researching the constraints of the system so they could be factored into the design of the application.

The page that was created for updating the user account information directly mirrored the registration page; they both had the same fields, but the update activity filled these fields with the information that the user already input for their account. Each field could be edited, and when the user pressed the submit button the information was updated. Due to not having the database, the updated information was instead put into the database stub. This page did not see further development due to time constraints.

The emergency contacts activity contained a list of phone numbers that could be called for help. This page utilized the RecyclerView widget for Android, which was used throughout the application in future iterations. This widget allows custom-formatted information to be displayed in a vertical list, which enables easily scrollable pages. In this instance, each item of the RecyclerView contained a profile photo and a phone number. When an item was pressed, the system's phone app was started and the corresponding phone number was automatically input.

This type of external integration with the phone application was what was planned for the profile page, however this was not accomplished in this iteration. The profile activity for this

iteration just contained the minimal UI that would be used on the page, including an ImageView to hold the profile image, a field for the user's name and a description field that the user would be able to update. The name field was populated with the name received from the database stub, but beyond that the profile page for this iteration did not contain any functionality.

After running into limits of the Android system, the team refactored the backend to meet these constraints. The group discovered that in the future, data fetching could not be run in the main thread of the application. This was because the data was going to be networked, and Android raises an error if a network request is run on the main thread. The team refactored the code to use Volley, a library for making requests on Android. After spending a significant time planning the design,  a singleton network manager to abstract some of the intricacies associated with networking was chosen. This included converting arguments to JSON strings, handling parsing errors, timeout errors, and setting proper headers. The team also ended up including the null object design pattern to represent some of the error states that can result from issuing a network request.

Although the progress of the application did not meet the groups standards, a few key decisions were made that would benefit the project in future iterations. To adhere to the scrum-agile methodology the group chose, they assigned Kyle as the scrum master and Alex as the product owner; from then on Kyle would be in charge of initiating the scrums, and Alex would be the main point of contact between the team and the project's sponsors. The team also created the user story backlog and agreed on a way to rate user stories; A rating of 1 would take the least amount of time to complete. A rating of 2 would take twice as long to do as a user story with a ranking of 1, a rating of 4 would take twice as long to do as a user story with a ranking of 2, and so on.

Since the database was still not available, everyone decided to focus on the user interactivity. Thus, user stories such as account tiers were saved for when the database was set up. The team agreed to work on the timeline user story for the next iteration, since it seemed to be the task that would take the longest to do. To aid in knowing what features to add to the application, the group decided to visit a local sober home and talk to the sober home owners. This way everyone would get a better grasp of the user group, and they would be able to ask about specific features that sober home owners wanted.

## Iteration 2

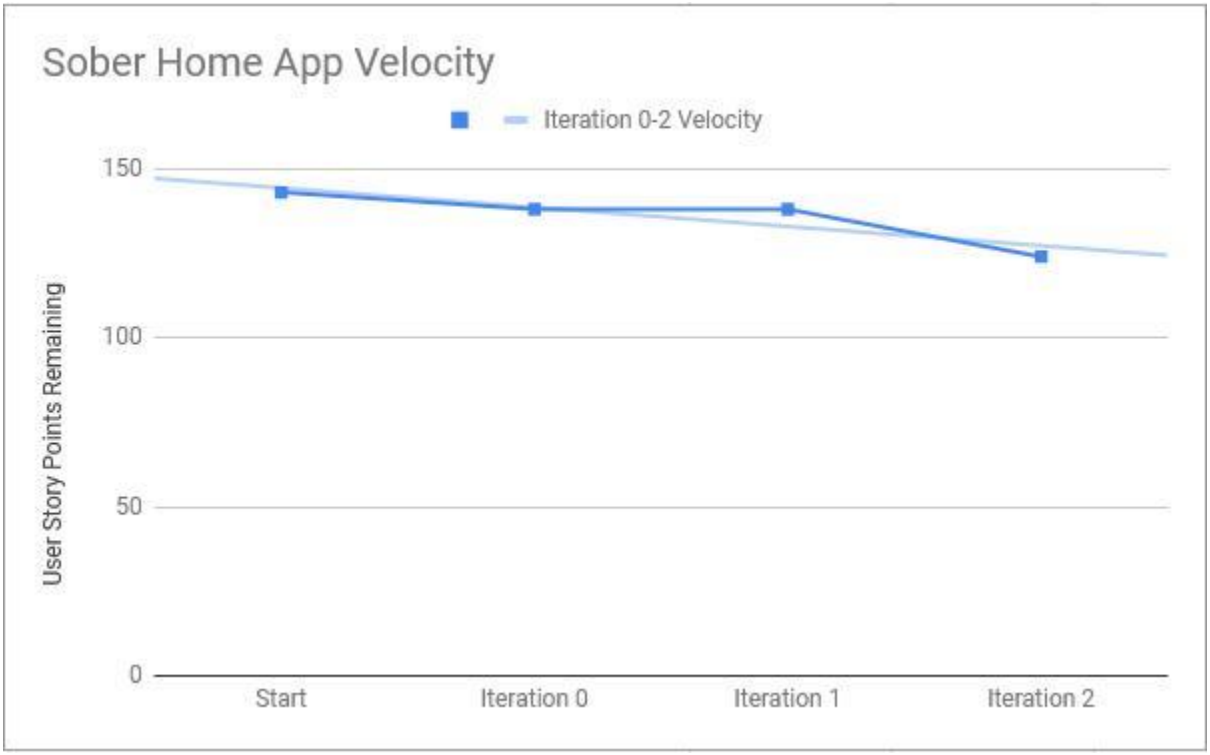



Figure 14. Iteration 2 Velocity Chart

At the end of this iteration the team completed three user stories - the ability for users to set a profile picture, the timeline system and the ability for a user to create a custom event for their timeline.

- As a RESIDENT I want to [view my timeline] so that I may [see my recovery progress]: 8 points
- As a RESIDENT I want to [create a custom event] so that I may [share personal progress]: 4 points
- As a RESIDENT I want to [add a profile picture] so that [others may know who I am]: 2 points

The sprint velocity was 14 points, and the velocity for the whole project for this iteration was 6.33 points. The team felt positive about the following iterations with one of the major features implemented unlike the previous iteration. Looking forward the team sought to implement the next main feature as well as begin to revise the look and feel of the application to make it more user friendly and visually appealing.

The second iteration of the application saw the first major features being added. In addition to completing the unfinished user stories from iteration 1 such as the user profile page, the team added the event timeline feature. This feature allowed users to track and save notable events throughout their recovery as well as share those event with other members of their sober homes. This feature was achieved through the use of an external timeline library that was integrated into the application. While initially thought to be a placeholder design, the design of the timeline was received well and thought to be an acceptable layout to remain in the application.

During this iteration the team also began to add more supplementary features such as a sidebar for navigation and the start of a settings page. These activities and features were added to the application, however at the end of this iteration their full implementations were not completed in favor of more major features. These features were planned to be finished in a later iteration once the core of the application had been completed and made fully functional, however due to time constraints they were never fully implemented.

## Iteration 3

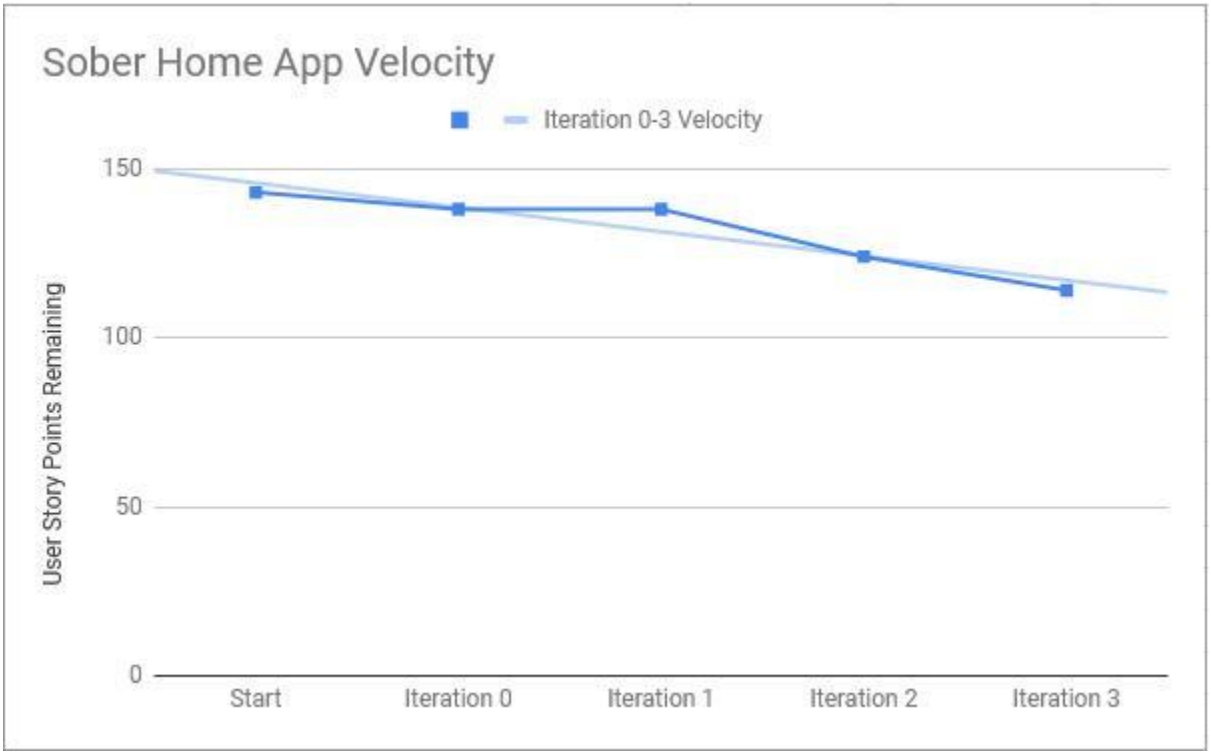



Figure 15. Iteration 3 Velocity Chart

During this iteration, three user stories were completed - the implementation of the chat feature, the different categories for the chat (global, house and owners) and having public timeline events appear in the house chat that a user belongs to.

- As a RESIDENT I want to [chat with other people who are recovering] so that I may [encourage others and be encourage by others]: 4 points
- As a HOUSE OWNER I want to [chat with other house owners] so that I may [share and get helpful tips for running a house]: 4 points
- As a RESIDENT I want to [make goals and events public] so that I may [show others how I am progressing]: 2 points

The sprint velocity was 10 points, and the project velocity for this iteration was 7.25 points.

