Software Requirements Specification

Consolidated Health App

Cody Price
Rafal Ryczek
Aaron Jordan
Quinn Poyneer

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to specify and outline the agreed upon requirements for the consolidated health app “HealthOut”. This document gives an overall description of “HealthOut” including its functionalities, constraints, and dependencies. The intended readers of this document are the Group 5 development team and the inquiring customer of “HealthOut”. The goal is for the inquiring customer to either approve of or make changes to the requirements specified in this document. Once Approval has been guaranteed by the inquiring customer, the Group 5 development team will continually reference the agreed upon document during development of “HealthOut”. [1]

1.2 Scope

“HealthOut” is a mobile application that allows the user to consolidate their health data from multiple 3rd party health apps (provided the 3rd party health apps have an available API that we have been given access to). The user can then set health-related goals, view their progress, and displays each goal graphically. [2]

To elaborate, the user will be provided with a list of all 3rd party health apps that we have access to through their available API. After a user selects which 3rd party health apps they’d like to consolidate from that list, we will receive their health data from those 3rd party health app APIs and consolidate their health data. After the user sets health goals (i.e. steps, miles walked, blood pressure, etc.), “HealthOut” will give the user a steady update on the user’s progress toward their desired health goals; educate the user on the standards for general physical health; and minimize the need for multitasking between different applications.

1.3 Definitions

HealthOut	Name of the consolidated health app project
User	Someone that interacts with “HealthOut”
Database	Program used for storing, finding, and changing information
API	Code that allows two software programs

	to communicate with each other
Health app	An app that collects the users health data.
Health data	Facts and statistics, pertaining to health (i.e. steps walked, miles walked, calories burned, etc.), collected together for reference or analysis.
Software Developer	A person, group, or company that creates software applications.
3rd party developer	Software developers who do not belong to the Group 5 software development team
3rd party app	Programs developed by a 3rd party developer
Android Studio	The official integrated development environment for Google's Android operating system.
Emulator	Hardware or software that enables one computer system to behave like another computer system.
Bug	An error, flaw, failure or fault in a computer program or system that causes it to produce an incorrect or unexpected result, or to behave in unintended ways.
Regression testing	The process of testing changes to computer programs to make sure that the older programming still works with the new changes.
Refactor / Refactoring	The process of restructuring existing computer code, not to change its function, but instead to make it run more efficiently and to be easier to understand/edit.
JUnit test	A method contained in a class which is only used for testing.

1.4 References

[1] IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications

[2] IEEE Std 1233, 1998 Edition, IEEE Guide for Developing System Requirements Specifications.8

[3] CSE.Chalmers 2010, Software Requirements Specification Amazing Lunch Indicator http://www.cse.chalmers.se/~feldt/courses/reqeng/examples/srs_example_2010_group2.pdf

[4] UCSS.edu, CS330 Software Engineering Software Requirements Specification (SRS) Template https://www.uccs.edu/Documents/tboult/srs.doc

[5] Jahoda, Philipp, and CommonsWare. "What Is the Maximum Amount of RAM an App Can Use?" *Stack Overflow*, 7 Sept. 2013, 16:40, www.stackoverflow.com/questions/18675557/what-is-the-maximum-amount-of-ram-an-app-can-use

1.5 Overview

The remainder of this document will contain six chapters. The second chapter will provide a detailed description on the functionality of the project and an overview on how the user will interact with "HealthOut" and how data will be transferred to and from external 3rd party applications. This chapter will also discuss the constraints, assumptions, and dependencies concerning the project. [2]

The third chapter will provide the specific requirements of the project through the use cases, context diagram, and activity diagram. These figures will be supplemented with detailed descriptions on how and when each part of the mobile app is linked to each other.

The fourth chapter will provide a series of user stories which will validate the context diagram and outline what "HealthOut" should accomplish and how the software can be tested.

The fifth chapter will provide details on the non-functional requirements of the project including the factors which will influence software maintainability, and testing methods thereof.

The sixth chapter will provide the responsibilities for each member of the project team. Each member will be assigned to work on a specific area “HealthOut” and to develop and collaborate with the other members.

The seventh chapter will provide a log of the meetings held by the project team and what was accomplished in those meetings.

