Platform for Collecting IoT Data

Final Project Report

(Tridens Technology, 2019)

Team Members:

Leo Li

Jake Tang

Jeffery Zhang

Kelvin Gao

Arvin Wang

Oscar Lekovic

Tutor:

Abdallah Lakhdari

Tutorial:

T15A

Client:

Basem Suleiman

29.11.2020

T15A_Group5

Executive Summary

Capstone Project 36: Collection of IoT data is a project which involves the design and development of an integrated mobile and web application system. The aim is to collect “Internet of Things” (IoT) data through the mobile application and to have this data available to researchers through the web application. Users of the mobile application will be able to set permission regarding the collection of data from specific sensors. Alternatively, users of the web application named ‘researchers’ query the collected data for further research and analysis. The application is for Android devices and uses the inbuilt sensors as a means for data collection. The overall outcome of this hybrid application is the creation of a highly usable platform which enhances the ability to explore IoT data more conveniently.

Group Members:

- Linxiao Li (Leo)
- Oscar Lekovic
- Tianye Tang (Jake)
- Qiangsheng Gao (Kelvin)
- Chenyue Wang (Arvin)
- Minjie Zhang (Jeffrey)

Table of Contents

Introduction

With the prevalence of interconnected devices and the abundant amount of data being gathered in today's society, there is an opportunity to study the enormous amount of data being sensed every second. "Internet of Things" (IoT) envisions a world of connected "things" that continuously sense data. Since this data can provide invaluable insights and the possibility for better solutions to problems faced in today's society, it is the goal of this project to allow the exploration of the collected IoT data. This report details the development of a hybrid mobile application which serves the purpose of collecting data and making it available for research via a web application.

There are several stakeholders that will benefit from the implementation of this hybrid application. Firstly, Dr. Basem Suleiman is the client in this project. He interacts with the project team through weekly meetings and email. He is responsible for stating the details of the project request, which includes goals, objectives, specifications and outcomes. The next group of stakeholders are the researchers/institutions who are looking to query the data. They use the web application to query already collected data, and can also publish their own research studies to collect specific data. Next, is the data providers. They are incentivised by a rewards program and are notified of rewards opportunities from newly posted research on the web application. Finally, there are third party businesses and researchers who may benefit from the research of specific data. These are the people that would pay money for the insights into the data and hence a crucial stakeholder. This report is broken up into several key sections:

System Specifications and Design: This section will start by giving an overview of the system based on user stories. It will outline the core functionality of the system. This section also details any technical (or other) constraints where applicable. This section will aim to demonstrate how the user stories provide the coverage requested by the client, and how they fulfill the requirements of the project. This section details how different components of the system interact. It talks about the system's architecture and how the system was designed and implemented. This section includes some high level diagrams and any other specifications or design elements specific to project nature.

Evaluation (Quality of Work): The quality of work section outlines the plans used for testing the integrated system. This includes the acceptance tests and criteria and also the design of test cases. This section also includes constraints and design patterns of the project. This section will also outline the use and application of disciple knowledge. This includes the

consideration of quality constraints for software design. This section also includes the tools used throughout the project.

Quality of Group Processes: This section aims to outline the processes of the group project, and how they compliment the team. It starts by detailing how the team manages and allocates tasks. It highlights the group roles from an XP perspective and talks about the tools such as Bitbucket and Slack which help with these roles. It then talks about the Bitbucket pipeline. Next, the section outlines the project specific deliverables and the processes surrounding each section's development.

Technical Documentation: The technical documentation section houses the evidence of project work. It documents tools used and techniques employed throughout the project.

System Specification and Design:

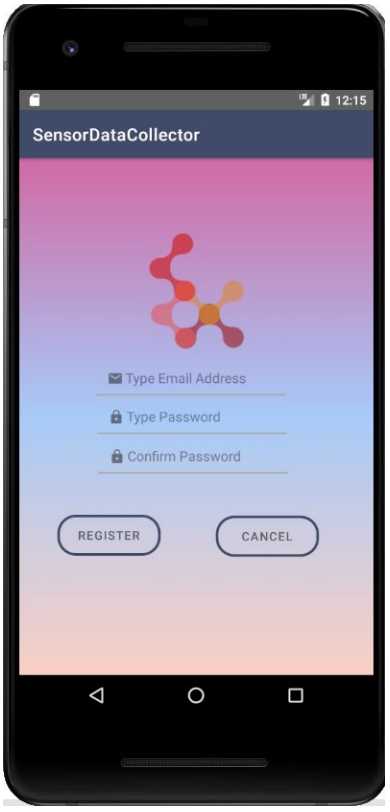
List of completed core requirements are expressed as user stories:

- As a provider, I want to provide as little personal information as possible when registering for the application so that I am certain that my personal information is not misused.

Implementation status: UI for the register page is implemented, only requiring email and password as mandatory fields.

Issues created on bitbucket:

1. Register Page UI and Interaction
2. Register page more immediate validation after user entered info

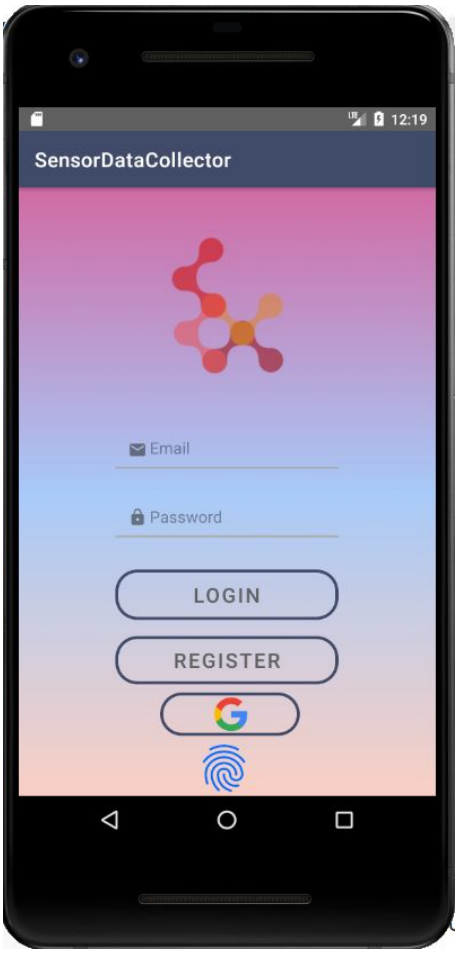


- In order to ensure data security, users want to be able to log in securely using a username and password.

Implementation status: Both the mobile and web passwords have been encrypted, and then transmitted to the back-end MongoDB and Firebase for data verification and test.

Issues created on bitbucket:

1. Web Login page UI design
2. Login database interaction

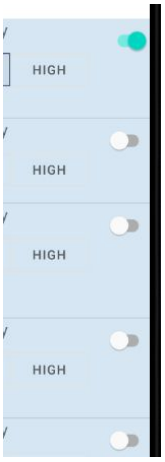


- As a provider, I want to control what sort of data I will be contributing so that I can avoid exposing my privacy and sensitive information.

Implementation status: A setting page is implemented which lists the sensors in the phone with a on/off switch beside it. Users can alter whether to let the app collect certain sensor information by pressing the on/off switch. Also, when user logout, all sensors will stop collecting data.

Issues created on bitbucket:

1. Mobile App: Get Data collection service updates in response to change in setting page

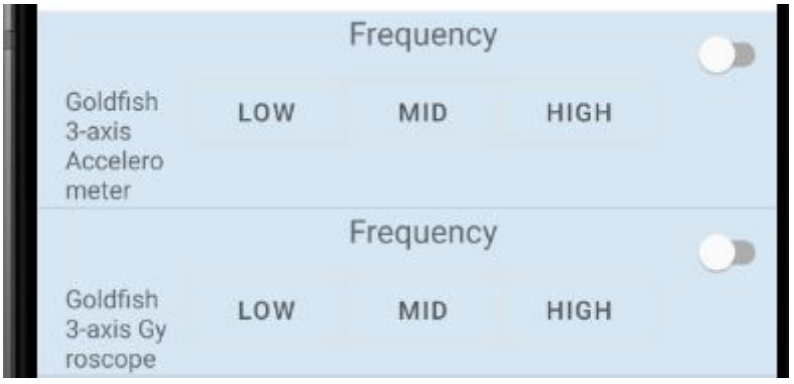


- As a provider, I want to be able to set how often that sensor data is collected so that I can restrict the amount of battery usage of the app.