The third iteration of the application sought to increase usability, visual appeal, and add the second major feature: a chat system. Up until the third iteration the UI was plain white with black text. During the third iteration the team created numerous potential color palettes and background options before settling on a blue theme with a gradient background. In order to prevent spending too much time on that design a placeholder gradient and color scheme were used, which was intended to be replaced by a custom designed theme in iteration 6. The user interface was also updated on the main screen by the addition of cards linking to the timeline and chat features. This allowed users to directly go to these main features rather than navigating through a maze of profiles and settings in order to find them.

The chart feature, a major goal for the finished application, was also implemented this iteration. The chat feature had three main rooms: a house room, a global room, and an owner room. The house chat room allowed application users to communicate with the people inside their sober home. This was designed to promote inner house communication and be a place for topics of discussion or announcements The global chat allowed all application users to communicate and have a platform for discussion that extended past in-house conversations. The administrator chat allowed house owners and MASH administrators to converse regarding house issues, openings, and other executive level discussions. It was designed so that each account was given permissions to the appropriate channels, which was implemented in the last iteration.

Iteration 3 also saw the expansion of the registration activity. A second page was added to account registration and included a basic survey. This data included topics such as employment history, recovery history, and any sober home history. This was predicted to be data MASH wished to collect from application users, however the team did not have a complete list of data to collect.

Finally, iteration 3 saw the refactoring of the event system to share events to the chat channels if they were set as public as well as changed some input types to reduce possible errors.

# Iteration 4

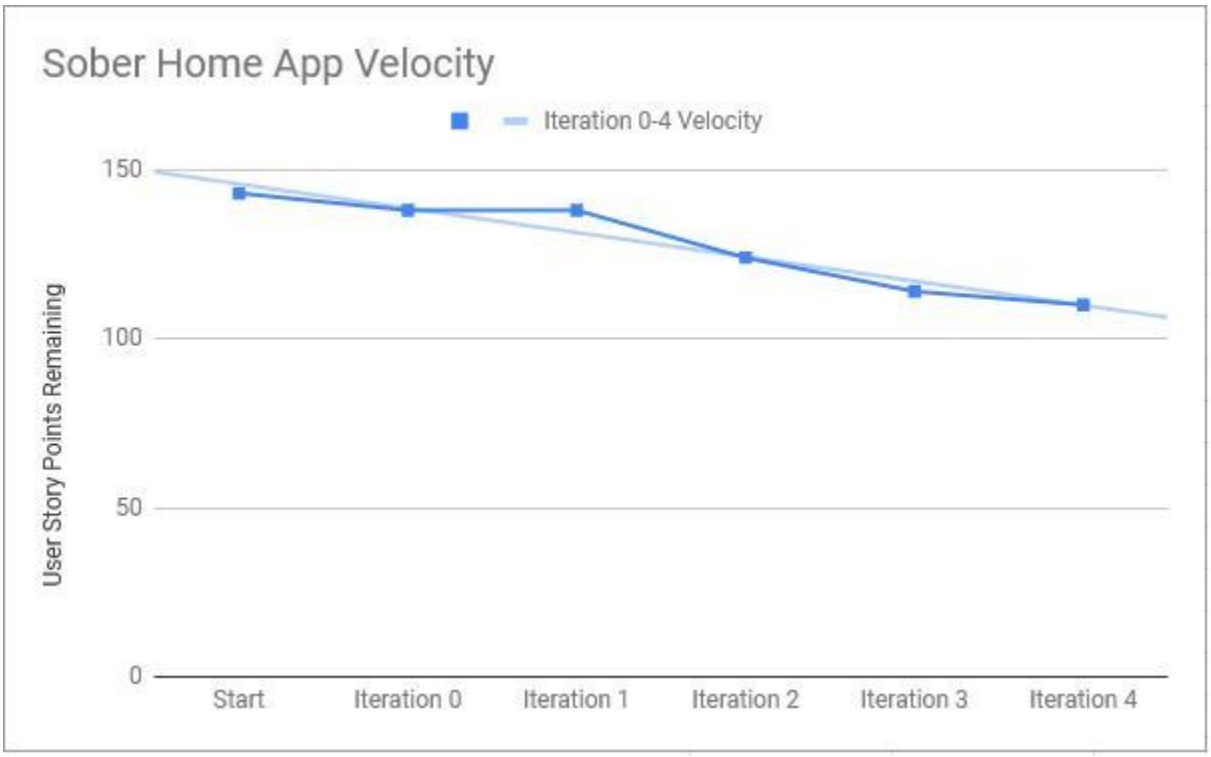



Figure 16. Iteration 4 Velocity Chart

During this iteration, two user stories were completed - the ability for a sober home owner to create a sober home within the application and the setup for the web application.

- As a HOUSE OWNER I want to [create a sober home in the application] so that I may [add residents to it]: 2 points
- As a APP OVERSEER I want to [collect basic data from residents] so that I may [know how best to support them]: 2 points

The sprint velocity was 4 points, and the velocity for the project during this iteration was 6.6 points. The velocity was much lower this iteration because only a couple smaller user stories were completed, but many user stories were partially completed.

Iteration 4 included maintenance of existing features, improvements of others, the start of a web application to view the collected data, and the creation of a supervisor page. The supervisor feature was designed to allow sober home owners to manage their home's page as well as resident's accounts inside their home. The page allowed the supervisor to approve accounts to join their home's chat channel, remove users that may be detrimental to other residents, and edit or update their houses picture or description. This was an extension of the permissions given to accounts that could access the administrator chat channel.

In addition to the supervisor page, a house page was created that allows user to view information about the sober home they reside in as well as post a review of the sober home. The information displayed on this page was entered through the supervisor page. This page was reworked into the list of sober homes residents could browse that was implemented near the end of the project.

This iteration also included the refactoring of numerous exists activities to extend functionality or improve performance on different devices.

Iteration 4 also marked the beginning of a web application to view the data collected during registration. In order to allow MASH administrators to easily interpret and use the data collected, the team elected to create a basic website using Vue.js to pull live information from the server and display it in an easy to understand presentation.

## Iteration 5

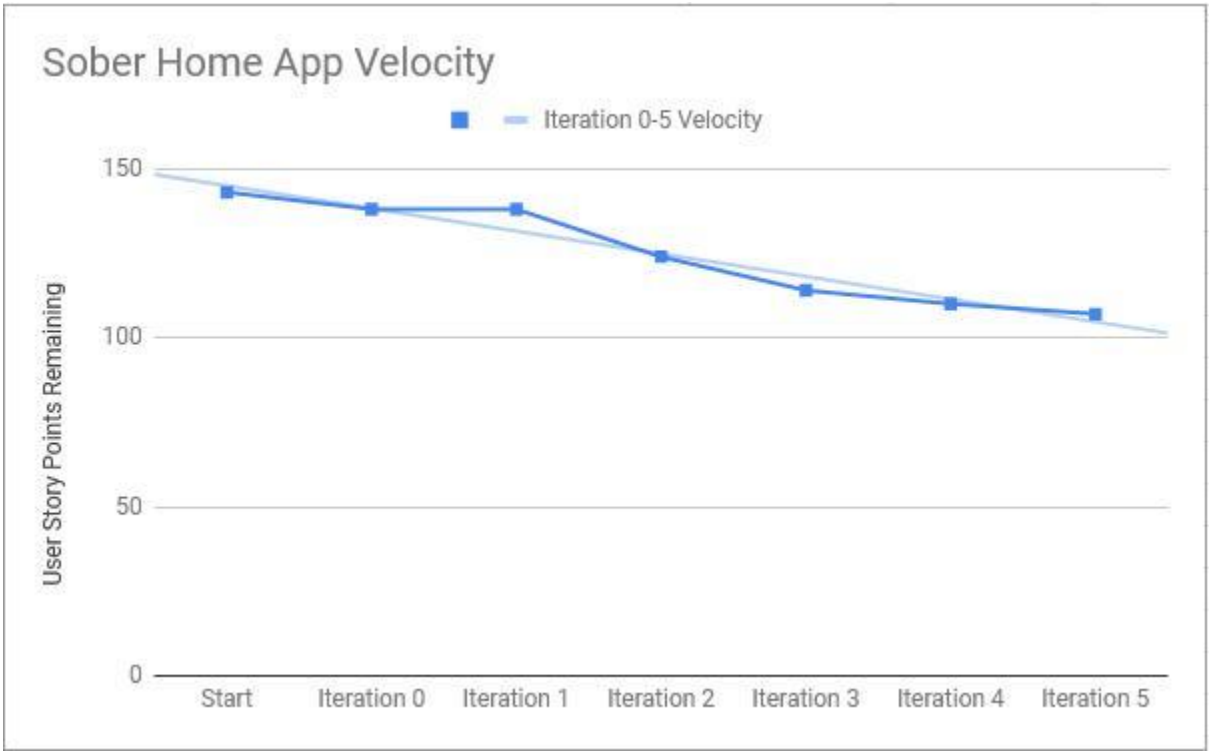



Figure 17. Iteration 5 Velocity Chart

During this iteration, two user stories were completed - the ability for sober home owners to process resident housing requests and the ability for an owner to remove a resident from their sober home by giving a reasoning.

- As a HOUSE OWNER I want to [approve new residents] so that I may [regulate who joins the application]: 2 points
- As a HOUSE OWNER I want to [explain why someone was kicked out of a sober home] so that I may [justify why a resident was removed]: 1 point

The sprint velocity was 3 points, and the velocity for the project during this iteration was 6 points. The velocity was lower this iteration because most of the iteration was spent planning and working on the paper.

Iteration 5 contained a few additional features but was focused on non-development project work such as paper additions, meetings with the sponsors, and future planning. The iteration began with a meeting between the team and Rally to Recovery, where the current state of the application was demonstrated and reviewed. This meeting provided insight into future expectations and good feedback on the current design of the application.

The web app was expanded upon and had the charts.js API added. This was done to allow data collected to be easily displayed and understood, and did much of the heavy-lifting for the web-app.

The supervisor activity was polished to include most of the interactivity that the final version would have. The different sub-activities within the page now interacted with one another, and all of the data needed to complete all of the various forms were added to the application.