2. Overall Description

2.1 Product Perspective

"HealthOut" operates in conjunction with multiple 3rd party health apps. Users will be able to register an account and login through the User Management system that stores the users account information in the Users Database. The user can then select from a list of 3rd party health apps (that we have access to through their API) for which they would like "HealthOut" to retrieve their health data from and consolidation. This will be done through the App Registration Management system which will then ask the user for their login information for all selected 3rd party health apps. The login information for all selected 3rd party health apps will then be sent to the 3rd Party Health App API Management system, which will send it to each 3rd party apps API. Each API will either confirm or deny access. If access is denied, then the 3rd Party Health App API Management system will sends a message to the App Registration Management system to prompt the user to try again. If access is confirm, then the API will also send the 3rd Party Health App API Management system the users log data for that 3rd party health app. The 3rd Party Health App API Management system will then store the users login information for that 3rd party health app in the 3rd Party User Database for easy and quick reuse for future interactions with that 3rd party health apps API. The 3rd Party Health App API Management system will then store the users log data for that 3rd party health app in the App Logs Database. The Health Goals Management system will then pull the users log data from the App Logs Database to allow the user to set health goals. Once health goals have been set, the Health Goals Management system will store those goals in the Goals Database. The Health Goals Management system will from then on pull the users goals from the Goals Database and the users log data from the App Logs Database to display the users goals. When a user then clicks a specific goal, the users log data and that specific goal will be sent from the Health Goals Management system to the Graphing Management system to be displayed graphically [4]

"HealthOut" will be optimized for Android devices that run Android OS version 5 and later. REST API protocol will be used to communicate between the application and databases[6]. Operations of the app include goal setting, general health guideline information, account/profile backup via login authentication, and accurate synced data representation. To ensure optimal performance of the application, we will account to use a maximum of 32GB of memory. [5]

2.2 Product Functions

"HealthOut" is an app that consolidates the health data from multiple 3rd party

health apps into one place for convenience. The user will select which 3rd party health apps they wish for “HealthOut” to gather health data from. The user will be able to set health goal(s) based off the health data collected from those 3rd party health apps, which will then be compared graphically so the user can visualize their progress. The user may adjust which 3rd party health apps “HealthOut” includes as well as their goal(s) at any time. [3]

For example, the user selects a 3rd party health app that track calories consumed in a day. He/she then sets calories consume in a day as one of their health goal(s). After they have logged their calories eaten in a day into the selected 3rd party health apps, “HealthOut” will use that data to produce a bar graph comparing how many days the goal was reached.

2.3 User Characteristics

There is only one type of user that will interact with “HealthOut”; the user who uses the app’s functions. [3]

The users of the mobile app are only able set goals and view their progress through graphs. The users will sign up and/or login, register external health apps to use their data, and input the goal(s) to be graphed.

2.4 Constraints

The mobile application is mainly constrained by the external health apps and their APIs. Most health-related apps are not open source and don’t provide access to its log files; “HealthOut” will need to check if retrieving data from a 3rd party app is allowed in order to successfully register it. [2]

Another design constraint would be the size of the databases. Since user log data is being pulled from 3rd party apps, the databases will need to accommodate an array of long strings of data per each health app.

2.5 Assumptions and Dependencies

It is assumed the application will only run on mobile Android devices that have good performance and enough storage space. The user will need to have a smartphone running Android version 5.0 or above. Product performance might vary depending on the user’s choice of Android device and it’s OS version. [3]

The application will depend on installed 3rd party health apps having available APIs to pull log data from, and then the user to have an account with at least one of those 3rd party health apps..