Implementation status: For the list of sensors in the setting page, we added a select bar beside each sensor. Users can manipulate the frequency of collection by controlling the select bar which has low, medium and high options. Moreover, when the device battery is lower than 15 percent, they cannot upload data.

Issues created:

1. Mobile App: Change Sensor Configuration seek bar to L-M-H

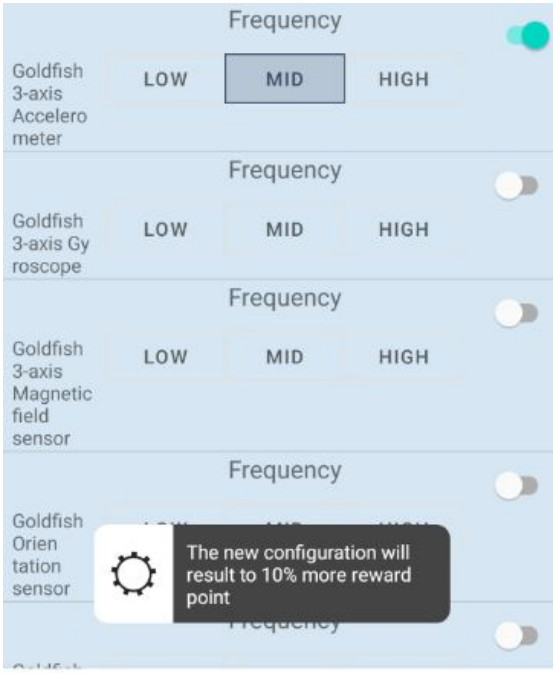


- As a provider, I want to have some incentives or rewards for contributing my data so that I find sacrificing my phone battery for data contributing worthwhile.

Implementation status: Developed a reward point block in the Mobile Dashboard page such that if the user is contributing data over a certain frequency threshold, they will be given some reward points which will be shown in this block. For different frequencies of sensors, we give different reward points.

Issues created on bitbucket:

1. Mobile App: Reward point for each sensor Mark from where

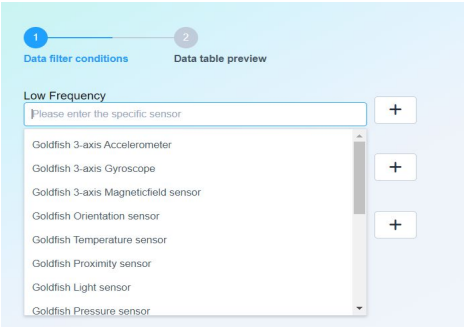


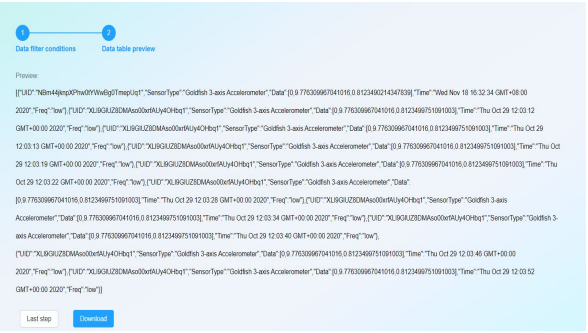
- As a researcher, I want the web application to have a service which allows me to search or filter based on data types so that I can avoid downloading data that is irrelevant to my project or research.

Implementation status: Designed a type of drop-down search box that only needs the keyword, which means that the researcher only needs to enter the keyword of the sensor, it will pop up the sensor name that he/she needs. In the data filter step2, we will give a preview of the first 10 data which means researchers can double check the data.

Issues created on bitbucket:

1. Web Improve data filtering conditions
2. Web: Get filter conditions and fix web page bugs



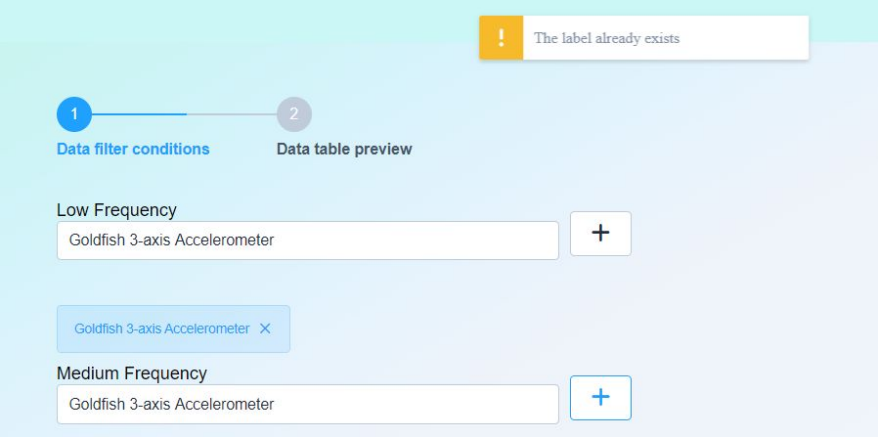


- As a researcher, I want the drop-down search box to be more convenient for me to select different frequency sensors, and when we choose a specific one in low frequency, this one can not be selected in other two frequencies.

Implementation status: Designed a function next to the drop-down search box which can allow researchers to add different sensors in the same frequency group at the same time. It will also mention researchers that they should choose at least one sensor in frequency groups or can not choose the same sensor in twice.

Issues created on bitbucket:

1. Web Improve data filtering conditions
2. Web: Get filter conditions and fix web page bugs

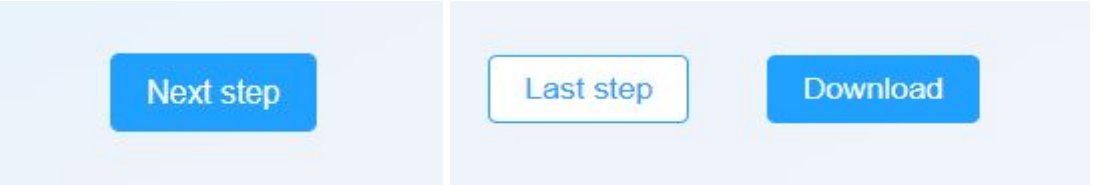


- As a researcher, I want the process of finding and downloading data to be intuitive and easy to follow so that I don't have to spend too much time learning how to use the service.

Implementation status: Designed the workflow of the web app to be simple such that the process of data retrieving from the site is broken down into steps. Users can complete and follow the process by pressing the "Next Step" button then wait a few seconds and there will be 10 data, "Download" button which will download data as a CSV file. "Last Step" button will take researchers back to the last step.

Issues created on bitbucket::

1. Web outline UI design
2. Web setting outline design

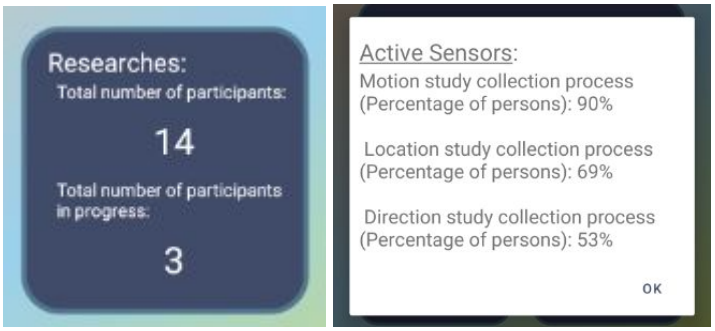


- As a provider, I want to have a summary of what data and how much data has been collected from me so that I can easily understand how much am I contributing and make a decision on whether to change the frequency of data collection.

Implementation status: Implemented a Dashboard page which will display the data collection summary such as number of records collected for each sensor. The block of Research shows the total amount of research and the number of the current processing research. When users click this block, the application will show them a clearer and more readable version as a pop-up window.