From the meeting with the sponsors the idea of Facebook integration was discussed. The team elected to spend time in this iteration to explore the Android Facebook SDK and begin to architect its potential integration into the teams application. The sponsors wished to add a form of private chat between users, and the team felt it may be beneficial to integrate an existing, well known and used platform rather than develop one from the ground up. It was discovered that the Facebook SDK was much more complicated to implement than was first thought, so implementing it was eventually deemed outside of the scope of this project.

The team also began to plan the UI and layout redesign following Google's material design standards. Material is the current standard for Android application UI and is used almost universally. Following those standards allowed the application to be more intuitive for use and fit in well with the look of the Android ecosystem.

## Iteration 6

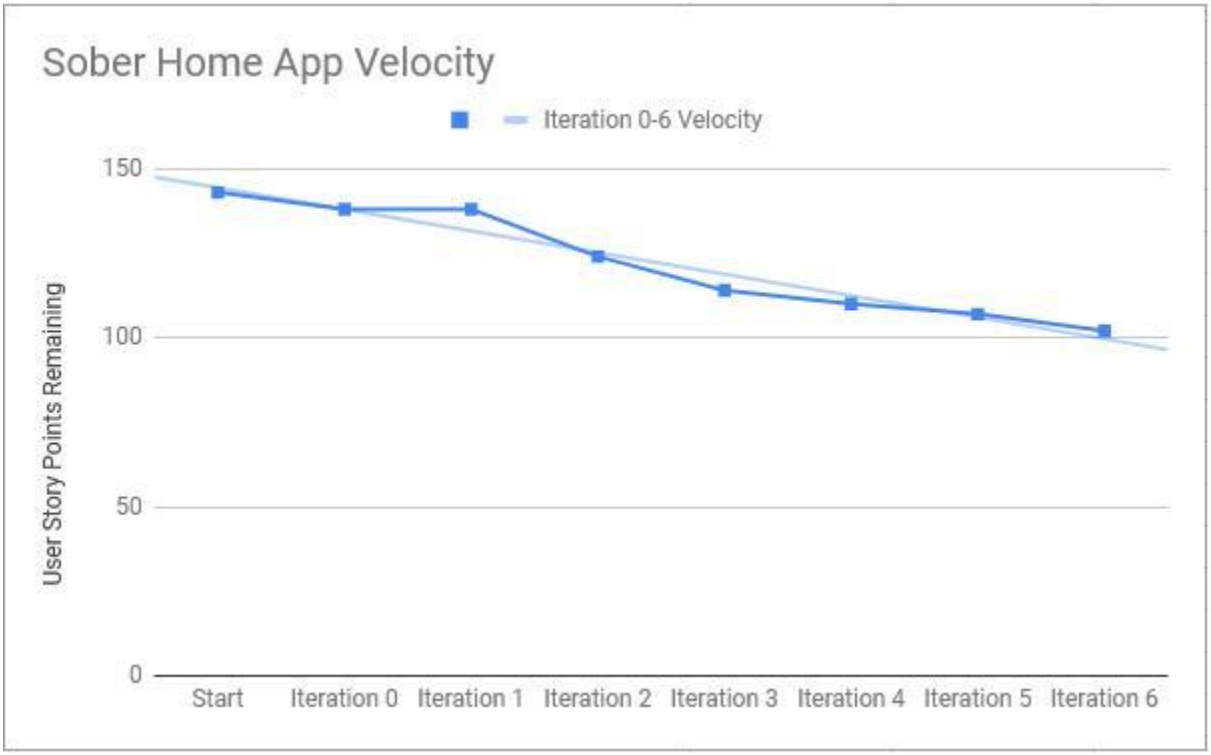



Figure 18. Iteration 6 Velocity Chart

During this iteration two user stories were completed - the ability for supervisors to approve resident events and the option for users to opt out of questions during registration and polling.

- As a HOUSE OWNER I want to [approve of resident events] so that I may [verify or deny whether an event actually took place]: 4 points
- As a RESIDENT I want to [opt out of sensitive questions] so that I may [be comfortable with the information that I am sharing with the application]: 1 point

The sprint velocity was 5 points and the velocity for the project during this iteration was 5.86 points. The velocity was low this iteration due to the important meetings that took place and because of how close this iteration was to the holidays. Because of the approaching holidays, the

team focused a lot of effort preparing the paper proposal so it could be edited during the winter break.

During this iteration the team started setting up the final database and implemented additional functionality and polish to parts of the application that were worked on earlier. There were also a couple important meetings that took place during this sprint that laid out the work for the rest of the development of the application.

The group had a meeting with one of the WPI faculty in charge of the WPI servers to discuss using some of the server space to temporarily host the database for the application. During this meeting the details for the server allocation and setup were finalized, allowing the group to start developing the final version of the database system. Thus, the backend setup for the application was heavily worked on for this iteration but was not completed.

Additionally, the team also had their first meeting with MASH, where the progress of the application and the features that they wanted to see in the application were discussed. The most important decision was how to measure the success of recovery in MASH certified sober homes; it was decided that success would be based on the events that users post to their timeline, which include length of time sober, length of time at the sober home and length of time in a current job, among other metrics. Another key part of the discussion was agreeing on what information to include regarding timeline events and the polling system. For timeline events this included goals that would activate every 30 days for sobriety and time spent in a sober home. Questions regarding experiences in non-MASH certified homes in the polling was suggested, but the team was unsure whether to add these at this point in the project.

During the MASH meeting a couple of small features were also suggested that were great additions to the application. A portal to a list of sober homes in Massachusetts was brought up

which was planned to be implemented in a future sprint. The ability to opt out of sensitive questions during registration and polling was also suggested, and this option was implemented during this iteration.

Besides features related to the meetings that took place, the material design UI was partially implemented, including the login screen and the home screen. The planning for the redesigned UI continued during this iteration as well, and was not fully completed at this point. The UI was mocked up using Adobe XD as a prototyping tool to test and refine the new design before implementing it in the application. The registration form was completely remade and changed from a two page design to a three page design. This change was made to make registration more personal, easier to comprehend, and to collect the necessary data requested by the sponsors.

The supervisor activity was also updated to include the ability for a supervisor to approve resident requests to join the sober home. The UI for this task was updated so that all supervisor functions were put on a single scrollable task bar, allowing additional functions to be added in the future more easily and to make more room for the UI of each individual task that can be done on the supervisor screen.

## Iteration 7

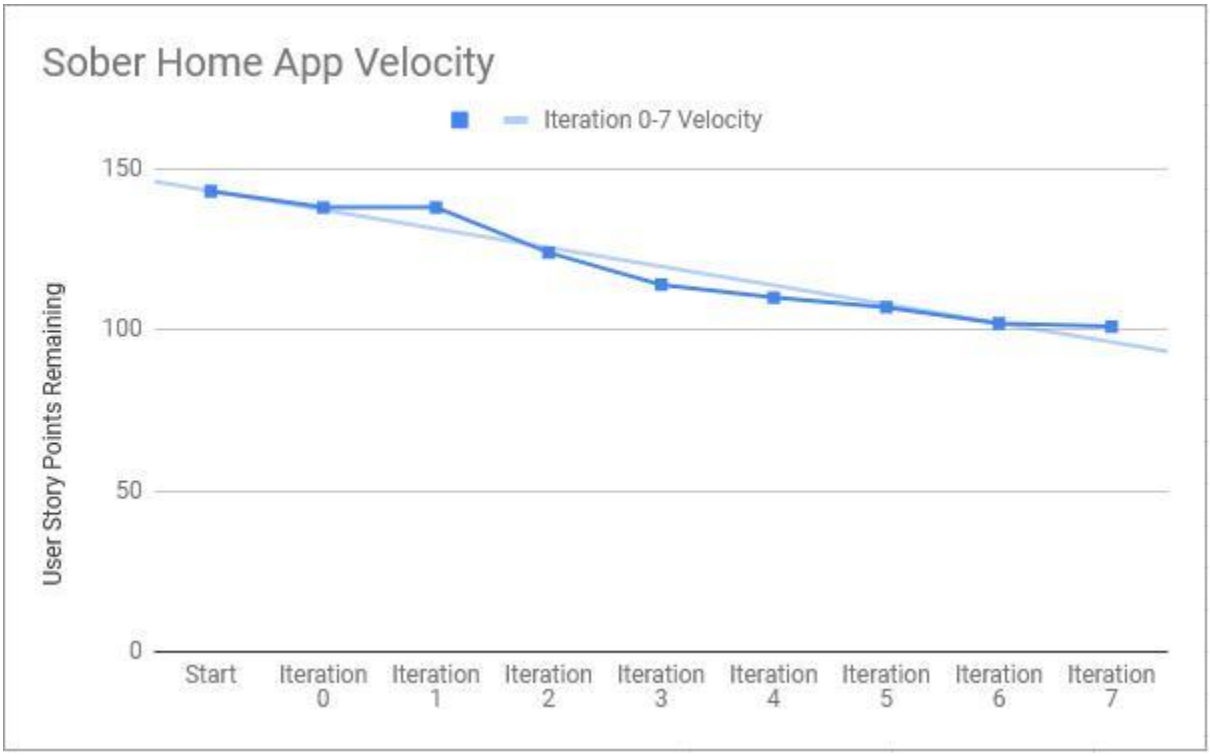



Figure 19. Iteration 7 Velocity Chart

During this iteration one user stories was completed - the portal for to the list of Massachusetts sober homes.

- As a RESIDENT I want to [see where local sober homes are located] so that I may [know which sober homes I can go to]: 1 point

The sprint velocity was 1 point, and the project velocity was 5.25 points. Due to larger user stories being worked on, such as the material UI redesign and the start of the ORM, the velocity for this sprint was low.

Due to how the beginning of the semester was scheduled, this iteration was quite short and not as much work was done as in other iterations. During this iteration, the portal to Massachusetts sober homes that was discussed in the MASH meeting from the previous sprint

was implemented. This was incorporated as a button on the login screen so that even people who have not registered for an account can have access to the list.

The group decided to switch to using an Object-Relational Mapping (ORM) interface for the database. Instead of making raw SQL queries, this interface allowed SQL queries to be generated from functions inside the ORM. SQLAlchemy, the ORM framework the team used, models tables as classes and relations as fields in the relevant classes. This allowed the team to interact with the database with an object-oriented mindset. As the number of tables and endpoints grew, the development time required to add new features to the backend started to increase exponentially. Using an ORM removed the need to think about crafting, testing, and optimizing raw SQL queries for each additional feature, some of which required multiple queries to fulfil a request. The ORM also enabled faster local development because it provided a programmatic way to generate all the tables and relations needed for the database.

