# Requirements Report - Dawson Psychological Services A

Michael Crouse, Meredith Lister, Lucas Pozza

COSC 499

21 October 2020

# Description

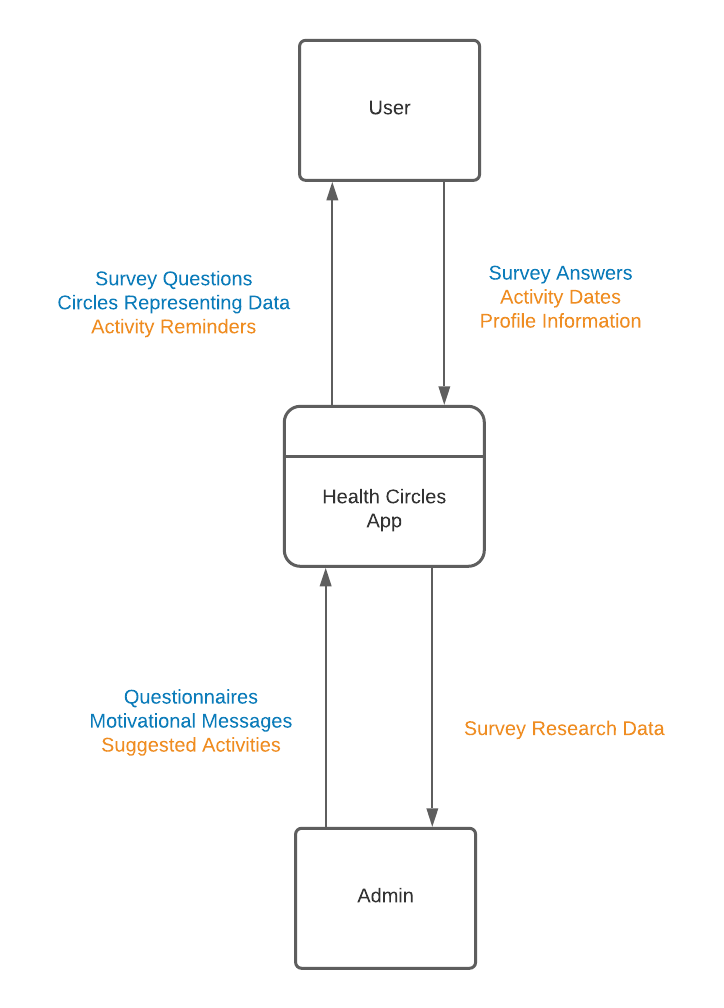
The Health Circles mobile app encourages users to have a holistic view of their own mental health. Users answer questions to quantifiably describe their well-being in several different domains (physical, mental, emotional, social, spiritual) which are represented visually as concentric circles of varying sizes. Based on their results, users are recommended activities to add to their calendar to improve their well-being. The administrator may add new questions or activity suggestions over time.

# User Groups

1. **Clients**: These users are clients of Dr Dawson and use the app for their own benefit, as well as to share insights with Dr Dawson.
2. **Other Users**: These are people who do not work with Dr Dawson but are interested in improving their own mental health.
3. **Administrator**: This user is Dr Dawson, who can add new questions or activity suggestions. He can also see information shared by users.
4. **Developers**: These users use the application for testing purposes.

# System Architecture

## Data Flow Diagram Level 0



The Level 0 diagram shows that the system at the Peer Testing #1 milestone receives survey answers from the user and questionnaires and motivational messages from the administrator. The user receives survey questions and circle visualizations representing their data from the system. At the Peer Testing #2 milestone, the user also provides dates for activities they choose to do, and their profile information when they register for an account. The administrator provides suggested activities for the user. From the system, the user receives notifications of activity reminders. The admininistrator receives anonymized survey research data.

Components for the Peer Testing #1 and #2 milestones are blue and yellow respectively.

## Data Flow Diagram Level 1Data Flow Diagram Level 1

The Level 1 diagram expands upon the Level 0 diagram to show that in Peer Testing #1, the questionnaires answered are stored locally on the device. Answering a questionnaire is defined as a process, because the answers inputted by the user are converted to numbers to represent their answers. For this milestone, we will populate data stores with activities and calendar events for UI design and testing purposes. The admin can input motivating messages and quotes to be displayed to the user into a data store.

For the Peer Testing #2 milestone, the user can input their information to register for an account, which is stored in the user profile data store. The results from questionnaires are quantified to determine the user’s well-being in each domain (social, emotional, physical, mental, spiritual). These results are outputted to both the research data store, where they can be viewed by the admin, and to the user profile, where they are further processed into data visualizations to be viewed by the user. In this phase, the activity data store received data from the admin in the form of additional activities to recommend to users. The calendar data store receives dates and chosen activities that the user wishes to schedule. The calendar outputs notifications of the scheduled activities to the user. Survey results from the user profile are combined with activities and their associated domains to make suggestions of activities to schedule that are relevant to the user’s domain scores.