Issues created on bitbucket:

1. Dashboard click researcher to show the processing studies, modify the renew button and upload block UI

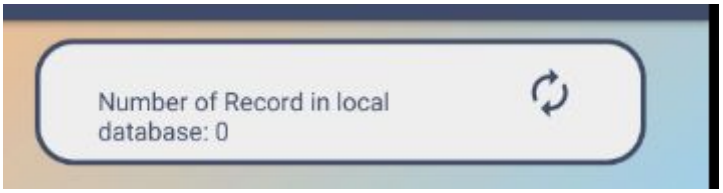


- As a provider, I want the app to have a function that allows me to upload my data manually, so I can control how to use my data.

Implementation status: Created the uploading functionality in the dashboard page. On top of the Mobile Dashboard page, there is a button called upload. This button can manually upload data and is offered for users to control when to upload their data.

Issues created on bitbucket:

1. Improve the Upload function and fix UI



- As a researcher, I want to download the data provided in a commonly used format such as csv or excel so that I can use them immediately for my project without additional preprocessing work.

Implementation status: In the Web Data Filter page, we displayed available data as csv dataset and also a “Download” button at the end of the 2nd step in this page, which will allow researchers to download the file after clicking this button. We add the line of the CSV file including UID, sensor information and time.

Issues created on bitbucket:

1. Generate CSV file

	A	B	C	D	E
	UID	SensorType	Data	Time	Freq
2	NBm44jknpXPhw0tYWwBg0TmepUq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123490214347839	Wed Nov 18 16:32:22 GMT+08:00 2020	low
3	NBm44jknpXPhw0tYWwBg0TmepUq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123490214347839	Wed Nov 18 16:32:28 GMT+08:00 2020	low
4	NBm44jknpXPhw0tYWwBg0TmepUq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123490214347839	Wed Nov 18 16:32:34 GMT+08:00 2020	low
5	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:03:12 GMT+00:00 2020	low
6	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:03:13 GMT+00:00 2020	low
7	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:03:19 GMT+00:00 2020	low
8	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:03:22 GMT+00:00 2020	low
9	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:03:28 GMT+00:00 2020	low
10	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:03:34 GMT+00:00 2020	low
11	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:03:40 GMT+00:00 2020	low
12	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:03:46 GMT+00:00 2020	low
13	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:03:52 GMT+00:00 2020	low
14	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:03:58 GMT+00:00 2020	low
15	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:04:04 GMT+00:00 2020	low
16	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:04:10 GMT+00:00 2020	low
17	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:04:16 GMT+00:00 2020	low
18	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:04:22 GMT+00:00 2020	low
19	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:04:28 GMT+00:00 2020	low
20	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:04:34 GMT+00:00 2020	low
21	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:04:36 GMT+00:00 2020	low
22	XLI9GIUZ8DMAso00xrfAUy4OHbq1	Goldfish 3-axis Accelerometer	0,9.776309967041016,0.8123499751091003	Thu Oct 29 12:04:36 GMT+00:00 2020	low

- As a researcher, I want the csv files to display immediately when I change my filter criteria on the data retrieving page so that I don't have to waste time waiting for results to be generated.

Implementation status: Prefetch file information from the Firebase when the back-end web system is loaded such that when the user carries out the filtering, the first 10 rows of data can be done and will be shown in the 2nd step page in the Data Filter page without querying the backend.

Issues created on bitbucket:

1. Web: Get filter conditions and fix web page bugs

Constraints (technical or other)

Limitation in testing mobile app

As all team members are non-Android users, the team is relying on emulators provided by Android studio which only has the Google Pixel series as the available model. It is well-known that different vendors modify the Android API in different ways in order to gain control over the system functionalities such as background tasks management. Thus, we cannot fully predict the behaviour of the app for Android phones produced by other vendors.

Furthermore, testing for sensor data reading and network transmission is a huge challenge for the team as well, especially if we want to ensure that the data collection service can be keeped alive 24/7 even when the foreground application is killed. Thus, the extent and robustness of testing is constrained by the limited resources the team have.

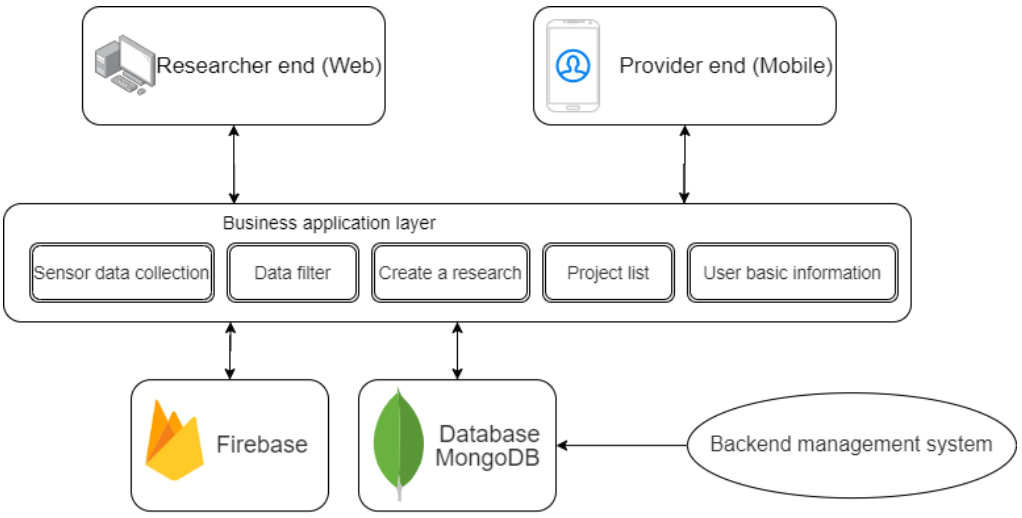
Concern of battery consumption of the mobile app
Different from web and desktop applications, mobile phones usually have limited processing power and rely on batteries to carry out daily work. Battery consumption of a mobile app has been a major concern of most mobile users. In response to that, Android has been increasingly strict on battery-draining activities such as background services and sensor listening for the newer versions. On one hand, we have to make the app power efficient which contradicts the implied object of the app to collect as much data as possible. On the other hand, due to the restrictions imposed by Android, the team has limited choices to implement the long running background service if we are targeting the latest release of Android and there is no guarantee that the service would function for the newer releases or future releases. Thus, we may have to limit the Android version we are targeting, if we want to achieve guaranteed functionality, predictable and reproducible behaviour of the application.

Privacy concerns of mobile data contributors
Privacy has been a huge concern for many mobile users nowadays, especially when it comes to activities or apps similar to our mobile application. Therefore, we have to limit the amount of personal information we require from the user. This would inevitably limit the potential extension of the services provided by the researcher facing web applications such as filtering on user information.

Limitations in adopting automated testing and CI/CD pipeline
The Bitbucket Pipeline is a powerful tool which integrates well with the software stacks and potential deployment platform used for the project. However, it has a limit of 500 minutes of free usage time. We verified that a build of the Android subproject alone typically takes 5 minutes to complete. It is very likely that we run out of free usage time if we do it for every commit. Thus, instead of having automatic test, build and deployment for every commit, we can only run the pipeline script manually for major releases.

Limitations in using ECMAScript 6
ECMAScript is a general-purpose programming language, standardized by Ecma International according to the document ECMA-262. This is a JavaScript standard designed to ensure the interoperability of web pages between different web browsers. However, due to webpack compatibility issues of ECMAScript 6 (ES6, the second major version of JavaScript), some react-native libraries provide uncompiled ES6 code, and ES6 code needs to be compiled before it can be run by Jest. All these make it impossible to connect to Firebase directly. We connected to Firebase in another way, CDNjs.

Figure 1: System architecture diagram:



No implementation of remaining user stories with reasons

- As a provider, I want the app to alert me when my data has been used so that I am aware of how my data is being used and whether it is used appropriately.
  No implementation reason: Because we did not have much experience linking to Firebase, we did not have enough time to complete the database and complete the operations that were too complex.

- As a provider, I want the app to make it obvious how to contribute data and change data settings so that even though it may be my first time using the app, I can easily get started and contribute.
  No implementation reason: Using the OnboardSupportFragment feature of Android to display the basic functionality and workflow of the app when the user opens the app for the first time or is not logged in. We are not implementing this function because we are not familiar with this feature and it's pretty hard to implement.