2.6 Apportioning of Requirements

Requirements that may be delayed for future versions include using Android notifications to inform users of their progress toward their goal. Another low

priority requirement includes giving the user customization options for the app to change the colors and themes. If no delays are present, these features will be implemented in the project. [1]

3. Specific Requirements

3.1 Use Cases

Use Case: Register Account
Primary actor: User
Goal in context: To create an account for HealthOut
Precondition: The user has already downloaded the app, HealthOut
Trigger: The user opens HealthOut, and selects register on the Login page
Scenario:

1. User: enters their email
2. User: enters their desired password
3. User: selects enter
4. User: receives an email to activate their account
5. User: activates their account by following the link in activation email

Exceptions:

1. User is already registered to the app
2. Email is invalid

Priority: Essential
When available: First Increment
Frequency of use: Most likely only once
Channel to actor: Register page
Secondary actor: None
Channels to secondary actor: None
Open issues:

1. Should the User Database store the wrong email or password
2. Should the user never receive an activation email

Use Case: Login
Primary actor: User
Goal in context: To login to HealthOut
Precondition: The user has already registered an account for HealthOut
Trigger: The user opens HealthOut
Scenario:

1. User: enters their email
2. User: enters their password
3. User: selects enter
4. User: logs into their HealthOut Account

Exceptions:

1. User is not registered to the app
2. Email is invalid
3. Incorrect password is entered

Priority: Essential
When available: First Increment
Frequency of use: Multiple times
Channel to actor: Login page
Secondary actor: None
Channels to secondary actor: None
Open issues:

1. Should the App not give access to a correct email and password
2. Should the App give access to an incorrect email and password

---

Use Case: Register App
Primary actor: User
Goal in context: To register a health app to HealthOut
Precondition: The user must be logged in and have health apps installed
Trigger: The user selects to register app
Scenario:

1. User: selects 3rd party app installed on device
2. User: enters login for the app
3. App: contacts 3rd party app API
4. 3rd party app API: sends app log data to HealthOut database

Exceptions:

1. User has no health apps installed
2. 3rd party app API unavailable

Priority: Essential
When available: Second Increment
Frequency of use: Many times
Channel to actor: Register app page
Secondary actor: 3rd party app API
Channels to secondary actor: 3rd party Health App API Management
Open issues:

1. Health app to register has unavailable API to use: issue error to user

---

Use Case: View Goal
Primary actor: User
Goal in context: Allow user to see their health goal
Precondition: The user must be logged in and have health apps registered
Trigger: The user selects a health goal to view
Scenario:

1. User: selects health goal
2. App: displays the goal and log data graphically

Exceptions:

1. User hasn't input data to 3rd party registered app: no log data to be graphed

2. No goals have been made

Priority: Essential
When available: Second Increment
Frequency of use: Multiple times
Channel to actor: Health goals page (main page)
Secondary actor: None
Channels to secondary actor: None
Open issues:

1. Duplicate goals: prevent same goals from being made

Use Case: Edit Goal
Primary actor: User
Goal in context: To remove, change, or add any goals of the user
Precondition: The user must be logged in, and either already have a goal created, or wish to create one
Trigger: The user selects edit goal
Scenario:

1. User:
    a. selects a goal to be edited (change or remove)
    b. creates a new goal (add)
2. User:
    a. enters intended changes to the selected goal (change or add)
    b. deletes goal data (remove)
3. User: saves changes by clicking “Done”
4. App:
    a. updates selected goal and View Goal display (change or remove)
    b. creates and displays new goal in View Goals display (add)

Exceptions:

1. There must be log data from a 3rd party app

Priority: Essential
When available: Second Increment
Frequency of use: Multiple times
Channel to actor: Health goals page (main page)
Secondary actor: None
Channels to secondary actor: None
Open issues:

1. Should there be a limit to how many goals can be created on one account

---

Use Case: Edit Account Information
Primary actor: User
Goal in context: Allow user to change their email or password
Precondition: The user must be registered and logged in
Trigger: The user selects to edit account

Scenario:

1. User:
    a. enters new email (change email)
    b. enters new password (change password)
2. App: sends updated credentials to Users database
3. App: sends confirmation to email address*