Components for the Peer Testing #1 and #2 milestones are blue and yellow respectively.

In the month between Peer Testing #2 and our Final Submission, we intend to add additional functionality to existing components, such as help pages or a tutorial. However, there are no components labelled for this milestone, because we do not plan to make major changes to the system architecture that late in the process.

# Functional Requirements by Milestone

## Peer Testing 1

1. Users answer the initial health questionnaire, preferably doing all 5 domains (social, emotional, mental, physical, and spiritual).
   1. Users can choose which domains to do the questionnaire for, and then can prioritize domains in order of importance
   2. The questionnaire has general questions to start, followed by each specific domain’s questions
   3. User answer questions to rank aspects of each domain (eg sleep for the physical domain) on a scale of 1-7
   4. Users can redo the initial health questionnaire as many times as they want to
   5. The user’s answers are saved locally
2. The first time a user opens the app each day, they are greeted with an uplifting message/quote or small action item (ex. “Have you smiled today?” or “Strike a power pose!”)
3. Users are prompted to pick domains to update (0 to all), while being shown how long it has been since last update for each
   1. Users answer a subset of the health questionnaire based on which domains they select
4. Users can view circle visualizations
   1. The visualizations consist of concentric circles representing each health domain
   2. The size of each circle correlates with their wellbeing in each domain
   3. The visualizations are static design elements at this phase, not generated from user data
5. Users can access a calendar within the app which shows the dates of scheduled activities and when domains were updated.
   1. They can view the calendar populated with test activity data, but cannot add, edit, or remove items
6. Users navigate the app with a bottom navigation menu
   1. They can tap buttons to jump directly to the calendar, questionnaires, stats/visualizations, and account/settings
   2. Standard back to previous screen button
   3. These pages can be viewed, but only the questionnaire page can be interacted with at this stage
7. Users can choose colour schemes for each health domain
   1. They can choose from a predefined colour schemes or create their own

## Peer Testing 2

1. A user can register for an account with email and password
   1. They then receive an email to confirm email address and registration
   2. Once their email is confirmed, a user sets up their profile with their name and whether they are a patient of Dr Dawson
   3. If the user selects the patient option, they are asked to agree to another information sharing agreement specific to patients
   4. A user can upload a profile pic or select an avatar
2. Users can answer the health questionnaire multiple times
   1. Each completed questionnaire is saved with a timestamp
   2. Scores are used to create the circle graph visualizations
3. Users can tap the circles to view recommended action items or choose their priority domains
4. Users can add action items to the calendar
   1. Action items are recommended based on which domains have low scores
   2. Users can move, add, change, or delete action items
   3. Users can set notification reminders for action items
   4. Users can switch the view of the calendar to a list of action items sorted by date or a standard month layout
   5. Users can change the list of action items to filter by domain instead of date
5. The user can set a time for daily notifications for a reminder to open the app and check in.
6. Users can change their settings
   1. They can change their data sharing and notification preferences
   2. They can erase all personal data from server/device
7. Users can recover their account if they have forgot their password
   1. The user types in their email
   2. If an account exists with the email, an email is sent with a temporary password
8. Dr Dawson is provided unique access to users’ shared information through a web app
   1. He can view visualizations of group data
   2. The web app provides list of his clients who are not updating their domains or who show (significantly) declining health domains
9. Dr Dawson can use the web app to add or update new content
   1. He can add new survey questions without affecting existing data
      1. Users will answer those questions when they next update the corresponding domain
   2. He can add new recommended activities and motivational messages

## Final Milestone

1. New users view introductory information when they first open the app
   1. They initially view a page about the goals of the app
   2. Next users can view a tutorial
      1. They can skip tutorial, but are warned of missed benefit
   3. Users are pointed to the menu, navigation, account registration, and the initial questionnaire
   4. Users must read and accept terms of service
2. Users can navigate to an online store to purchase prints of their visualization
3. Users can share to social media
   1. They can share visualizations from the domain visualization screen
   2. They can share their daily uplifting quote/message
   3. They share their success when they complete an activity