The UI redesign was further improved, with the entire application being planned in Adobe XD. All activities and forms were completely redone to follow a Material color palette and to be more intuitive to users. Buttons were made to follow a consistent size, location, and color. The full implementation was planned for the following iteration.

# Iteration 8

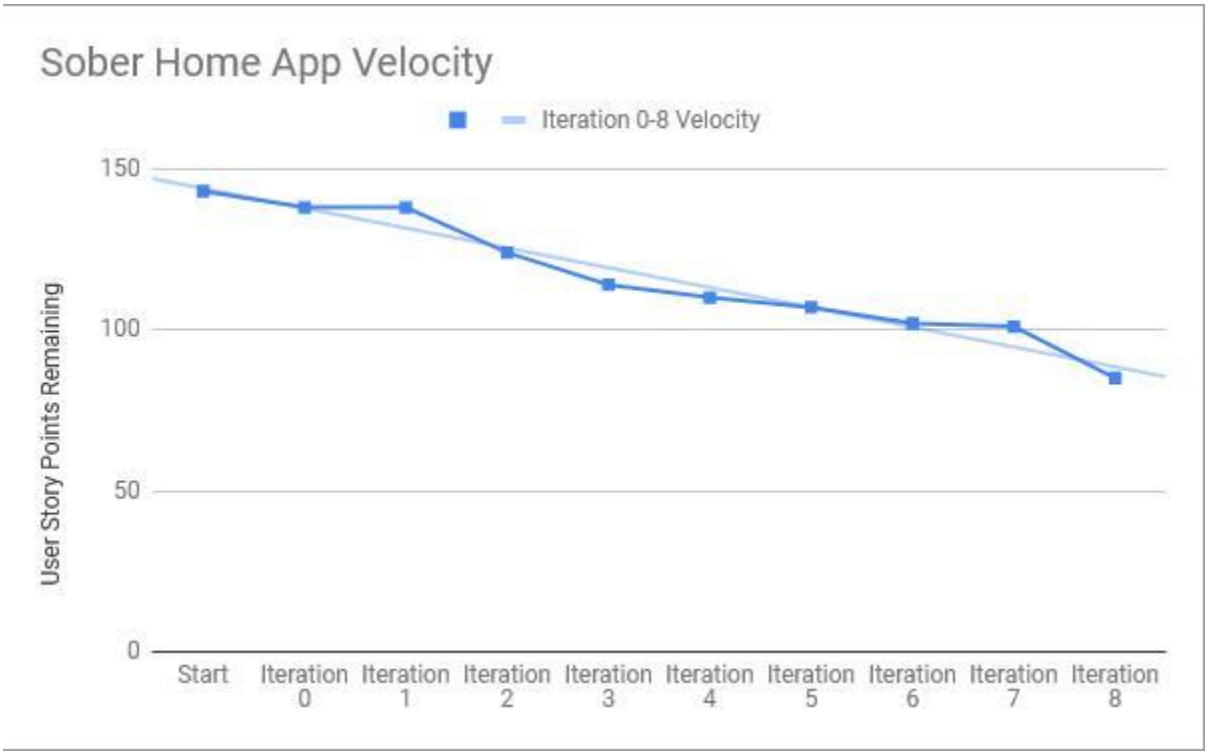



Figure 20. Iteration 8 Velocity Chart

During this iteration two user stories were completed - tracking the number of days sober and displaying milestones, and displaying polling questions to residents occasionally for additional data collection.

- As a RESIDENT I want to [have a count showing how long I have been sober] so that I may [be motivated by the progress that I have made]: 8 points
- As an APP OVERSEER I want to [collect information through periodic polls] so that I may [know the state of the residents across houses]: 8 points

The velocity for the sprint was 16 points, and the project velocity was 6.44 points.

This iteration saw the completion of many of the important backend features such as the ORM and its implementation with the RESTful API. Development of the web application went into full swing as well.

The ORM that the group decided on developing in the previous sprint was completed during this sprint, allowing easy database implementation for the mobile and web applications. It was also connected to the RESTful API that was created for the application server. The NGINX file was completed to account for the new set-up of the backend.

The web application was developed for the first time since the fourth sprint to prepare it for ORM integration. A login system was created so that users could not access the data from the mobile application unless they used a valid username and password from an admin account. Once the admin logged in, access to the data and statistics was enabled. In the next iteration this login was removed due to completely refactoring the web application framework.

The UI redesign was fully implemented in this iteration, with all activities, forms, and fragments updated to follow the new look. This created a consistent user experience within the application and ensured all parts of the app appeared cohesive. Many menus were redesigned to be more clear and intuitive in their operation. A basic logo was also designed for the application. Overall the redesign made the application easier to use, provided a clear consistent look, and increased the functionality of the application both for users and administrators. The UI in this iteration was also changed to ensure full compatibility on all Android devices of different screen sizes. Each UI element was updated so the constraints were dynamic and would automatically resize on different devices.

# Iteration 9

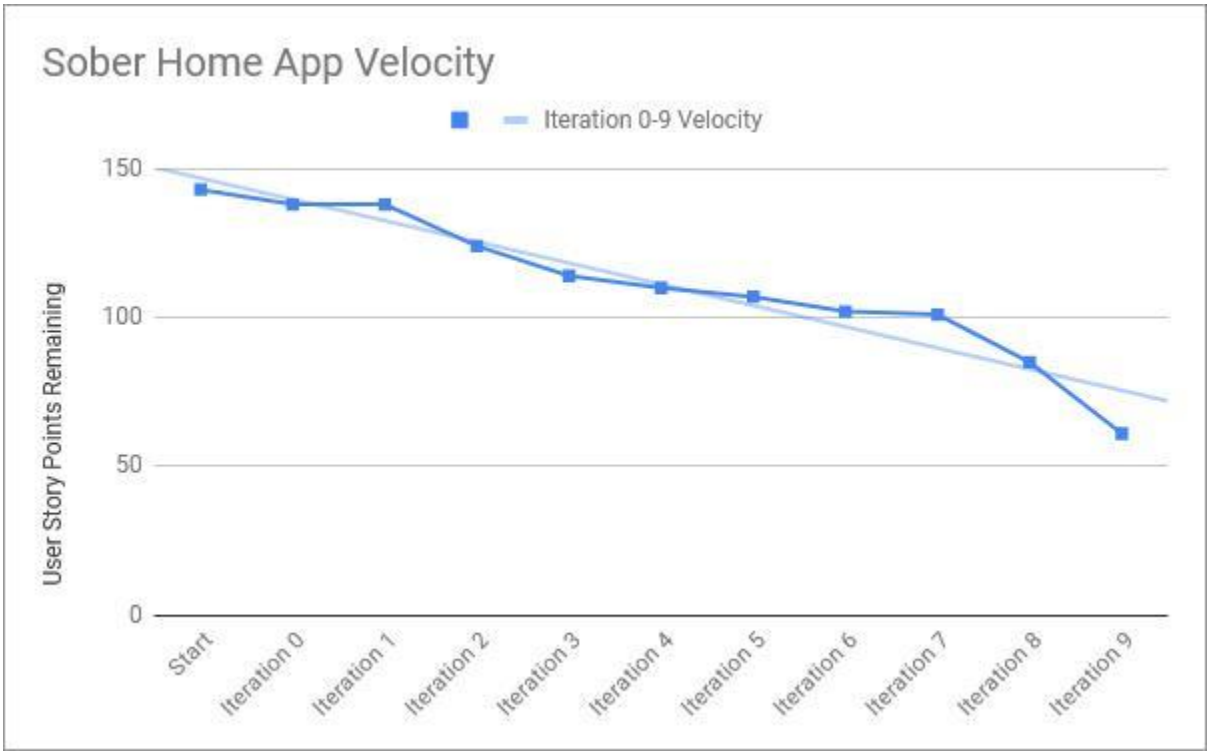



Figure 21. Iteration 9 Velocity Chart

During this iteration, seven user stories were completed - different account tiers for residents and sober home owners, the ability for residents to browse sober homes, the ability for residents to request access to a sober home, making data downloadable in JSON and CSV in the web application, allowing data in the web application to be sorted by home, displaying amount of goals met from residents' timelines and displaying dynamic visualizations on the web application based on resident data.

- As a HOUSE OWNER I want to [have a separate tier of account] so that I may [perform supervisory functions for my houses]: 4 points
- As a HOUSE OWNER I want to [approve new residents] so that I may [regulate who joins the application]: 4 points

- As a RESIDENT I want to [see where local sober homes are located] so that I may [know which sober homes I can go to]: 4 points
- As an APP OVERSEER I want to [access data in usable formats] so that I may [perform further analysis]: 4 points
- As an APP OVERSEER I want to [see visualizations of the data] so that I may [get a quick overview of the data]: 4 points
- As an APP OVERSEER I want to [compare data between sober homes] so that I may [know which sober homes are doing better and why]: 2 points
- As an APP OVERSEER I want to [display percentages of goals met] so that I may [get an overview of the data]: 2 points

The sprint velocity was 24 points, and the velocity for the project during this iteration was 8.2 points.

Iteration nine was the final iteration of the project. Development on the mobile application drew to a close during this iteration as the team focused on creating database endpoints to add connectivity to the current features. Endpoints are pieces of code in the application that connect to the database allowing data to be correctly displayed and entered. These endpoints created the connected experience the application was designed around. The application was also refactored in places to allow the database connections to be simpler and to 'clean up' portions of code.

Outside of connecting the application to the database, the mobile app received minor UI updates including customized placeholder images for user profiles and sober home profiles. Additionally, the registration page was refined. Date input was switched to calendar dials for ease of use and fields were edited so they are always visible while the keyboard is active.

A final feature that was added to the mobile application was an activity to browse and send a request to join a sober home. A RecyclerView was used to display all sober homes that were present in the database, and when one was pressed it brought the user to a detailed description of the home. This details screen was a refactored version of the sober home activity that was developed previously, and kept many of the same fields. An addition to the home page was a button that would send a request to join the home. This request would appear in the supervisor activity for the owner of the house.

The web application also saw improvements with the ability to view a number of charts and graphs. The charts were intended to show a brief overview of the underlying data. The charts were dynamically generated, meaning that they display the most recently data available in the database. The data displayed on the web app was also improved in this iteration by adding the data from the events logged by each resident. This data was particularly valuable because it pertained to the residents' paths of recovery rather than just their demographic backgrounds.