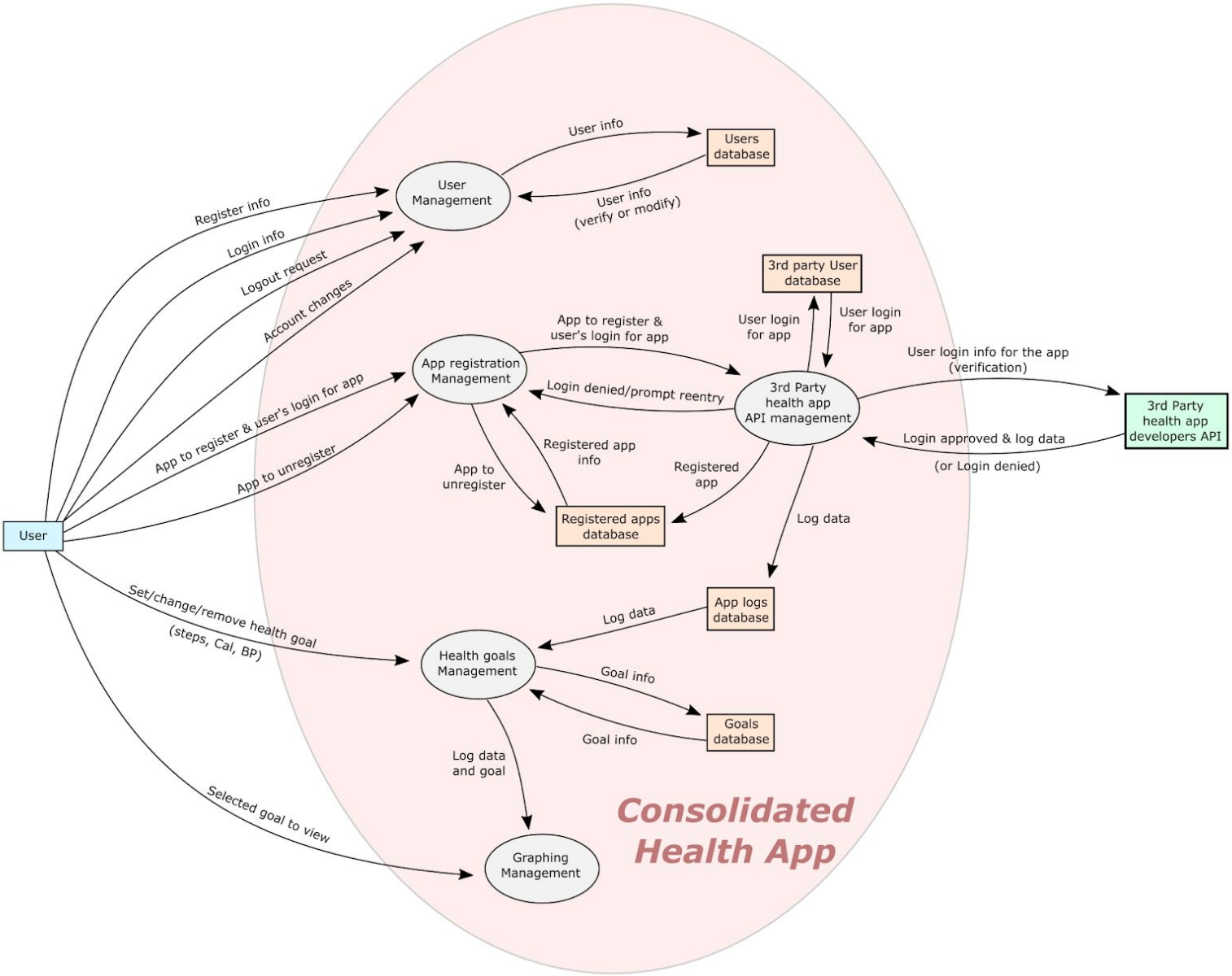
Exceptions:

1. Password does not fit within listed requirements (i.e. too short)

Priority: Secondary
When available: Final Increment
Frequency of use: Some times
Channel to actor: Menu sidebar
Secondary actor: None
Channels to secondary actor: None
Open issues:

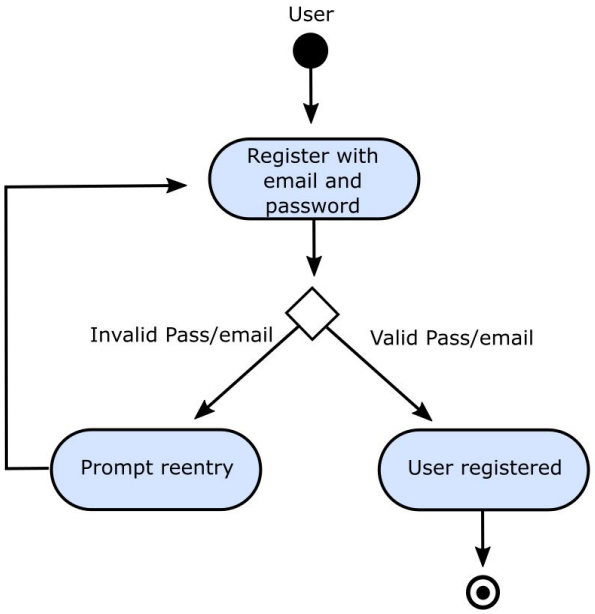
1. User fails to confirm new information

3.2 Context Diagram

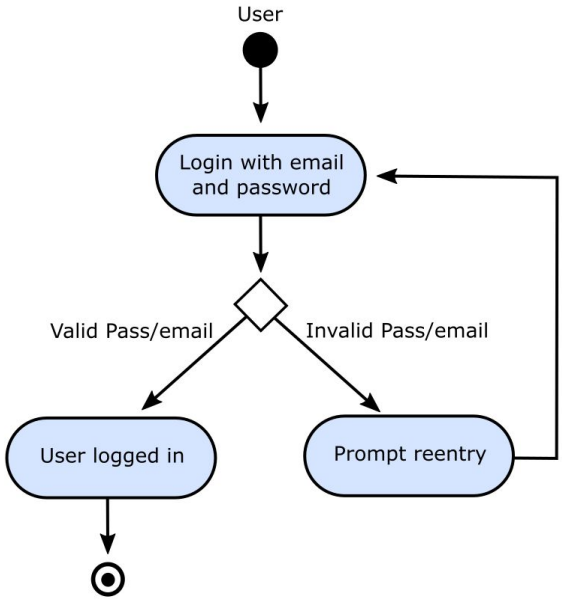


3.3 Activity Diagrams

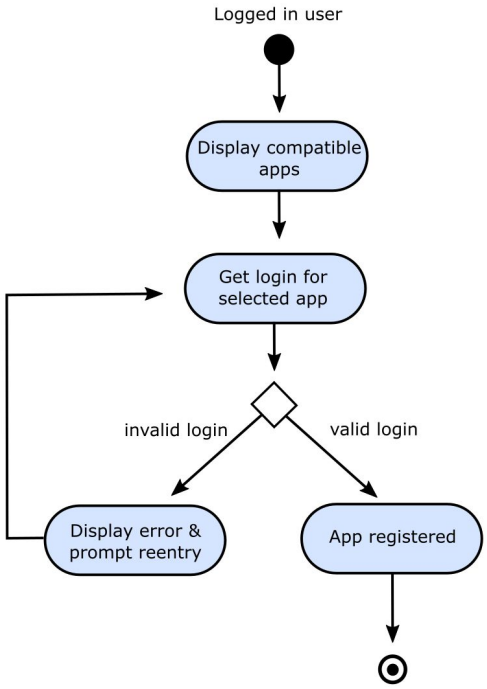
Register Account



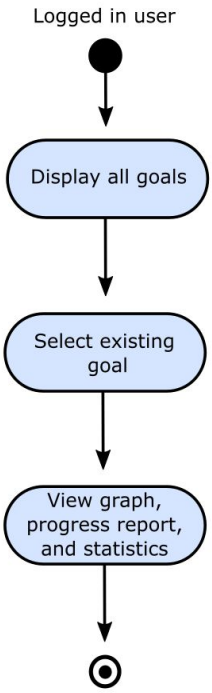