Key changes requested by first client deployment:

Mobile terminal:
We mainly changed the display and control mode of the sensor on the mobile phone. And support multiple login methods.
Firstly, We changed the sensor controller from the original seekbar to selectbar which provides low, medium and high.
Secondly, In order to make the user login more convenient and fast, we provide third-party login and fingerprint.

Web terminal:
Changed the whole operation logic, the Dashboard will no longer be displayed in the login page, even if the Dashboard cannot be clicked. Many unnecessary user data input has been deleted, and only Emails, passwords and occupations are retained. Improve the information encryption function and hide the Login and Register Pages passwords. Modified and added the functions of the dashboard: "Create a project", "Project list" and "Profile" these three pages to facilitate researchers to publish studies or collect data. A new Firebase database is added to store and retrieve study and Mobile sensor data. Improved sensor frequency drop-down search boxes, allowing users to select multiple sensors in the same frequency group (Low, Median or High Group) at the same time.

Extra notes on design elements

Since this cross-platform app is essentially a data collection system, we should be transparent in operation. For example, when the mobile system is installed for the first time, the user needs to be asked whether to open the specified sensor data for acquisition and analysis. Furthermore, data security must be high, and data encryption must be implemented to avoid data leakage. The system should be concise and clear, easy to read and operate, which means lower learning costs.

System structure overview

Overall, the IoT Data collection platform consists of 4 main components as shown in Figure 1.There will be a mobile app for collecting sensor data from the users. The data records will then be sent over the networks regularly to the RESTful microservice server which then persists the data to the Firebase. On the other hand, we have the researcher-facing web application which acts as the “consumer” of the data produced. When researchers login and register, the web application will fetch and send information (Emails, passwords and occupations) through using MongoDB. It will query the different microservice server (Firebase) for data transmission and retrieval services. The server would then fetch and send the required data from the database and render it to the web application, and researchers will get the data feedback.

We decided that the project application needs a scalable database, a flexible backend microservice, and a Database as a Service platform for the backend. It means that all data operations (such as data providing, filtering, and querying) will be completed with the microservice. Since the purpose of this application is to share and manage data sensed from smart devices (eg, smartphones, smart watches), we would expect intensive network communication between a large number of devices to the server. As a crowdsourcing project in nature, the server should be highly scalable and able to withstand intensive concurrent access as we would hope there will be as many people contributing as possible.

For the system structure of the mobile terminal, users (data providers) need to manually authorize whether to accept app privacy access, such as location information. The second is to obtain the username and password information from them through the login page then send a request to the Firebase and receive a password and compare with the user provided password to check whether the data matches. If there is no match or there is no account, the system will ask users to register an account. The email and password will be transferred to Firebase as relevant data. When users successfully log in, the mobile app will jump to the "Configuration" page. We implement third-party login by connecting to the google API and fingerprint login which use the same fingerprint interface with phone screen unlock. At the same time, our app will read user personal information from the database such as surname, weight and so on. The user can adjust the data upload frequency according to their preferences and get the system score which is calculated based on the frequency of data uploads, power consumption and other influencing factors. When the user opens some sensors, the app will through the API interface to obtain relevant data. And the app uploads data according to the information selected by the user and manually uploads it to Firebase

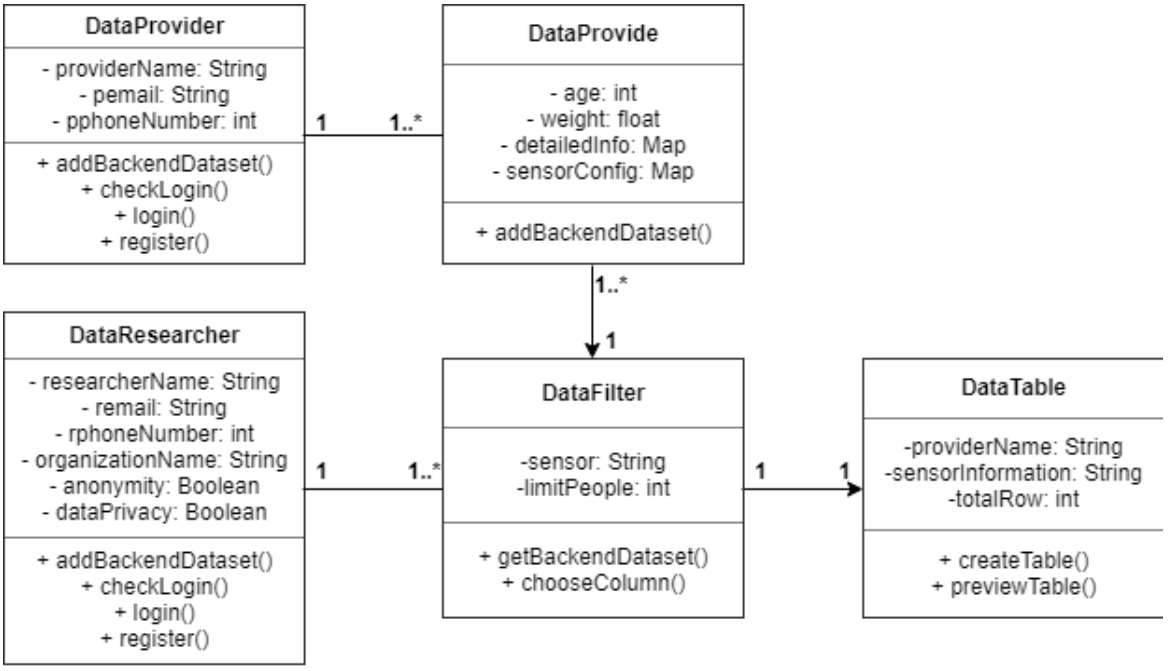
by the button at the top of the Mobile "Dashboard" page so that data is invoked and analyzed on the Web end. Moreover, the web end will be based on the user's habit to send the recommended project according to their preferences.

Similarly, researchers also need to login to the web page by providing username and password information so that the Backend server Node.js can pass them to MongoDB for comparison. If they do not match or no account exists, the system will prompt a corresponding error warning. In the "Register" page, researchers only need to supply 3 personal information: unique Email, passwords and occupations. If Email is not unique or test formats are incorrect, researchers will be asked to correct the information. When these above conditions are fixed, the system will pop up the window to mention researchers that register successfully. Then they can use the account and password to log in, if the Email and passwords are correct, the web system will jump to the "Data Filter" page through the Backend server. Researchers can choose sensors of different frequencies according to their needs, using the dropbox to choose different frequency sensors is more effective than before. When clicking on the next step, the application will show the first 10 rows of sensor data. It is convenient for researchers to decide whether these data fit their research requirements or not, if this data is fit, they can directly click the download button. Web system will obtain sensor data through Node.js and Firebase, and generate a data CSV file and a link for researchers to download. Otherwise, they can turn to the "Create a Project" page to publish new research. The second page is to create new research. Researchers need to customize the title of the data table, their own name and this study background, and use the sensor search box to determine which sensors and corresponding frequencies need to be obtained and analyzed. The backend server will look for these data in Firebase based on these requirements. When researchers press the publish button, the application will transmit this information to Firebase. The mobile terminal also will have a prompt message asking the data provider whether to join the study through the Firebase. The sensor information selected through the "Data Filter" page, which avoids that sensor frequency information is repeated filling in by researchers. For the “Project list” page, Web gets research information from Firebase and shows the study published by the researcher before. When they click the refresh button, it will recollect these data from Firebase and show project names and descriptions in this page. Turn to the last page, researchers can get their personal information through MongoDB (Register Email and Occupation).

For the back-end server, it is equivalent to a glue that connects Mobile and Web terminals through Node.js, Express, Firebase and MongoDB. This is essential for cross-platform software development. Mobile end uses Java and Android SDK for development, Web end

uses Vue.js framework for production, they can transmit data to Firebase and MongoDB for processing and calling through Node.js and Express. So that our database cross-platform transmission efficiency has been significantly improved.