## Iteration Addendum

While the team was able to successfully implement the majority of the user stories in the project, there were some user stories that saw partial implementation. These features were added to the application but did not see total completion. The main partially completed user stories include the sober home rating system, an emergency contact information page, application notifications, and application settings.

The sober home rating system appears on the sober home pages in a five-star format, however there is currently no place for residents to submit a rating. The emergency contact information activity was created, however it was removed from later iterations due to design issues and difficulty in providing correct location-based phone numbers. An application notification framework was created and added to the application, however it was not developed to the point of actively pushing the notifications to users. Finally, an application settings page was introduced in an early iteration, but it was updated as a low priority feature and when the application was completed, the team determined that there was no time to create adjustable application settings.

# Testing

## Unit Testing

The testing framework that the team used is known as ‘Robolectric’ and is recommended by the Android documentation.  This framework allowed the team to test features using the Android SDK without having to launch an emulator or building to a device, which can take a long time. The development team created tests for features to cover corner cases in features they were creating as time went on.

## Integration Testing

During integration of feature branches the team manually tested the features being merged to ensure that they were working in the master build. In testing these new features in the master branch the team made sure that database queries still operated the same as they did in the feature branch, that buttons and interactivity still worked the same way, and so on. Integration was done using a top-down approach, stubbing the lower levels for testing, as each feature branch was merged with each other the above procedure was done.

## Generating Fake Data

To test the web visualizations, the team needed a way to generate large sets of data. To accomplish this, the team used a Python library, ‘faker’. Using faker, the team described the types of fields needed and specified any restrictions such as length or possible value, and then the library generated data within those parameters. This data was uploaded to the server and was then pulled from the database to the web server via the RESTful API.

## Manual Testing

While integrating new features into the application, the team made sure to manually test common workflows to ensure that the feature felt right to interact with. For example, during registration, the team wanted to ensure that the transitions between questions and activities felt natural to encourage users to complete the process. When switching over from the local datastore to a remote database, the team took the time to test each networked feature to see if the size of the responses that were designed took too long to return, causing the experience of the app to slow down. In addition to personal testing during development, the team received help from students to try and identify designs in the app that did not feel intuitive. Using this feedback, user tests continued on the workflows and the team made sure that the structure of the features did not interfere with the user's goal.

# Assessment

At the start of the project the team hoped to accomplish a few goals; create a mobile application, have that application collect data that would be useful for MASH administrators, and have the features of the application help residents of sober homes in their recovery.

Throughout the numerous iterations of this project the team learned many things. The project challenged the team to combine numerous languages, learn development techniques for a new platform, apply proper design patterns and AGILE methodology, and manage a long term development life cycle.

The project combined four main languages throughout a mobile application, RESTful API server, and web application in order to provide a suite of tools to assist in a resident's personal recovery as well as collect aggregate data for MASH administrators to analyze.

The project began with the team choosing the tools to use for the duration of the project. It required the team to investigate all of the potential options and then not only choose but justify why each tool was the best of for the task.

During development the team practiced AGILE methodologies to manage the project and iteratively create the application from the ground up, focusing on basic structure initially and fine-tuning design at the end. While the mobile application was being built, an entire database was setup on a WPI hosted server that required the team to combine their mobile application knowledge with that of networking and databases. Furthermore, the team built a web application for the mobile application, so web administrators could view the data collected by the application.

Finally, the team was able to demonstrate the working mobile application to the sponsors and document the entire project timeline, leaving documentation of the application’s

development along with recommendations so that any future projects would benefit from the data collection, development, and techniques used by the team.

The project overall went very smoothly with only few hiccups in each stage. At the very beginning with the start of the data collection and requirements gathering, the team was able to quickly focus on the important section of the project which became the foundation for the application. The team did run into the issue of sometimes vague and unclear application requirements from the project sponsors but was able to persevere and forge requirements from their understanding of the presented problem which lead to a successful application.

When developing the application the team was able to, with little interruption, work each week presenting a new and improved iteration of the application. There were some weeks where an iteration saw little progress and had to be continued into the following week, however those weeks were often around holidays or school term breaks. When an iteration was not fully completed the user stories were moved into the following weeks iteration with the team completing both the new user stories as well as the previous iterations.

Finally when nearing the end of the project, the team was able to make the effort necessary to finalize a finished product for the sponsors of the project. The final iterations were a cohesive and comprehensive application that met the requirements set at the beginning of the project. Beyond the mobile application, the team's successful creation of a web application helped to build in more features for application supervisors to view data and perform administrative operations. At the conclusion of the project, the team's sponsors, MASH and Rally to Recovery, viewed a demonstration of the application and began to work with the team to transfer the project data to them with great interest. The sponsors were impressed with the team's work and wished that the project may be continued and eventually deployed in the future.

# Future Work

While the team was able to accomplish many goals during the course of the project there is room for the project to be continued. The team has the following recommendations for future projects on this topic.

First the team recommends ensuring requirements are gathered at the very start of the project. The team encountered communication problems with the sponsors due to a change in leadership and as a result, did not receive the sponsors requirements until nearly two-thirds of the way through the project. Due to the timing of receiving the sponsors requirements the team was forced to compromise on some features and was unable to implement others. Another result of the delayed requirements was a lack of clear direction when starting the development portion of the project. This, in turn, required the team to predict what features would be beneficial to the application and design the application based on their perception of what the sponsors would require.

Second, the team recommend bringing the application to iOS. The team chose to target Android due to its open ecosystem and larger market share. iOS is still a popular phone platform in the United States and therefore there will be some potential users who are unable to utilise the benefits of the application due to their phone's platform. By bringing the application to iOS all MASH residents will be able to use the app and benefit from the features it contains.

The team's third recommendation is to further develop and expand upon the web application. The development of the web application began near the end of the development phase of the project and is simple in design and features. A future project could be expanding the capabilities of the web application to be more dynamic, better displayed, with more administration features for viewing the data or regulating users inside the mobile application.

There is also potential for the web app's data to be used for true statistical analysis; something the team was unfamiliar with and unable to complete.

The team's fourth recommendation is to implement the security features of the mobile application and web application described in the paper. Due to time constraints and the delayed setup of the server, the team was not able to implement the previously discussed security features. As the application designed was a minimal viable product, and is not meant to store personal data in its current form, the security was scheduled to be implemented last. In a final production level application the security must be implemented.

The team's fifth and final recommendation is a number of development features to be implemented or expanded. The features are as follows:

1) Add employment status to the database to keep records of who is employed and for how long
2) Add timeline events to the path visualization
3) Support survey assignment to specific demographics of users inside the mobile application
4) Store the images uploaded in the application.

These features will further the application's functionality and make it more effective. The team was able to implement an outline of a notification system, and the team recommends a future project expanding and utilizing the notification feature. The team has created the infrastructure to automatically add milestones to a user's timeline, however the server connection is not complete. The team also recommends expanding the chat features to increase house connectivity; potentially using temporary or noSQL data storage.

# Conclusion

Overall, the project experience was beneficial for both the members of the development team and for the project's sponsors. The group was able to gain first-hand experience designing a large-scale mobile application and a companion web application, while utilizing frameworks such as Robolectric. Through developing the application, the team gained a greater understanding of Android development and mobile interfaces. Even though there were difficulties gathering requirements for the application, the team was able to learn a great deal about communicating with clients that can be used in their future endeavors, and requirements have been gathered for future development of the application. This project was beneficial to the sponsors by showing them what can be done to help them gather information on how sober houses are running. The application that was produced will help the sponsors of the project to elicit more features that can be added in the future and will pave the way for improving support for sober home residents.

# References

About SQLite (n.d.). Retrieved September 12, 2018 from https://www.sqlite.org/about.html

Alexander, M. (2018). Agile Project Management: A Comprehensive Guide. CIO. Retrieved From https://www.cio.com/article/3156998/agile-development/agile-project-management-a-beginners-guide.html

Alshenqeeti, H. (2014). Interviewing as a Data Collection Method: A Critical Review. English Linguistics Research, 3(1), 39-45. Android Developer (2018) Kotlin and Android. Retrieved October 7, 2018 from https://developer.android.com/kotlin/

Android Studio (2018, September 11) Use Java 8 Language Features. Retrieved October 7, 2018 from https://developer.android.com/studio/write/java8-support

APMG International. (2017). Why is Agile becoming so popular in project management? Retrieved from https://apmg-international.com/article/why-agile-becoming-so-popular-project-management

Asri, V. (2018). *Timeline-View*. Vers. 1.0.6. *Android Arsenal*, https://android-arsenal.com/details/1/2923#!description

Beck, K. B., Mike; Bennekum, Arie; Cockburn, Alistair; Cunningham, Ward; Fowler, Martin; Grenning, James; Highsmith, Jim; Hunt, Andrew; Jeffries, Ron; Kern, Jon; Marick, Brian; Martin, Robert; Mellor, Steve; Schwaber, Ken; Sutherland, Jeff; Thomas, Dave. (2001). Principles behind the Agile Manifesto. Retrieved from http://agilemanifesto.org/principles.html

Binstock, A. (2015, May 20). Java's 20 Years Of Innovation. Retrieved September 23, 2018, from https://www.forbes.com/sites/oracle/2015/05/20/javas-20-years-of-innovation/#79edd75811d7

Breslav, A. (2016, February 15). Kotlin 1.0 Released: Pragmatic Language for JVM and Android. Retrieved from https://blog.jetbrains.com/kotlin/2016/02/kotlin-1-0-released-pragmatic-language-for-jvm-and-android/

Broggio, B. (Producer). Agile Processes - Scrum. [Presentation] Retrieved from https://www.unf.edu/~broggio/cen6016/Lecture%2012%20-%20Agile%20Processes-Scrum.ppt

Certification Standards. (2016). Retrieved from https://mashsoberhousing.org/standards-ethics/narr-quality/

CHESS Health. (2018, July). Connections: A-CHESS Platform - Apps on Google Play. Retrieved September 13, 2018, from https://play.google.com/store/apps/details?id=com.cmh.achessapp&hl=en

Daimajia, V. (2017). *Android View Animations*. Vers. 2.3. *GitHub*, https://github.com/daimajia/AndroidViewAnimations

Data and File Storage. (2018). Retrieved October 8, 2018 from https://searchdatabackup.techtarget.com/definition/asynchronous-replication

Data Brief: Opioid-Related Overdose Deaths Among Massachusetts Residents. (2018). Retrieved from https://www.mass.gov/files/documents/2018/08/24/Opioid-related%20Overdose%20Deaths%20among%20MA%20Residents%20-%20August%202018_0.pdf

Database Market Share (2018). Retrieved from https://www.datanyze.com/market-share/databases/Alexa%20top%201M/

DB-Engine Ranking (2018). Retrieved September 25, 2018 from https://db-engines.com/en/ranking

Development Process. (2012). Retrieved from https://sumatosoft.com/software-development-lifecycle

Dunleavy, K. (2017). 12 Best Practices for Creating Effective Surveys. Retrieved from MovableInk website: https://movableink.com/blog/12-best-practices-for-creating-effective-surveys/

Eason, J. (2015, June 26). An update on Eclipse Android Developer Tools. Retrieved from https://android-developers.googleblog.com/2015/06/an-update-on-eclipse-android-developer.html

Facebook. (2018). Facebook SDK for Android. Retrieved December 17, 2018, https://developers.facebook.com/docs/android

Facebook. (2018). React Native · A framework for building native apps using React. Retrieved

October 8, 2018, from https://facebook.github.io/react-native/

Farvin Packeer Mohamed, S. B., Fauziah; Deraman Aziz. (2014). An Exploratory Study on Agile based Software Development Practices. *International Journal of Software Engineering and its Applications, 8*(5), 29. doi:10.14257/ijseia.2014.8.5.09

Futrell, R. T., Shafer, D. F., & Shafer, L. I. (2002). *Quality Software Project Management* (1st ed.). Prentice Hall.