Login



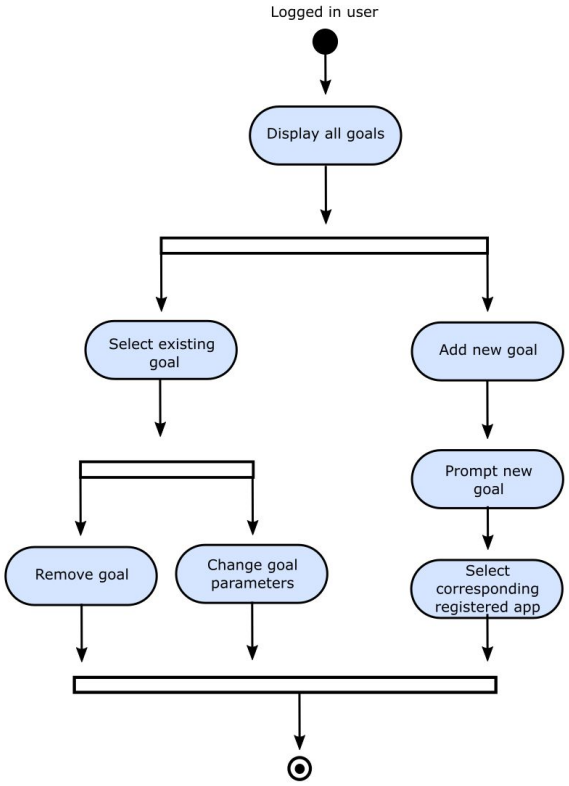
Register App



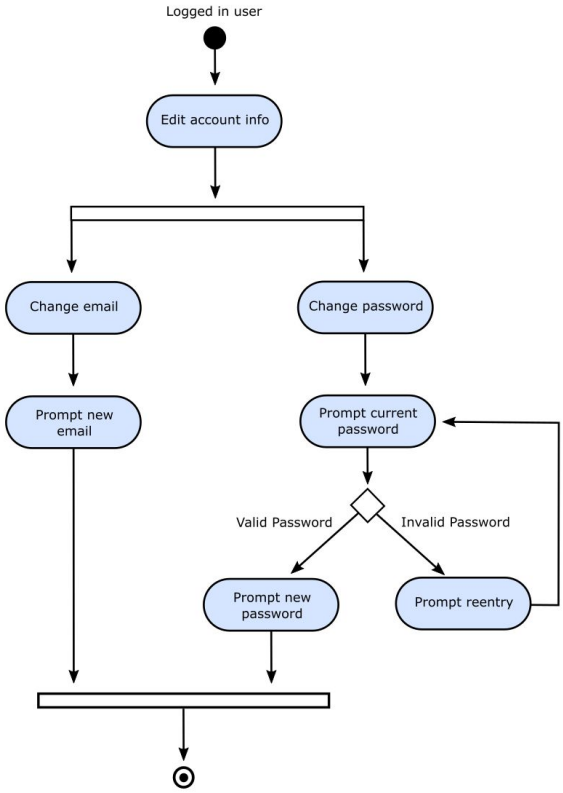
View Goal



Edit Goal



Edit Account Information



3.4 User Interfaces

Concepts of the user interfaces for the different app pages:

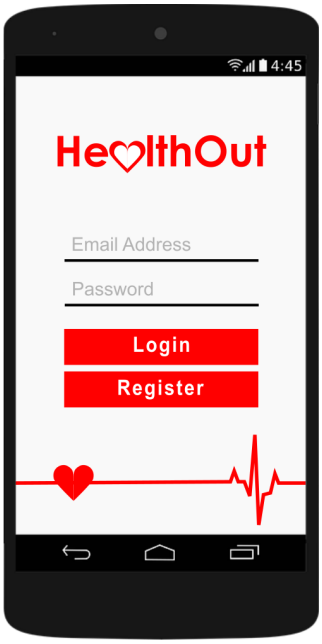


Figure 1 - Login page

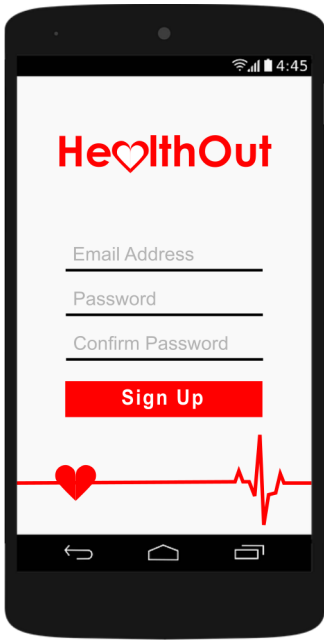


Figure 2 - Register page

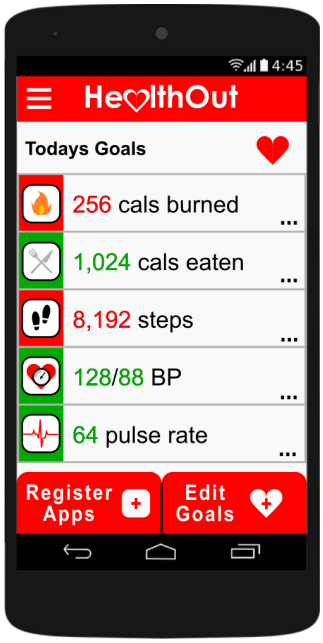


Figure 3 - View goals page(1)

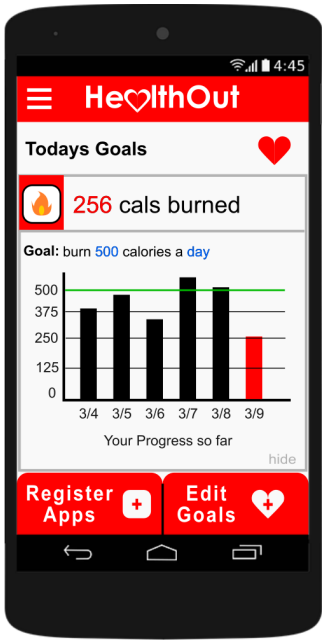


Figure 4 - View goals page(2)

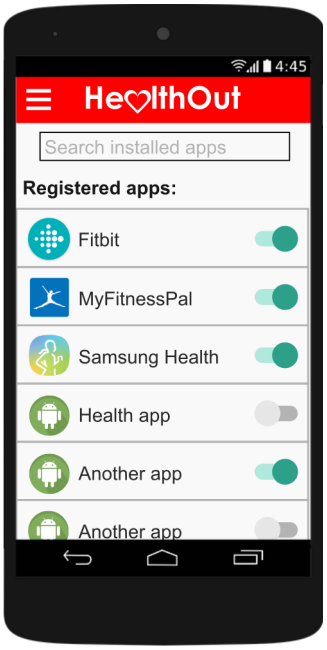


Figure 5 - Register app page

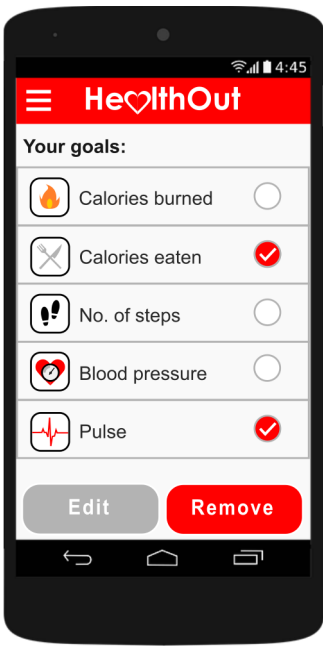


Figure 5 - Edit goals page

4. User Stories

As a user, I want to register for the consolidated app so that I have an account to access the app.

As a user, I want to login to the consolidated app so that I can have access to its features.

As a user, I want to log out of the consolidated app so that my information is secure.

As a user, I want to register a health app to the consolidated app so I can use that app's data for my goal.

As a user, I want to unregister a health app from the consolidated app so its data is not used for my goal.

As a user, I want to set a daily/weekly/monthly/yearly goal for the number of steps per day so that I can fulfill my goal.

As a user, I want to set a daily/weekly/monthly/yearly goal for the number of miles walked per day so that I can fulfill my goal.

As a user, I want to set a daily/weekly/monthly/yearly goal for the number of calories consumed so that I can fulfill my goal.

As a user, I want to set a daily/weekly/monthly/yearly goal for the number of calories burned so that I can fulfill my goal.

As a user, I want to set a daily/weekly/monthly/yearly goal for my blood pressure so that I can see if I meet my goal.

As a user, I want to set a daily/weekly/monthly/yearly goal for my pulse so that I can see if I meet my goal.

As a user, I want to be able to change my goals at any point so it better fits my needs.

As a user, I want to be able to remove a goal at any point if I no longer interested in that goal.