System component diagram (UML):



Quality of Work

a.Evaluation

a.1 Testing

Testing plan

1) Introduction

For the whole project, there are two parts that will be tested, web application and mobile application. For both applications, firstly, each app development team tester will go through a usability test, security test, acceptance test, regression test on their application. Finally, crash and stress testing, and performance and reliability testing will be implemented by the project tester. Unit testing and APIs testing is partly completed for our app, because we use a large amount of APIs, and strongly follow the APIs guidelines, to do unit testing and APIs testing for each API need to read original API code, it will consume much time. Most importantly, since we need to do two apps, it is difficult for us to complete the whole project in limited development time.

1.1 Scope

The user functional requirements will be tested, however some of the non-functional requirements may not be tested due to the project timeline being too short and the requirements kept changing all the time. The testing process is implemented step by step as the project development .

1.2 Out Of Scope

Meeting agenda and also each week meeting minutes will not be tested, but will be reviewed by the tester each week whether the document has been recorded correctly.

1.3 Roles and Responsibility

Since the whole project development team only contains 6 members, therefore, the whole team members will participate in programming. For different roles, the team member will sign a group contract firstly from week 2 to week 4, during that process the role will be assigned, but also rotate the role in different weeks to see who could fit in which roles very well. After week 5, Arvin and leo will be the tester for testing the rest following weeks.

2) Test Methodology

2.1 Overview

Since we are using extreme programming in this project, which means the whole project will be divided into different small pieces of engineering tasks. In different cycles the tester should perform certain tests to make sure the different parts of the project works normally and correctly.

2.2 Test Levels

Firstly, the usability test, security test and acceptance test will be implemented after the development. Moreover, the performance and reliability testing ,Crash and stress testing, and regression testing will also be tested with the project development. Finally, if possible, we should let users test our program to see anything we need to improve.

2.3 Bug Triage

For each bug, we have a list of issues on the bitbucket, for which issues are waiting to be solved by when, or which issues are solved by who and when. Moreover, this issue list should have the priority, which bug should be considered first, for example, the priority of each issue for which issue is the major, which is the minor concern.

3) Test Deliverables

The initial test plan for the whole project has been written. Moreover, each week the team will have a test for the new adding function, and also do the regression testing for compatibility with existing functions.

4) Resource & Environment Needs

4.2 Testing Tools

The junit4 will be used for testing the mobile application. For the web application, the jest may be used to test the vue framework. For the bug tracking tools, our team used the bitbuckets issues as the tools to record the issues.

4.3 Testing Environment

The project for the mobile application will be tested on the android mobile(maybe the real mobile or VAD), the project for the web application will be tested on the web side. Finally, the whole project will be tested under a self-created server.

Relevant Testing Techniques

Equivalence Partitioning Test Cases Design Technique:

- Test a field which accepts correct Email address format

Email

EQUIVALENCE PARTITIONING		
Invalid	**Valid**	**Invalid**
12399494	m33asdow@icloud.com	asd123k@adsad

Valid Input: before @: contain name of mailbox after @:contain valid domain name
Invalid Input: does not contain @ or does not contain valid domain name

Valid Class: Enter email contains @ and has valid domain address: m33asdow@icloud.com
Invalid Class: Enter email does not contain @ : 12399494
Invalid Class: Enter email does not contain valid domain name after @ : asd123k@adsad

- Test a field which accepts the length of Password within range 8-16 positions

Password

EQUIVALENCE PARTITIONING		
Invalid	**Valid**	**Invalid**
asd123	asd123456	asdfghjkl123456789

Valid Input: within 8-16 positions
Invalid Input: less than 8 positions or larger than 16 positions

Valid Class: Enter password within 8 - 16 positions : asd123456
Invalid Class: Enter password less than 8 positions : asd123
Invalid Class: Enter password more than 16 positions : asdfghjkl123456789

Boundary Value Analysis Test Cases Design Technique:

Test a field which accepts the length of name within range 4-25 characters.

Name

BOUNDARY VALUE ANALYSIS		
Invalid (min - 1)	**Valid (min, +min, -max, max)**	**Invalid (max + 1)**
3 characters	4, 5, 24, 25 characters	26 characters

Minimum boundary value is 4

Maximum boundary value is 25

Valid text length is 4, 5, 24, 25

Invalid text length is 3, 26

Test case 1: Text length of 4 (min-1) = Invalid

Test case 2: Text length of exactly 4 (min) = Valid

Test case 3: Text length of 5 (min+1) = Valid

Test case 4: Text length of 24 (max-1) = Valid

Test case 5: Text length of exactly 15 (max) = Valid

Test case 6: Text length of 26 (max+1) = Invalid

Quality constraints

Mobile:

1. Only raw sensor data is collected.
2. According to the new policy and restriction imposed by Android in API level 26 and above, the data reading process may be killed at any time when the foreground app is killed.

Web:

From web send notification to any android Device there are only two ways:

1. Push 2. Pull

For Push: Firebase Cloud Messaging (FCM).

For Pull: Mobile apps continuously keep tracking and connecting to the server and trying to fetch data if it is available from there.

Advantages of GCM: Low battery consumption. Since it will push the message to the user device and it will push the message when the user gets connected to the server. Both push and pull are not easy to complete.

a.2 Detail of tests

Acceptance testing:

For Provider:

1) As a provider, I want to provide as little personal information as possible when registering for the application so that I am certain that my personal information is not misused.

Acceptance tests:

For normal cases, our group should test whether we have a button to lead users to register a page, and also whether we have certain blanks for users to type their email, phone, name and password.

For boundary cases, our group should test for whether there is too much information(more than email, phone, name and password) on the register page which is unnecessary for user input.

For the abnormal cases, if the register page is disliked somehow, we will test whether we have the additional option to fix this problem or remind the user how to register.

2) In order to ensure data security, users want to be able to log in securely using a username and password.

Acceptance tests:

For normal cases, our group should test whether we can log in with the username and password.

For boundary cases, If the user uses a really simple or complicated username and password, do we allow the user to login.

For the abnormal cases, our group should test if we cannot login with the correct username and password, does it support any optional methods?

3) As a provider, I want to control what sort of data I will be contributing so that I can avoid exposing my privacy and sensitive information.

Acceptance tests:

For normal cases, our group should test whether we have a linked to sensor setting page. Our group should test whether the user can control the on/off sensor switch and also the sensor state will remain on what user setted.
For boundary cases, we should test if all switches are on or off, if the user switches one of them to a different state, is the app still stable as usual or not.
For the abnormal cases, our group should test whether we have additional options to see if the switch on/off cannot be switched anymore.

4) As a provider, I want to be able to set how often that sensor data is collected so that I can restrict the amount of battery usage of the app.
Acceptance tests:
For normal cases, our group should test on the frequency that the data has been collected which has been setted by the user, whether it is satisfied with the battery requirements.
For boundary cases, our group should test if the frequency setted becomes really high, whether the battery usage still corresponds to the requirements.
For abnormal cases, our group should test if the user missed the frequency to an impossible number, do we have the option to fix the problem, and test this option whether it can solve the problem or not.

5) As a provider, I want to have some incentives or rewards for contributing my data so that I find sacrificing my phone battery for data contributing worthwhile.
Acceptance tests:
For normal cases, our group should check whether the rewards have been recorded correctly, and give them correctly by rules.
For boundary cases, If the user set the highest amount of data collecting frequency or lowest frequency, whether the reward points are recorded correctly.
For the abnormal cases, our group should test if the points have been recorded incorrectly, does the additional option is suitable to fix this problem.

6) As a provider, I want to have a summary of what data and how much data has been collected from me so that I can easily understand how much am I contributing and make a decision on whether to change the frequency of data collection.
Acceptance tests:
For normal cases, our group should test whether we have a link to the dashboard pages, and also whether we have displayed the data collection information reports, and the reward points.
For boundary cases, we should test whether we cover all the data that is displayed on the reports.

For the abnormal cases, we should test whether we have additional options if the data displayed is not accurate or displayed all the data information.

7) As a provider, I want the app to have a function that allows me to upload my data manually, so I can control how to use my data.

Acceptance tests:

For normal cases, we should test whether there is an option for users to upload their data, and whether the data being correctly uploaded.

For boundary cases, we should test whether if there is no data, whether it can be uploaded or not.

For the abnormal cases, we should test whether we have additional options if we cannot upload the data correctly.

For Researcher:

8) As a researcher, I want to download the data provided in a commonly used format such as csv or excel so that I can use them immediately for my project without additional preprocessing work.