40.3 Materialized Views. (n.d.). Retrieved October 8, 2018 from https://searchdatabackup.techtarget.com/definition/asynchronous-replication

Gartner. (n.d.). Global mobile OS market share in sales to end users from 1st quarter 2009 to 2nd quarter 2018. In Statista - The Statistics Portal. Retrieved October 7, 2018, from https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/.

Global Mobile OS Market Share 2009-2018, by Quarter. (2018). Retrieved from https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/

Gorman, M. M., Marinaccio, A., & Cardinale, C. (2010). Fair Housing for Sober Living: How the Fair Housing Act Addresses Recovery Homes for Drug and Alcohol Addiction. The Urban Lawyer, 42(3), 607-614.

Hughey, D. (2009). Agile Methodologies. Retrieved from http://www.umsl.edu/~hugheyd/is6840/agile.html

Hughey, D. (2009). The Traditional Waterfall Approach. Retrieved from http://www.umsl.edu/~hugheyd/is6840/waterfall.html

Horton, M. J. (2015). Six Sobriety Apps You Should Know About. Retrieved from

https://www.addiction.com/12575/six-sobriety-apps-you-should-know-about/

I Am Sober. (2018). Retrieved from https://iamsoberapp.com/

Lotz, M. (2013). Waterfall vs. Agile: Which is the Right Development Methodology for Your Project? Retrieved from https://www.seguetech.com/waterfall-vs-agile-methodology/

MA Sober Home Laws. (2016). Retrieved from https://mashsoberhousing.org/certification/ma-sober-homes-law/

Madziwa, M. (2016). Interviewing as a Data Collection Method. Retrieved from https://www.linkedin.com/pulse/interviewing-data-collection-method-munyaradzi-madziwa

Maldonado, L. (2018). Sober Living and Halfway Homes. Retrieved from ProjectKnow website: https://www.projectknow.com/research/sober-living/

MiKinley, Holly Lynne. (2003). SSL and TLS: A Beginners Guide. Retrieved from https://www.sans.org/reading-room/whitepapers/protocols/ssl-tls-beginners-guide-1029

MySQL :: MySQL Workbench & Utilities (n.d.). Retrieved December 1, 2018 from https://dev.mysql.com/downloads/tools/

National Institute of Standards and Technology. (n.d.) NIST Special Publication 800-53 (Rev. 4). Retrieved October 5, 2018 from https://nvd.nist.gov/800-53/Rev4/control/AC-6

National Security Agency. (2016). Commercial National Security Algorithm Suite and Quantum Computing FAQ. Retrieved from https://cryptome.org/2016/01/CNSA-Suite-and-Quantum-Computing-FAQ.pdf

Open Web Application Security Project. (2010). Retrieved from https://www.owasp.org/index.php/SQL_Injection

Opioid Overdose Crisis. (2018). Retrieved from

https://www.drugabuse.gov/drugs-abuse/opioids/opioid-overdose-crisis

Phillips, Bill, et al. *Android Programming: The Big Nerd Ranch Guide*. Version 3. Big Nerd Ranch, 2017.

Radigan, D. Kanban. Retrieved from https://www.atlassian.com/agile/kanban

React Native · A framework for building native apps using React. (n.d.). Retrieved September 24, 2018, from https://facebook.github.io/react-native/

Real Recovery (2017). The Difference Between Sober Living and Halfway Houses. Retrieved from https://myrealrecovery.com/difference-between-sober-living-and-halfway-houses/

recoveryBox. (2014). Retrieved from https://recoveryboxapp.com/

Retzlaff, D. (Producer). (2013). Agile Development Using Scrum. [Presentation] Retrieved from https://nces.ed.gov/whatsnew/conferences/MIS/2013/ppt/X_J_Retzlaff.pptx

Rouse, Margaret; Posey, Brien. (2015). Asynchronous Replication. Retrieved October 6, 2018 from https://searchdatabackup.techtarget.com/definition/asynchronous-replication

Saltzer, J. H., Reed, D. P., & Clark, D. D. (1984). End-to-end arguments in system design. ACM Transactions on Computer Systems (TOCS), 2(4), 277-288.

SDLC - Waterfall Model. (2018). Retrieved from https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm

Segijn, D. (2018). *Konfetti*. Vers. v1.1.2. *Android Arsenal*, https://android-arsenal.com/details/1/5884

Serenity House Health (2018) Retrieved from https://play.google.com/store/apps/details?id=com.caredfor.shd

Shaughnessy, Pat (2014). Following a Select Statement Through Postgres Internals. Retrieved

from
http://patshaughnessy.net/2014/10/13/following-a-select-statement-through-postgres-internals

Singh, A., Taneja, A., & Mangalaraj, G. (2009). Creating Online Surveys: some wisdom from the trenches. IEEE Transactions on Professional Communication, 52(2), 197-212.

Sober Grid. (2017). Retrieved from https://www.sobergrid.com/

Sober Living App. (2017). Retrieved from https://soberlivingapp.com/

Stackoverflow. (2018) Newest Java Questions. Retrieved October 7, 2018 from https://stackoverflow.com/questions/tagged/java

Stackoverflow. (2018) Newest Kotlin Questions. Retrieved October 7, 2018 from https://stackoverflow.com/questions/tagged/kotlin

Standards. (2016). Retrieved from https://mashsoberhousing.org/standards-ethics/

SQLite Frequently Asked Questions (n.d.). Retrieved December 10, 2018 from https://www.sqlite.org/faq.html

Taymor, E. Agile Handbook. 1-34.

What is a Bottleneck and How to Deal With It? (2018). Retrieved from Kanbanize website: https://kanbanize.com/lean-management/pull/what-is-bottleneck/

What is Android SDK? (2018). Retrieved from Techopedia website: https://www.techopedia.com/definition/4220/android-sdk

What is a Sober Home. (2016). Retrieved from https://mashsoberhousing.org/what-is-a-sober-home

What is Kanban? (2018). Retrieved from https://resources.collab.net/agile-101/what-is-kanban

What is SDLC Waterfall Model? (2018) Retrieved from https://www.softwaretestinghelp.com/what-is-sdlc-waterfall-model/

What's the Difference? Agile vs Scrum vs Waterfall vs Kanban: Agile Methodology. (2018). Retrieved from https://www.smartsheet.com/agile-vs-scrum-vs-waterfall-vs-kanban

# Glossary

Agile Methodology - A project management methodology using incremental, iterative, fixed-length sprint cycles as its foundation for development. In each sprint cycle, a subset of user stories is completed following the steps of SDLC.

Android - An operating system overseen by Google that is used by the majority of mobile devices.

Android Application Package (APK) - A file format used to distribute Android applications to Android-compatible devices.

GitHub - A web service used to store code repositories. Also contains built-in issue tracking and project management tools.

Halfway House - An institution housing people who are recovering from an addiction or a criminal background that are typically government funding. This type of housing tends to be more affordable and dorm-like.

Integrated Development Environment (IDE) - A framework of tools used to aid in the writing of code, consisting of source code editor, compiler, and debugger for checking syntax errors and debugging issues.

iOS - A proprietary operating system by Apple that runs on all Apple mobile devices.

Java - A programming language that was created by Sun Microsystems in 1995 that was designed to be object-oriented and have as little dependencies as possible.

Kanban - A subset framework of Agile project management methodology. The goal of this method is continuous delivery, where a Kanban board is used to organize tasks, allocate resources, and identify work bottlenecks.

Kotlin - A programming language that was created by JetBrains in 2011 designed to work with existing Java resources and to be concise.

Massachusetts Alliance for Sober Housing (MASH) - An organization dedicated to promoting critical management, operational, and ethical standards for sober homes in Massachusetts. MASH helps sober homes meet these standards needed to gain certification.

Mobile Application - A program that runs on a handheld device such as phone or a tablet.

Model-View-Controller (MVC) - An architectural pattern in software that divides the application into three segments: model, view and controller. The model portion controls the data-flow within an application, the view portion controls the user interactivity of the application, and the controller manipulates data and processes requests between the model and the view.

Narcotics Anonymous (NA) - An organization that facilitates help meetings for people recovering from drug addiction.

Opioid - A class of drugs that is used to reduce pain and is based on opium-like substances, which includes illegal drugs like heroin and prescription pain relievers, such as oxycodone and morphine.

Object Relational Model (ORM) - Framework that allows developers to describe database tables as classes and interact with database data as if they were objects.

Rally 2 Recovery - An organization dedicated to raising awareness of the opiate epidemic and helping those whose loved ones are struggling with addiction.

React Native - A programming framework developed by Facebook that uses JavaScript for creating mobile applications.

RSA Keys - A security technology that provides secure authentication for a user. An unique key is required for each user.

Scrum - A subset framework of Agile software development methodology that involves the use of development cycles (Sprints) and daily ‘scrum meetings’ as a team to discuss what was accomplished the previous day, what were the challenges encountered, and what will be accomplished that day.

Software Developer’s Kit (SDK) - A set of software development tools that are typically used to create applications based on software packages or frameworks.