# Non-Functional Requirements

**Capacity**: supports at least 100 simultaneous users uploading data to the server without affecting its performance

**Interoperability**:

* adapts to both portrait and landscape mode
* supported on both Android and iOS

**Reliability**: content is maintained locally to accommodate sporadic server connection

**Security**: developers will not have access to unencrypted user data

**User Experience**: each domain questionnaire can be completed within 5 minutes

**Regulatory**:

* all user data is stored in Canada
* reasonable security arrangements are made to prevent unauthorized access, collection, use, disclosure, copying, modification, or disposal of data
* personal data is retained for at least one year after using it
* before collecting data, the purpose of the collection of data is disclosed
* on request of an individual, all personal information stored and the ways in which it has been used is provided

**Data**:

* Health data that is entered into the app is initially stored locally, with the option of storing in a database and giving access to the client, Dr Dawson, for psychology research purposes
* Personally identifying data is encrypted.
* Dr Dawson only has access to the names of users who confirm that they are his clients
* Questionnaire data is saved locally if app is exited mid-questionnaire
* Basic app usage metrics are saved and sent to database
* Updateable list of action items/activities specific to improving health in each domain
* Updateable list of uplifting messages and quotes

# Environmental Constraints

* modern Android or iOS smartphone
* internet connectivity to share questionnaire data
* the Personal Information Protection Act must be followed in British Columbia
* development must be finished by the Capstone course end in April

# Tech Stack

## Frontend

We believe React Native + TypeScript is the best choice for our tech stack. Web apps are arguably simpler to develop and test, however given the functional requirements for the app we believe the app is best suited for Mobile. Using Android Studio is not a good option for us since we do not have a way to test Android apps on a device. In addition, limiting the app to be Android only could limit its outreach.

Although both TypeScript and JavaScript are supported by React, we believe TypeScript is a better option as TypeScript has been proven reduce programming errors as the codebase grows. TypeScript is not a large jump from JavaScript since JavaScript is also valid TypeScript.

## Comparison of Options Considered

### Android Studio (Mobile)

Android Studio is the default way of building applications for Android. Apps are developed within the Android Studio IDE and compiled to Android packages.

**PROS:**

* Group is experienced with Android app development.
* Provides a true native experience on Android with a native look and feel.
* Better performance on Android than the alternatives.

**CONS:**

* Only supports one platform; porting to iOS would be cumbersome.
* Java is not as flexible as JavaScript/TypeScript which could slow down development.
* More difficult to design nice-looking interfaces compared to React, which allows for much more customization for UI components.
* None of our team members own an Android device for testing

### ReactJS (Web)

React is a frontend UI library for JavaScript developed by Facebook. React is developed using JavaScript or TypeScript + HTML and CSS.

**PROS:**

* Currently the most popular frontend web framework backed by a large community of users, which means lots of resources (libraries, tutorials, etc.) are available.
* Simplifies UI code by separating UI elements into components which can be reused.
* Availability of component libraries like Ant Design, which includes many pre-made components to speed up prototyping.

**CONS:**

* Team has less experience with React compared to Android code.
* JavaScript is arguably more prone to errors compared to Java since it doesn’t have any compile time checks (this is mostly solved by TypeScript).
* Web is not the ideal platform given our requirements.

### React Native (Mobile + Web)

React Native is mostly identical to ReactJS, but it replaces HTML+CSS with abstracted components which are implemented differently depending on whether the app is running on iOS, Android, or Web. This abstraction allows us to write the code once and deploy it anywhere.

React Native is developed using JavaScript or TypeScript and is compiled to iOS apps, Android packages, or the Web.

**PROS:**

* Largely identical API to ReactJS, meaning learning resources designed for ReactJS also apply to React Native.
* Supports multiple platforms (iOS, Android and Web).
* Allows for more customizability than true Native mobile frameworks.

**CONS:**

* On mobile, React apps won’t feel as native compared to Android Studio for Android and Swift/Objective C for iOS.
* Generally has more ‘bloat’ than a native application since it requires a JavaScript engine to be installed alongside the app.
* Team has no experience with React Native.

## Backend

We intend to use Express (NodeJS) as our backend with MongoDB as our database. Express provides a very flexible API while still being able to handle hundreds of thousands of requests per second. The main benefit of using Express is that it can be used with TypeScript which is what we are using for the frontend. For our database, we have chosen MongoDB as it has great support in Express, though this is subject to change as the project evolves.

Fiber provides incredible performance, but this level of performance is only beneficial for very large tech companies in products that are beyond the scope of our project. Flask provides a very clean API, but there is little reason to prefer it over Express, since Express provides an API that is just as flexible.

## Comparison of Options Considered

### Express (NodeJS) + MongoDB

Express is the standard web framework for NodeJS. It is widely used by large tech companies.