Acceptance tests:

For normal cases, our group should check whether the downloaded csv or excel file are corresponding with the data on the sites.

For boundary cases, if there are too many files downloaded, we should check whether each file data is correct.

For the abnormal cases, if the downloaded data is incorrect, does the additional option will fix this problem.

9) As a researcher, I want the csv files to display immediately when I change my filter criteria on the data retrieving page so that I don't have to waste time waiting for results to be generated.

Acceptance tests:

For normal cases, our group should check for results displayed whether in a certain time or not.

For boundary cases, our group should check if the data is too large, whether the result will be generated in a certain time.

For abnormal cases, if the result cannot be generated in a certain time, we will test if the additional option could fix this problem.

10) As a researcher, I want the process of finding and downloading data to be intuitive and easy to follow so that I don't have to spend too much time learning how to use the service.

Acceptance tests:

For normal cases, our group should check whether we have correctly registered and login into sites. And whether it is easily to download the data.

For boundary cases, for different sex and age, we will test does it will influence the complexity of data download process.

For abnormal caess, if it is too hard to download the data for researchers, will the additional option fix the problem.

11) As a researcher, I want the drop-down search box to be more convenient for me to select different frequency sensors, and when we choose a specific one in low frequency, this one can not be selected in other two frequencies.

Acceptance tests:

For normal cases, our group should test whether when one type of frequency sensor is chosen, the other two different frequency sensors cannot choose this type of sensor.

For boundary cases, our group should test when we choose all the sensors for one type of frequency, whether for the other two different types of frequency, there is no sensor can choose.

For abnormal cases, we should test if one sensor can be chosen in two different types of sensor, then whether we have any solution for this kind of problem.

12) As a researcher, I want the web application to have a service which allows me to search or filter based on data types so that I can avoid downloading data that is irrelevant to my project or research.

Acceptance tests:

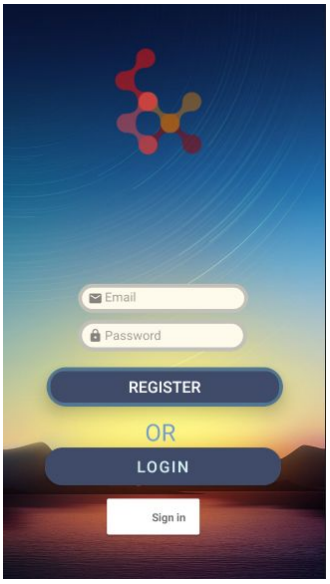
For normal cases, our group should test whether the search and filter could filt the correct wanted data.

For boundary cases, our group should check whether the search input is null or blank, does the correct output will show.

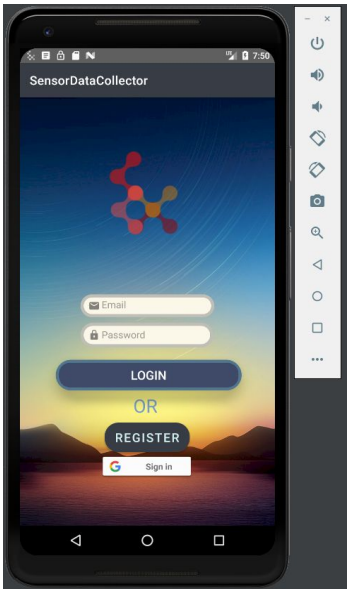
For abnormal cases, if the unwanted data has been searched, does the additional option will fix the problem.

Usability testing:

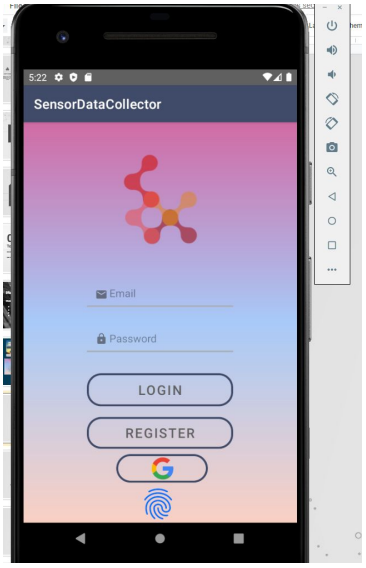
For the Login page
An important usability issue was discovered which was that the login button was below the register button, it means that the users probably will press the wrong button after typing their Email and passwords. Then we fixed the issue as you can see from the middle image here,we put login button beyond the register, however, the usability issue still exists, which the color of google login button is not consistent with other buttons, as well as the size of buttons are also not consistent with each other, and the word OR between login and register button looks weird. Now, we fix all these usability issues.



Before

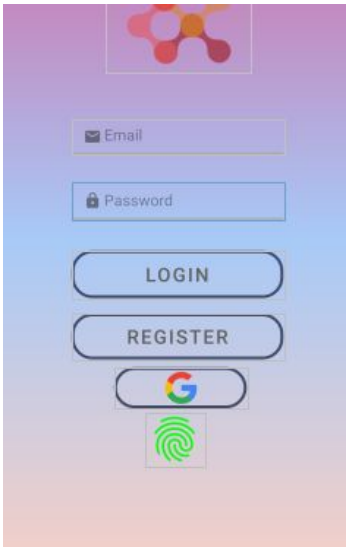


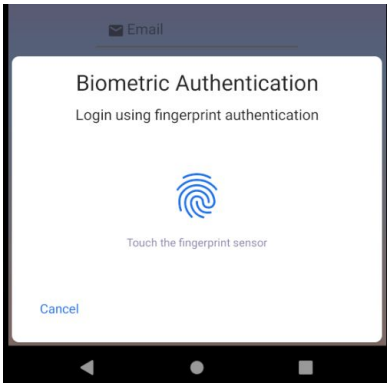
After

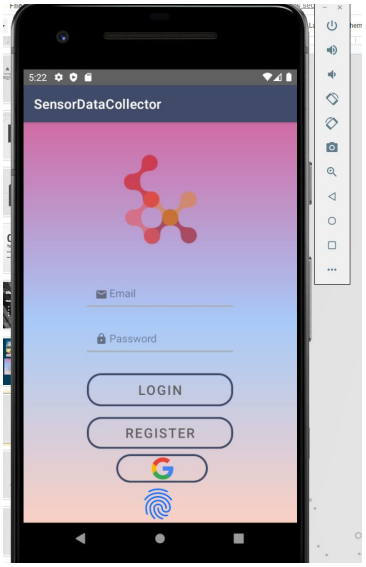


Now

The other usability issue is that the color of the fingerprint icon is not consistent with the pop up window fingerprint icon, one is green, the other one is blue. we change both colors to be the same, which is blue.



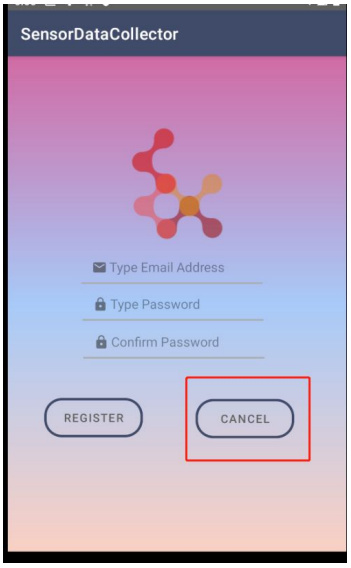




Before

Now

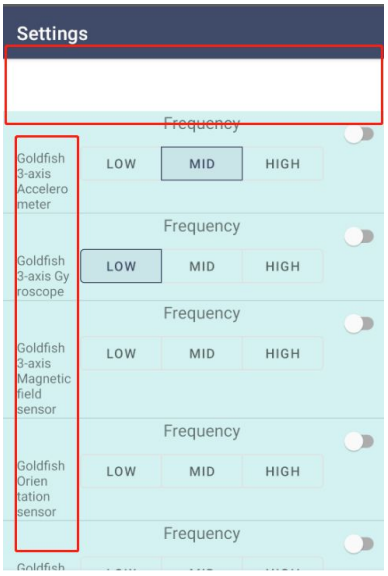
For the Register Page, there was not a cancel button before. We realised that some users may misclick the register button on the login page, which the user is unable to go back to login. So, we fixed the issue and added a cancel button, as you can see from the image, and now the user is able to go back. Also we make the register page consistent with the login page. Same color and style.