Secure Sockets Layer (SSL) - A security technology that creates an encrypted link between a client and a server.

Sober Home - An institution housing people recovering from an addiction that are typically affiliated with addiction treatment centers. The main focus of this type housing is to provide a safe living environment that tends to be more home-like than halfway houses.

Software Development Lifecycle (SDLC) - A process for developing software intended to create the highest quality deliverable in the shortest time and with the lowest cost while meeting or exceeding customer expectations. This process is categorized into seven phases: Planning, Requirements Gathering, Design, Development, Testing, Deployment, Maintenance.

Sprint Cycle - A development cycle that lasts a constant interval of time for a phase of a project, targeting a set of features (or a subset of user stories). Each time one sprint cycle finishes, another one is started. A working deliverable is expected after every sprint cycle in the Scrum-Agile methodology.

User Story - A description identifying one or more software features from the end-user’s perspective, which is used primarily in the Agile methodology written in the format: As a (role) I want (something) so that (benefit)

Version Control - A system that organizes changes over time and maintains different versions of the code.

Waterfall Methodology - A project management methodology where the phases of the SDLC are completed one at a time sequentially for the entire project. The next phase can only be worked on once the previous phase is completely finished.

Web Application - A program that runs in a web browser and can be used on any device that supports a web browser.

# Appendix A – MQP Sober Home Survey Questions

Consent Agreement on Participation in Sober Home Study

You have been invited to participate in a Worcester Polytechnic Institute (WPI) sponsored Major Qualifying Project (MQP) concerning Sober Homes. The goal of this research is to validate and optimize the effectiveness of sober homes through the use of a mobile application.
Your participation will provide useful data regarding which features and considerations should be taken into account during the application development process. Information gathered in this survey will remain completely anonymous. No names or identifying information will be recorded.
Participation in this survey is voluntary. You may choose to stop this survey at any time. You may skip any question without consequence.

By clicking ">>" below you are consenting to participate in this survey. Thank you once again for supporting this project. If you have any questions regarding our study, feel free to contact our project advisor Wilson Wong (wwong2@wpi.edu). For questions regarding your rights as a participant, please contact the Human Research Protection Program at WPI (irb@wpi.edu).

What is your age?

- ○ < 18 Years Old
- ○ 18-25
- ○ 26-30
- ○ 31-35
- ○ 36-40
- ○ 41-45
- ○ 46+

Length of addiction:

- ○ < 1 month
- ○ 1 month - 6 months
- ○ 6 months - 1 year
- ○ 1 year -3 years
- ○ 4 years - 6 years
- ○ 6+ years

Length of stay in current sober home

- ○ < 1 month
- ○ 1 month - 6 months
- ○ 6 months - 1 year
- ○ 1 year - 3 years
- ○ 4 years - 6 years
- ○ 6+ years

What kind(s) of support would be or has been encouraging for recovery?

_____

What kind(s) of support would be or has been ineffective for encouraging for recovery?

_____

How often do you use your phone on an average day?

- ○ Never
- ○ Very Rarely
- ○ Rarely
- ○ Occasionally
- ○ Frequently
- ○ Very Frequently

Since you've had a smartphone, what is the longest period of time you've been without access to one?

- ○ Less than a day
- ○ Less than a month
- ○ 1-3 Months
- ○ Less than 6 months
- ○ Less than a year
- ○ Longer than a year

How likely would you be to sell your phone?

- ○ Extremely unlikely
- ○ Somewhat unlikely
- ○ Neither likely nor unlikely
- ○ Somewhat likely
- ○ Extremely likely

How well would you say you know the people you live with?

- ○ Not well
- ○ Slightly not well
- ○ Moderately well
- ○ Slightly well
- ○ Very well

What kind of phone do you have?

- ○ Android Phone
- ○ iPhone
- ○ Windows Phone
- ○ Other
- ○ I do not own a phone

What phone application(s) do you find easy to use?

_____

_____

_____

_____

_____

What makes that applications easy to use?

_____

_____

_____

_____

_____

What phone application(s) do you find difficult to use?

_____

_____

_____

_____

_____

What makes those applications difficult to use?

_____

_____

_____

_____

_____

What feature(s) would you like to see in a phone application for sober home residents?

_____

_____

_____

# Appendix B – Interview Questions

Hello (Name of Participant), thanks for coming! Please take a seat.

We are conducting a Worcester Polytechnic Institute sponsored project to make a mobile application for sober homes in Massachusetts. The goal of this project is to validate and optimize the effectiveness of sober homes through the use of this application.

Your participation will provide useful data regarding which features and considerations should be taken into account during the application development process. Information gathered in our study will remain completely anonymous. No names or identifying information will be recorded.

Participation in this interview is voluntary. You may choose to stop this interview at any time. You may choose not to answer any questions without consequence.

Thank you for consenting. We will now start the interview with some basic questions:

**MASH Sober Housing / Rally-2-Recovery Coordinators:**

1. What goal(s) do you have in mind for the application to accomplish?
2. What information would you like the application to collect?
3. What features would you like the application to have?
4. What requirements do you have for the application?
5. From your experience, what type of support do you think is most effective to residents?
6. What is your vision for how this application will support your operations?

# Appendix C – Product Backlog

| Scale: |
| --- |
| 1 - less time |
| 2 - twice 1 |
| 4 - twice 2 |
| 8 - twice 4 |
| |

| User Story: | Points: | Iteration: |
| --- | --- | --- |
| Account creation form/view | 4 | 0 |
| Login functionality | 1 | 0 |
| Ability to set account public or private | 2 | |
| Ability to set a profile picture, otherwise provide a default anonymous photo | 2 | 1 and 2 |
| Ability to change account information | 2 | |
| Multiple tiers of accounts for residents and home owners | 4 | 9 |
| | | |
| Ability to create a post for a resident's timeline | 8 | 2 |
| Ability to see own and other residents' timeline | 4 | 2 |
| Built in 'achievements' that show up on a resident's timeline | 4 | |
| Custom goals that resident can set that when completed show up on timeline | 4 | 2 |
| Ability to like a post on a resident's timeline | 2 | |
| Track days sober and display milestones | 8 | 8 |
| | | |
| Chat feature that allows users to chat with other users | 4 | 3 |
| Different chats for global, house and owners | 4 | 3 |
| Have events that are public appear in chats for a user | 2 | 3 |
| Integration with another social media service | 8 | |
| | | |
| Ability for sober home owner to create a sober home | 2 | 4 |
| Ability for sober home owner to accept/reject resident housing requests | 2 | 4 and 5 |
| Ability for resident to rate a sober home | 1 | |
| Reasoning for why resident was removed from a home | 1 | 4 and 5 |
| Ability for sober home owners to approve of resident events | 4 | 6 |
| Psuedo-role of home alumni so that users can be part of the house chat after they leave | 4 | |

| User Story: | Points: | Iteration: |
|---|---|---|
| Ability for users to request to join a sober home. | 4 | 9 |
| | | |
| Recommend nearest sober homes for residents | 8 | |
| Ability for residents to see list of sober homes | 4 | 9 |
| Recommend nearest NAH meetings for residents | 8 | |
| Emergency information page | 1 | 1 |
| Emergency button/contact for residents | 2 | |
| Portal to list of MA sober homes on launch activity | 1 | 7 |
| | | |
| Ability for overseers to compare data between certified and non-certified homes | 8 | |
| Ability for overseers to see why residents were kicked out of a sober home | 4 | |
| Make data available in JSON and CSV format | 4 | 9 |
| Web app setup server and index | 2 | 4 |
| Web app login | 1 | 8 |
| Allow data to be sorted by home | 2 | 9 |
| Display dynamic visualizations based on data | 4 | 8 and 9 |
| Ability to see why residents were removed from homes | 2 | |
| | | |
| | | |
| Opt out of sensitive questions | 1 | 6 |
| Display goals met on web app | 2 | 9 |
| Survey questions | 8 | 8 |

# Appendix D - MASH Guidelines

Retrieved from: ("MA Sober Home Laws," 2016)

**Core Principle: Operate with Integrity**

1. Are guided by a mission and vision

1.1 A written mission statement that corresponds with NARR's core principles

1.2 A vision statement that corresponds with NARR's core principles as stated in this document

2. Adheres to legal and ethical codes

2.1 An affidavit that attests to complying with non-discriminatory state and federal requirements.

2.2 Marketing materials, claims and advertising that are honest and substantiated as opposed to:

False or misleading statements or unfounded claims or exaggerations;
Testimonials that do not really reflect the real opinion of the involved individual;
Price claims that are misleading;
Misleading representation of outcomes.

2.3 Prior to the initial acceptance of any funds, the operator must inform applicants of all fees and charges for which they will be, or could potentially be, responsible. This information needs to be in writing and signed by the applicant.

2.4 The operator must maintain accurate and complete records of all resident charges, payments and deposits. A resident must be provided with a statement of his/her personal charge and payment history upon request.

2.5 The operator must disclose refund policies to applicants in advance of acceptance into the home, and before accepting any applicant fees.

2.6 Staff must never become involved in residents' personal financial affairs, including lending or borrowing money, or other transactions involving property or services, except that the operator may make agreements with residents with respect to payment of fees.

2.7 Policy and procedure that ensures refunds consistent with the terms of a resident agreement are provided within 10 business days, and preferably upon departure from the home.

2.8 Policies and procedures that ensure all residents are age eighteen or older at time of admission.
3. Are financially honest and forthright

3.1 Identifying the type of accounting system used and its capability to fully document all resident financial transaction, such as fees, payments and deposits.

3.2 Policy and procedure for disclosing to potential residents their financial obligations, including costs for which they might become liable, such as forfeiture of any deposits and fees as a result of prematurely leaving the home.

3.3 Policies about the timing of and requirements for the return of deposits, if financial deposits are required.

3.4 The ability to produce clear statements of a resident's financial dealings with the operator (although it's not a requirement that statements be automatically produced).

3.5 Policies and procedures that ensure the follow conditions are met, if the residence provider or a staff member employs, contractors or enters into a paid work agreement with residents:

Paid work arrangements are completely voluntary. Residents do not suffer consequences for declining work. Residents who accept paid work are not treated more favorably than residents who do not.

Paid work for the operator or staff does not impair participating residents' progress towards their recovery goals.

The paid work is treated the same as any other employment situation.

Wages are commensurate with marketplace value, and at least minimum wage. The arrangements are viewed by the majority of the residents as fair.