**PROS:**

* Very modern and easy to use API. Similar APIs have been added to other languages due to the popularity of Express.
* Scales very well with multiple NodeJS instances.
* Uses JavaScript/TypeScript which complements our frontend tech stack.
* MongoDB uses JSON for storing data, which is designed to be used with JavaScript.

**CONS:**

* Not as fast as web frameworks available for compiled languages (369,533 requests per second). This can be mitigated by using multiple NodeJS processes.

### Flask (Python) + MariaDB

Flask is a minimalistic Python micro web framework. It provides only the bare minimum for a web framework, everything else (such as database layers or form validation) must be done using third party libraries.

**PROS:**

* Minimalistic nature of Flask means the API is very well thought out and easy to pick up.
* Does not come with any extra ‘bloat’ which keeps the backend lightweight.

**CONS:**

* Very slow throughput (number of requests per second) compared to other web frameworks, though it is among the fastest Python frameworks (83,398 requests per second).
* Large Python codebases are difficult to maintain.

### Fiber (Go) + MariaDB

Fiber is a small web framework written using fasthttp in Go. It is designed to be similar to Express (NodeJS).

**PROS:**

* Flexible and very modern API.
* Based on fasthttp, which means Fiber is among the fastest web frameworks available (6,566,971 requests per second).

**CONS:**

* Go is not as easy to learn as Python or JavaScript.
* Does not complement our frontend tech stack as nicely as Express.

*Performance metrics from*[*techempower benchmarks*](https://www.techempower.com/benchmarks/#section=data-r19&hw=ph&test=composite&a=2)*.*

# Test Strategy

## Unit Tests

For testing system level features—that is, features not including visible UI components—we are using unit testing in combination with continuous integration. To achieve this, we are using the Mocha unit testing library. This testing framework was chosen as it is one of the most popular unit testing frameworks for our chosen tech stack (TypeScript and React). Another popular framework that was considered is Jest. However, Mocha is more customizable as it is a modular framework. Unit tests are designed to run in isolation of the rest of the application, so we will be using mocking to procedurally generate input data for the code that we are testing. Using mocks for the data ensures we can cover as many test cases as possible.

## UI Testing

For the interactive user interface aspect of our application which cannot be tested using unit testing, we will be using the Expo framework for React Native. Expo allows us to visualize in real time any changes made to the code on a mobile device. This means that we can quickly test new UI functionality on the actual target device. As our code base grows, we may also use Storyblocks, which is a way to test UI components in isolation. Once Storyblocks is set up, we can test specific components without running the rest of the application, allowing the UI to be tested in isolation.

## Continuous Integration

For regression testing, we are using GitHub actions to set up continuous integration to ensure new code does not break previous functionality. In our setup, whenever someone opens a pull request to add more functionality to our code base, a GitHub action will run all of our unit tests automatically before the pull request is allowed to be merged with our main branch.

# Questions

#### How will you ensure that tests are sufficient to show the code works as intended?

For our backend, we will write tests for each API route. This will ensure that all paths exposed to the user are tested. For the frontend, most of the code will likely consist of UI code, where testing the UI visually is more feasible.

#### Not a single member of the team had an Android device? Heck, I’m not even mad, thats amazing. Did you consider flutter?

#### Did you consider using Flutter for the project since it also works on iOS and Android? What are you doing to ensure that your app follows the BC Personal Information Protection Act?

We did consider using Flutter, however learning a completely new programming language (Dart) could slow down development. Unlike Flutter, React is also widely used in Web applications, which would make porting our app to the web almost trivial.

For the second question, our client has mentioned that they will be consulting a lawyer to ensure our app follows the constraints of BC’s Personal Information Protection Act. We will use test data—no real client/patient data while building the app—and it will be the client’s responsibility to ensure privacy laws are followed for his deployment.

#### I noticed you mentioned data being “saved locally” - are you planning on updating this at any point to save data externally, i.e. have the database be running on an external server before the project deadline? Or are you only focusing on local data storage?

#### When you say “data will be saved locally,” is this only at this stage with a database being set up on a server at a later date?

We will be doing both. We will save the data locally and only upload data that the user has given permission to share. Saving the data locally also ensures our app can be used offline; once online the app can synchronize with our server.

#### Since you’re using React Native, why not add support for more than just mobile devices? Does this add significant amounts of work (in your opinion, i.e. style and HCI components)? Did I just answer my own question?

Technically there’s no reason why we couldn’t support other devices aside from the fact that a mobile app does not always look great when it is just copy pasted into a desktop/web app, due to the things you have mentioned. Having components designed for specific screen sizes generally yields something that looks and feels nicer.

#### Will you be collecting and storing any data other than the data from the surveys that the users fill out? Or will the recommended activities be determined solely by the surveys?