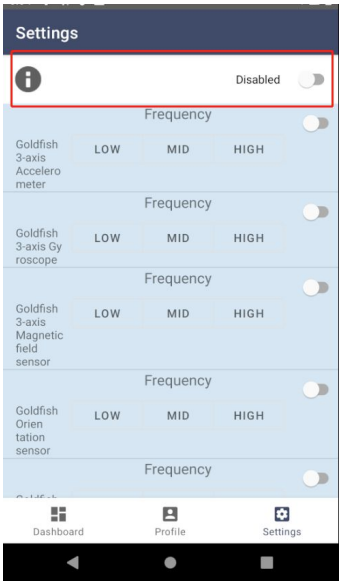
No cancel button before

For the main page, the usability problem is that the sensor name is not clear enough to the users, And there was also white space under the settings.

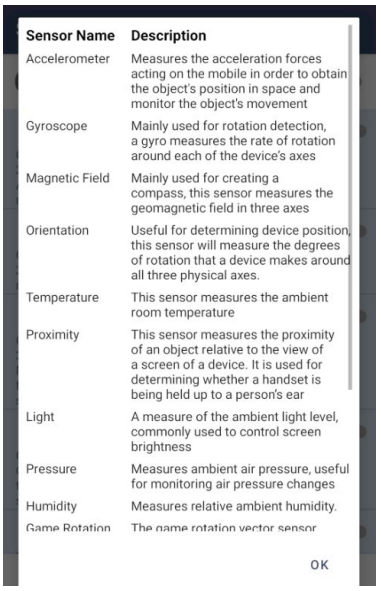
We then add an information button at the top left corner, after clicking that you will see the description for each sensor, however, we may change the exclamation(x mation) mark icon to the question mark icon, it would make more sense.



Before



After

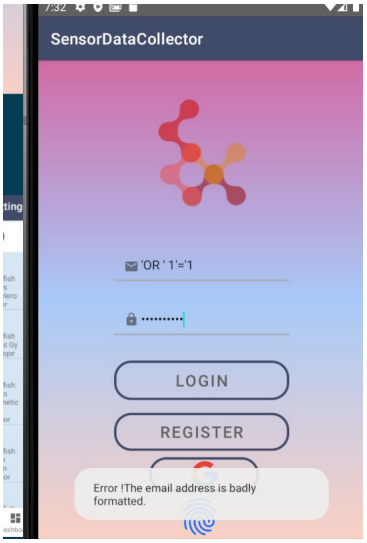


Sensor description Page

There are only two usability problems for the web, which at the home page of the web, there is no log out button, and no web title.

Security testing:

We did SQL injection on each input box, and it works well, there is no bug found, and the user is unable to go to the home page if he did not login.



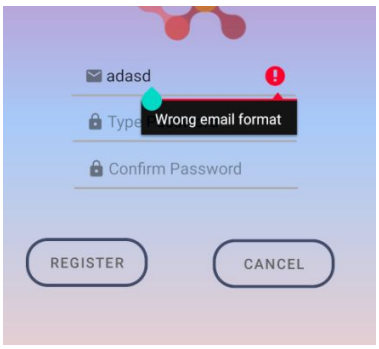
SQL Injection

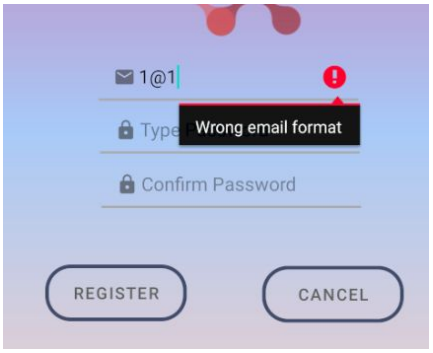
For other Security testing, we found that if the user login by google account, he is unable to logout. We found that the issue is that we only tried to log out the normal account, but we did not logout the google account.

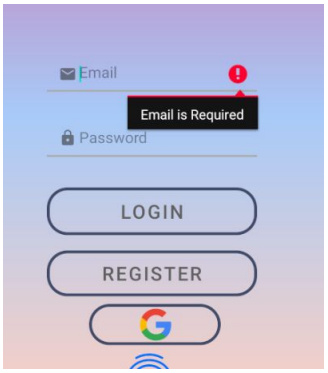
```
private void logoutBtnOnClick(){
    FirebaseAuth.getInstance().signOut(); //; log out firebase

    GoogleSignIn.getClient(  //google sign out
            getContext(),
            new GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN).build()
    ).signOut();
```

LogOut Error

We checked all input, no matter in the registration page or login page, such as shown in this image. We check the email format, and reject any empty inputs.

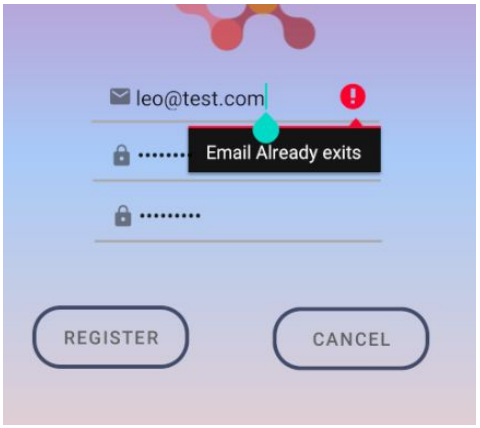


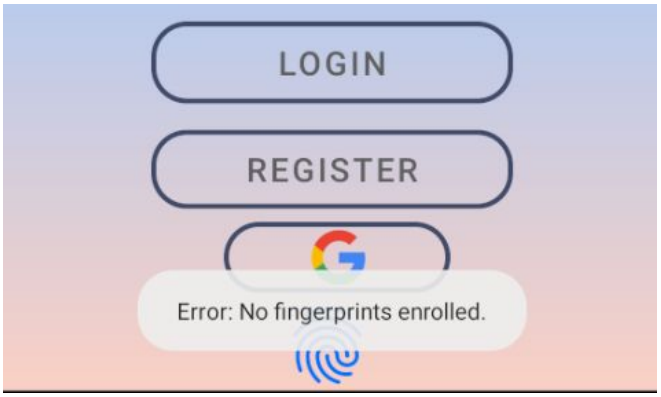




Email Format Check Empty String Check

as well as instant check if the email already exists, and pop up an error message if the user did not set up his fingerprint and clicked the icon.



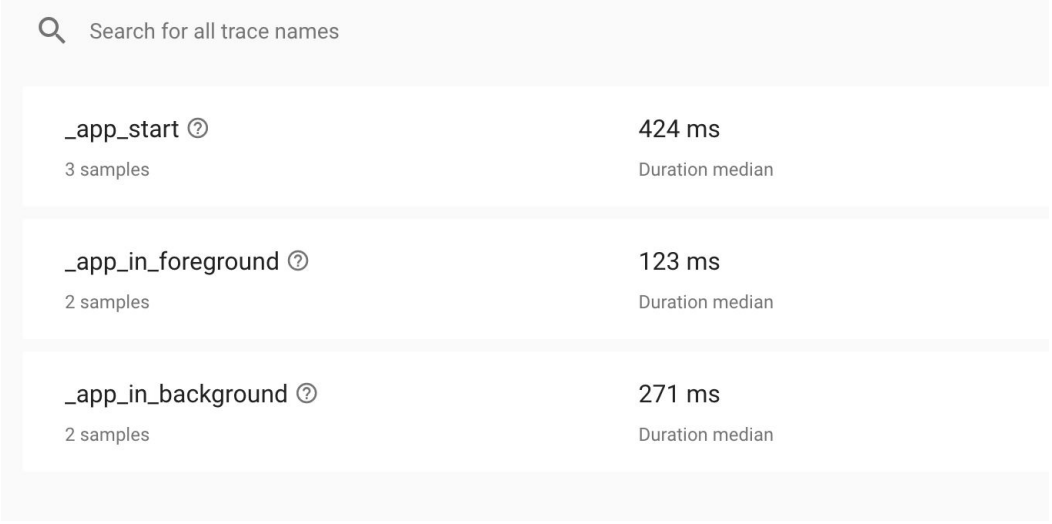


Instant Check fingerprint Check

For the web Security testing, We did SQL injection and XSS attack, the web is able to defend both attack methods. On the register page, 1@1 email format is able to register, which is not secure enough, and by changing the URL page name, it can lead to unusual pages, and sometimes can go to the project list page without login.