Paid work does not confer special privileges on residents doing the work. Work relationships do not negatively affect the recovery environment or morale of the home. Unsatisfactory work relationships are terminated without recriminations that can impair recovery.

**Core Principle: Collect data for continuous quality improvement**

4. Collect data for continuous quality improvement

4.1 Procedures that collect resident's demographic information

4.2 Adoption of procedures that collect, evaluate and report accurate process and outcomes data for continuous quality improvement, once data items and protocols to be developed by NARR are adopted by MASH.

**Core Principle: Operate with Prudence**

5. Operate with Prudence

5.1 Documentation that the owner/operator has current liability coverage and other insurance appropriate to their level of support.

5.2 Written permission from the owner of record to operate a recovery residence on the property.

5.3 Document that there are no taxes or other municipal assessments that constitute liens on the real estate upon which the recovery residence is located by providing a municipal lien certificate issued by the treasurer or collector's office in the city or town in which the recovery residence is situated.

**Core Principle: Uphold resident rights**

6. Communicate rights and requirements before agreements are signed

6.1 A process that ensures residents receive an orientation on agreements, policies and procedures prior to committing to terms.

6.2 Written resident's rights and requirements (e.g. House Rules and grievance process) posted in common areas

6.3 Written resident agreement that includes recovery activities provided (required and optional), including house meetings.

6.4 Resident documents that fully disclose policies regarding possessions (personal property) left in a home.

7. Promote self and peer advocacy

7.1 Grievance policy and procedures, including the right to take grievances that are not resolved by the house leadership to the operation's oversight organization for mediation

7.2 Policy and procedure for identifying the responsible person(s) in charge to all residents

8. Support housing choice

8.1 Applicant screening policies and procedures provide current residents a voice in the acceptance of new members

8.2 Policies and procedures that defend residents' fair housing rights

9. Protect privacy

9.1 Policies and procedures that keep resident's records secure, with access limited to authorized staff only

**Core Principle: Are recovery-oriented**

10. View recovery as a person-driven, holistic and lifelong process

10.1 Demonstrating that residents participate in the development of their recovery including an

exit plan and/or lifelong plan

10.2 Documenting that the operator cultivates alumni participation

11. Are culturally responsive and competent

11.1 Policies and procedures that identify the priority population, which at a minimum includes persons in recovery from substance use but may also include other demographic criterion.

11.2 A staffing or leadership plan that reflects the priority population's needs

**Core Principle: Are peer staffed and governed**

12. Involve peers in governance in meaningful ways

12.1 Some rules made by the residents that the residents (not the staff) enforce?

12.2 A resident council or process is in place that ensures resident's voices can be heard

12.3 The resident council has a voice in the governance of the home

13. Use peer staff and leaders in meaningful ways as evidenced by: at least one of the following:

13.1 Residents' responsibilities increase with their length of stay or progress in their recovery.

13.2 Staffing or leadership plan that formally includes a peer component

13.3 Written job description or house manager duties and/or contracts for peer staff and leaders

14. Maintain resident and staff leadership based on recovery principles

14.1 A home staffing or leadership plan that includes current residents and where possible, former residents that model recovery principles

14.2 Leader and/or staff job descriptions and selections are based in part on modeling recovery principles

15. Create and sustain an atmosphere of recovery support

15.1 Integrated recovery support in the daily activity schedule

15.2 The schedule includes formal and informal opportunities for staff and resident interaction in support of recovery

16. Ensure staff are trained

16.1 Documentation that a house manager or operator in functioning as the house manager possesses an appropriate level of knowledge and understanding of the MASH standards and practices as evidenced by the completion of MASH training class Recovery Residence 101 and subsequent training as may be required by MASH from time to time

16.2 Written staffing or workforce development plan.
17. Provide supportive staff supervision

17.1 Policies and procedures for supervision of staff

17.2 Ongoing skills development, oversight and support policies and NARR procedures appropriate to staff roles and level of support

## Recovery Support Domain

**Core Principle: Promote health**

18. Encourage residents to own their recovery

18.1 Policies and procedures that encourage each resident to develop and participate in their own personalized recovery plan (Person-driven recovery)

18.2 Policies and procedures that encourage residents to make their own outside appointments
19. Inform and encourage residents to participate in a range of community-based supports

19.1 Staff that are knowledgeable about local community-based resources

19.2 Resource directories or similar resources are readily available to residents
20. Offer recovery support in informal social settings

20.1 Staffing plan that corresponds to the delivery of this service

20.2 Traditions, policies or procedures that foster mutually supportive and recovery-oriented relationships between residents and/or staff through peer-based interactions
21. Offers recovery support services in formal settings

21.1 Weekly schedule of recovery support services recognized by the respective NARR Affiliate organization

21.2 Weekly schedule of recovery-oriented presentations, group exercises, and activities

21.3 Staffing plan that corresponds to the delivery of this service

**Core Principle: Provide a home**

22. Provide a physically and emotionally safe, secure and respectful environment

22.1 Policies and procedures, such as applicant screenings, that establish the home's priority population and cultivate physically and emotionally safe environments for discussing the needs, feelings and sustaining recovery-supportive connections
23. Provide an alcohol and illicit drug-free environment

23.1 Written and enforced policies and procedures that address:

Alcohol and/or other prohibited drug-seeking or use
Possession of hazardous and other prohibited items and associated searches
Drug-screening and or toxicology protocols*
Prescription and non-prescription medication usage and storage consistent with the relevant state law

*Note: "The MassHealth agency does not pay for the following services: […] (4) tests performed only for purposes of civil, criminal, administrative, or social service agency investigations, proceedings, or monitoring activities; (5) tests performed for residential monitoring purposes; […] (9) test that are not medically necessary as defined in 130 CMR 450.204: Medical Necessity; …"130 CMR 401.411: Noncovered Services and Payment Limitations.
24. Are cultivated through structure and accountability

24.1 Written resident rights, requirements, agreements, social covenants and/or "House Rules"

24.2 Requirements and protocols for peer leadership and/or mentoring policies that foster individual and community accountability

**Core Principle: Inspire purpose**

25. Promote meaningful daily activities

25.1 A weekly schedule of the typical resident's activities

25.2 Are residents encouraged to (at least one of the following):

Work, going to school, or volunteer outside of the residence community
Participate in mutual aid or care giving
Participate in social, physical or creative activities
Attend daily or weekly programming

25.3 Person-driven recovery planning & peer governance

**Core Principle: Cultivate community**

26. Creating a "functionally equivalent family" within the household. As evidenced by meeting at least 50% of the following:

26.1 Are residents involved in food preparation?

26.2 Do residents have control over who they live with?

26.3 Do residents help maintain and clean the home e.g. chores?

26.4 Do residents share in household expenses?

26.5 Family or house meetings at least once a week?

26.6 Do residents have access to the common areas of the home?
27. Foster ethical, peer-based mutually supportive relationships between residents and/or staff

27.1 Encouraging residents to engage one another in informal activities and conversation

27.2 Encouraging staff to engage residents in informal activities and conversations

27.3 Coordinating community gatherings, recreational events and/or other social activities among residents and/or staff
28. Connect residents to the local (greater) recovery community

28.1 Residents are informed of or linked to mutual aid, recovery community centers, recovery ministries recovery-focused leisure activities and recovery advocacy opportunities;

28.2 Mutual aid meetings are hosted on site and there are typically attendees from the greater recovery community

28.3 The recovery residence helps participants find a recovery mentor or mutual aid sponsor if they are having difficulty finding one

28.4 Participants are encouraged to find a recovery mentor or mutual aid sponsor before leaving the recovery residence

28.5 Residents are formally linked with the community such as job search, education, family services, health and/or housing programs

28.6 Residents engage in community relations and interactions to promote kinship with other recovery communities and goodwill for recovery services

28.7 Sober social events are regularly scheduled

## Property and Architecture Domain
**Core Principle: Promote recovery**

29. Create a home-like environment

29.1 Furnishing are typical of those found in single family homes or apartments as opposed to institutional settings

29.2 Entrances and exits that are home-like (vs institutional or clinical)

29.3 70 sq. ft. for first bed; 50 sq. ft. additional beds

29.4 One sink, toilet and shower per eight female residents and one sink, toilet and shower per ten male residents

29.5 Each resident has personal item storage

29.6 Each resident has food storage space

29.7 Laundry services are accessible within onsite or within walking distance to all residents

29.8 Working appliances

29.9 A staffing plan that provides for addressing repairs and maintenance in a timely fashion
30. Promote community

30.1 Community room (space) large enough to reasonably accommodate community living and meetings.

30.2 A comfortable group area, a living room or sofas, for participants to informally socialize

30.3 A kitchen and dining area(s) that encourages residents to share meals together

30.4 Entertainment or recreational areas and/or furnishings that promote social engagement

30.5 Furniture that is in good condition

**Core Principle: Promote safety**

31. Promote home safety

31.1 Affidavit from the owner or operator attesting that the residence meets nondiscriminatory local health and safety codes OR document from government agency or credentialed inspector attesting to the property meeting health and safety standards

31.2 Signed and dated safety self assessment checklist which includes:

Functioning smoke detectors in the sleeping rooms
Functioning carbon monoxide detectors
Functioning fire extinguishers in plain sight and/or clearly marked locations
Interior and exterior of the property is in a functional, safe and clean condition and free of fire hazards

31.3 Smoke-free living environment policy and/or designated smoking area outside of the residence

31.3 Naloxone (Narcan) available and accessible; evidence that staff and residents are oriented in its use
32. Have an emergency plan

32.1 Post emergency numbers, procedures and evacuation maps in conspicuous locations

32.2 Collect emergency contact information from residents and orient them to emergency procedures

## Good Neighbor Domains

**Core Principle: Are good neighbors**

33 Are compatible with the neighborhood

33.1 If recovery residence is in a residential neighborhood, there are no external indications that the property is anything other than a single family household typical of its neighborhood The property and its structures are consistently maintained
34. Are responsive to neighbor concerns

34.1 Policies and procedures that provide neighbors with the responsible person(s) contact information upon request

34.2 Policies and procedures that require the responsible person(s) to respond to neighbor's concerns even if it is not possible to resolve the issue

34.3 New resident orientation includes how residents and staff are to greet and interact with neighbors and/or concerned parties

35. Have courtesy rules

35.1 Policies that are responsive or preemptive to neighbor's reasonable complaints regarding:

Smoking
Loitering
Parking
Noise
Lewd or offensive language
Cleanliness of public space around the property

35.2 Parking courtesy rules where street parking is scarce