As a user, I want to view my progress towards a goal, graphically, so that I know if it's fulfilled.

5. Non-functional Requirement

5.1 Definition of Maintainability

Maintainability refers to the process of correcting and refactoring code, implementing supplementary features to the app, and improving the storage and CPU performance. Some factors considered for maintainability:

1. We will rework design aspects of the app to make them more simplistic and intuitive for the user. The product should be simplistic and intuitive enough that any user 16 years or older can learn to navigate the app in under 2 hours of use.

2. Create new databases with larger size if old ones begin exceeding size limit; transfer data from old databases to new and test for compatibility. The databases "Healthout" uses should be large enough to fit the data of all users of the app.

3. Code will be refactored to make it run more efficiently, as well as be easier to understand for any programmers that may need to edit it in the future. All code should be written in a clear enough manner that all new team members can understand the code without needing to contact the developer who wrote it.

4. "HealthOut" will be written such that additional open-source 3rd party APIs may be made compatible as development continues.

5.2 Testing Methods

Acceptable range for each factor of maintainability is defined as followed:

1. Whenever a change is made to "HealthOut", the group 5 development team will use regression testing, by going through each system and through each path within the activity diagram to detect if any new bugs or vulnerabilities have arised. If a single bug or vulnerability is detected, then further production on the project will not continue until that bugs or vulnerability is fixed. "HealthOut" should get the correct user information from registered 3rd party health apps. When a user clicks a textbox, a keyboard should pop up so they can type. "HealthOut" should correctly read what the user types into a textbox. When a user clicks a button, that button should begin the task that is labelled on it. When a task (that does not rely on a 3rd party app API) begins, that task should complete in no

longer than a minute. “HealthOut” should not crash more than once a month. The user’s login information for their “HealthOut” account should not accessible to anyone other than the group 5 development team. The user’s login information for 3rd Party Apps that they register should only be accessible to the group 5 development team and the corresponding 3rd Party App API.

2. In early stages of development, “HealthOut” will be tested via an emulator within the Android Studio suite. After a prototype of “HealthOut” has been developed, the app will also be tested on real Android Devices.

3. Once a prototype of “HealthOut” is developed, testing will not only be done by the group 5 development team, but also by select members of the general public (ages 16 or older) to ensure that they can learn to navigate the app in under 2 hours of use.

4. All databases will be tested to make sure they can handle the projected size of data we plan to store on them. The database “HealthOut” uses should be large enough to fit the data of all users of the app. The group 5 development team will test all databases using a JUnit test that runs on a real Android devices to make sure the databases do not crash.

5. Members of the group 5 development team will all review and refactor each other’s code to ensure that it is easy to understand and edit.

6. Group member responsibilities

Aaron Jordan
In charge of creating the **main menu** where the user will have their health goals displayed, as well as be able to navigate to the other pages of “HealthOut”.

Quinn Poyneer
In charge creating all five **databases** that “Healthout” will depend on, being the Users database, 3rd Party User Database, Registered Apps Database, and App Logs Database, and the Goals Database.

Cody Price
In charge of creating the **App Registration Management** system so the user can register which 3rd party health apps they’d like to consolidate on “HealthOut”; also in charge of developing the **3rd Party Health App API Management** system that gains access to the APIs of registered 3rd party health apps to pull the users log data.

Rafal Ryczek
In charge of creating the **Health Goals Management** system so the user can add/edit their health goals; also creating the **Graphing Management** system so the user can view a graph of their progress for a specific goal.

Stellan Sullivan
In charge of creating the **User Management** system so the user can register/ login to “Healthout”, as well as be able to change their account information.

7. Meeting minute notes

Meeting 1 (02/07/19):
User stories were developed for the consolidated health app.

Meeting 2 (02/15/19):
Context diagram version 1 was created; cover page, table of contents, and introduction of the SRS was made.

Meeting 3 (02/22/19):
Introduction of SRS was tweaked to improve readability; Overall description and its chapters was made; context diagram was revised multiple times; updated list of references.

Meeting 4 (03/01/19):
Use case and Activity diagram was created, added both diagrams to SRS; use cases, concept art for app UI, and non-functional requirements was made for the SRS. Introduction and Overall description of SRS was tweaked.