We will likely record basic app usage data to give our client an idea of app engagement and usage by the users. The recommendations will be mainly determined by the results from the surveys. We are still discussing with our client more ways to improve the recommendations.

#### Could you explain how the algorithm will be formatted for how the exercises are decided?

#### How is the algorithm for interpreting survey results being developed?

#### Are the actions that are recommended based on survey results decided automatically or manually?

Recommendations for activities are generated based on the results from the user surveys. These surveys focus on Dr Dawson’s 5 key domains of health. Each survey question is tagged with the corresponding domain, so we can see which domain the user is struggling with and suggest relevant activities. For example, say a user enters a low score for the question “Do you exercise often?” We could the suggest exercises for the user to do.

#### How will you be testing on Android if none of the users have an Android device?

We will still test on Android devices closer to our milestone deadlines by borrowing these devices. However, React Native will look identical on both iOS and Android so we do not need to test on Android very often.

#### Do you anticipate any issues with using a multi-platform mobile framework (React Native)?

Not really. Of course, we expect some issues to arise in deployment, but native code would also have similar issues. ReactNative is used by mobile applications with large user bases (such as Discord) so it is a tried and tested technology.

#### Do people need to sign up to use the service if they are a general user?

Users will not be forced to register, but they will need to sign up if they want to be identified in the server. Say a user was to change mobile device at some point: they would need a login to identify themself and keep their previous data. If they are a patient of our client, they will need to register to consent to share information with him.

#### Can users choose to retake the questionnaire if they feel needed?

#### Will users be able to fill out the questionnaire multiple times depending on if their needs change so that the activities will be updated?

Yes they can! In fact, the app will be designed to remind users to update domains on a regular basis.

#### In your tech stack what backend will you be using to handle the data?

#### What kind of backend data storage will be used?

#### Does your application require any backend systems or database connections? What frameworks and technologies are you using for this area?

We will be using Express (NodeJS) as our backend web framework. For our database, we’ve decided for now to use MongoDB, although we are also considering a SQL database like MariaDB. We will finalize the database decision after the Peer Testing 1 deadline, as that is when we will start working on the server. By then, it should be clear how the frontend should access the data, making the database choice more obvious.

#### Why did you choose mocha over Jest?

Both frameworks are very similar. Jest is probably more popular right now since it is backed by Facebook. Mocha is nice because it relies on other libraries for assertions, so you are not limited to Jest’s ‘behaviour’ style assertions (like expect(value).toBe(0) versus assert.eq(0, value)). The differences are very minor though; both are very good test frameworks.

#### How will you go from survey answers to health circles. has this system already been created?

This is still being actively discussed with our client. He has provided a basic scoring system where the radius of the circle will be relative to the score in the domain, so a high scored domain has a large circle.

#### What sort of useful feedback will the app give the user?

The app suggests activities that the user can do to improve in key aspects of their health, as well as visualizations of their current well-being in those aspects.

#### How does your app UI fit in different screen sizes?

ReactNative components use styles very similar to CSS on the web, which means features designed to make pages responsive such as Flex Box are supported. We will only support mobile devices, which limits the range of screen sizes we need to support.

#### How do you plan on debugging the application if the developers don’t have access to the raw user data?

We will use mock data during development and for debugging. We will not need access to real user data when debugging the app.

#### How are you going to test 100+ simultaneous users using your app?

Since we are using NodeJS for our backend, we can scale up easily depending on traffic by launching more instances of the server.

#### When you say modern iOS and Android, which devices and versions will be considered as modern?

#### Do the code for both Android and IOS stay the same? Is there version requirements for the Android and IOS phones?

Android: 6.0 (Marshmallow) or newer. iOS: 10.0 or newer. The code for either stays the same unless we want to implement some component that uses a functionality only available on a certain device.

#### What kind of information will be shared between clients/patients and their health care professional? Will it be entirely anonymized data, or will a client be able to share answers directly with their psychologist?

#### Is the data shared between client and doctor be anonymous?

Dr Dawson’s actual patients will be able to agree to share full info with him. Anybody else will be anonymized. The anonymous data is used in aggregate in Dr Dawson’s research.

#### What is the higher level of access that patients would receive over general users?

Patients have the option to share their data directly with Dr Dawson with their name attached.

#### What criteria will be used to develop the questionnaire(how are questions formulated to ensure their proper functionality and purpose)?

The questionnaires are provided by Dr Dawson as he is the domain expert, then reworded by our team to ensure brevity and consistency.

#### How do these “circles” function?

This is a work in progress for us but right now we show each colour coded domain as a circle with a common centre point. The radius of each concentric circle is relative to the score achieved in that domain.