Performance and reliability testing

For the Performance and Reliability Testing, the Performance Monitoring SDK is added to our mobile app, our performance is quite good, the app start time is 424ms meaning the time from when the code is initialised until the UI is responsive.

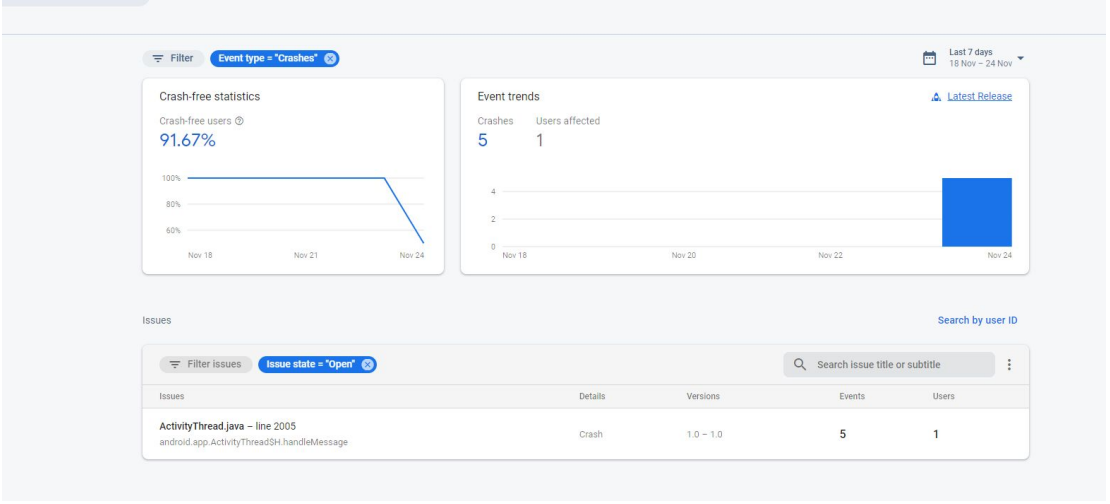


Performance Monitoring Page

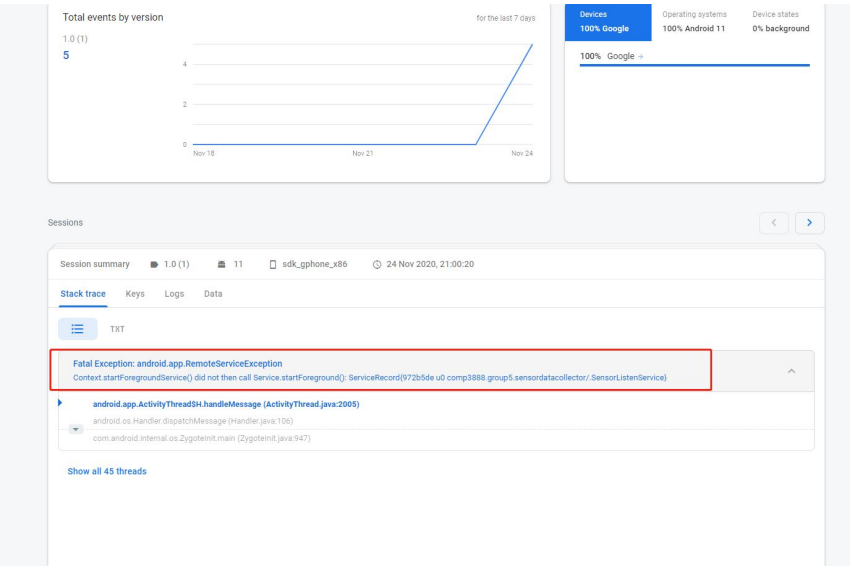
[Performance Monitoring Portal]

Crash and Stress testing

With the Crash Monitoring SDK, we are able to collect, analyze and organize app crash reports. And also to see the reason why it crashed, as you can see from the image, there are totally 5 crashes experienced by only one user, which is me to crash it purposely, around 92% of users who have not experienced a crash during the entire time, it should be 100% percentage if exclude myself. Once you click the issue activityThread.java you will be able to see the crash reason.



Crash Report



Reason for Crash

[Crash Monitoring Portal]

Regression testing

For the regression testing, we did it for each version release. As I mentioned before the sign out error, which is logging from google account is not able to log out, we resolved that issue. And later we test again about fingerprint login, it works well, it doesn't impact all other

functions such as login and third party login. For the other function, during development, it all works appropriately.

Web Testing Video Demo

Note that there is Web Testing Demo Video (Appendix 4), that includes usability, security and others testing.

a.3 Conclusions

Test summary and limitations

For the usability testing, it provides a better user experience and offers insight into how satisfied users are with our app, it also identifies problems within the app which may not have been obvious otherwise. However, the limitation for usability is that we testers may have bias for both conscious and unconscious that include personal opinions and preferences.

For performance and reliability testing, it provides benefits for user experience, which the app is more stable, by checking the performance the developer is also able to improve it if it is too low.

For crash and stress testing, this testing method will prove that our application will not be crashed if there are too many users online at the same time.

For the regression testing, this testing method will prove that the new version of application will not break the existing functions.

For the security testing, this testing method will prove that our web and mobile application will be more secure, and will decrease the attacking from the malicious code. And also will protect our user and research database, make it more stable and reliable. However, the limitation for security testing is that there are only few secure testing methods being implemented which means there are many ways to crack the app.

For the acceptance testing, this testing method will prove that we achieve all the user requirements. However, the limitation for the acceptance testing is that we only focus on part of the user story, because we hardly concern all the needs for all types of users.

System Limitations

The limitation for our app system is that team members have different backgrounds, and programming experience and others, hence the creativity and productivity may change according to developers experience, and the app may be not creative, the app may only be understood by professional users, and another technique limitation is that the app can only run at below API level 26, because the data reading process may be killed at any time when

the foreground app is killed if the API level is above 26. More limitations can be found below sections.

b.Tools used to build systems.

Summary of tools for development:

Development type	Tool/Language/Framework
Mobile Development	Java and Android SDK
Mobile backend and storage	Firebase
Web Development	Vue.js framework
Web end Storage	MongoDB managed on mLab
Backend web server	Node.js with Express
Source Code management	Git, Bitbucket, Bitbucket Issues and Bitbucket Wiki
Continuous Integration and Deployment	Bitbucket Pipeline, Docker
Team Communication	Slack

Language and framework:

- Java and Android SDK:

Firstly, the team chose to use a native development framework (e.g. Android) rather than a web-hybrid framework because the nature of the app involves a substantial number of interactions with sensors which are native features of a phone. Hybrid frameworks such as React-native are usually robust in web-oriented features but may have poor support when it comes to interacting with hardwares in a mobile device.

Secondly, the team chose to focus on Android development rather than targeting multiple platforms or using a cross-platform framework such as flutter. It is because we foresee that sensor configuration and the means for developers to interact with sensors may vary drastically across different platforms or even device models. The team would bear high risk and the amount of workload would be too heavy for the team to figure out how to deal with sensors on different platforms. Focusing on one platform would allow us to develop features more quickly.

- Vue.js:

The team decided to use the Vue framework to build the web application. Compared to other frameworks such as React and Angular, Vue has a smoother learning curve with comprehensive documentation available. As most of the team members did not have any experience with front end framework, it is really important to choose a framework that is easy to learn and understand so we can get started on the development as soon as possible. Moreover, as a popular framework with an active community maintaining the framework, it is stable to use. In situations of development obstacles, it is likely that we are able to find online resources to solve our issues.

Similar to Angular, Vue implements the Model View View-Model (MVVM) architecture. UI components and its associated data are usually grouped in one *.vue* file in a declarative manner which achieves high cohesion and high readability. It is also well known that the two-way data binding feature of the framework makes it perfectly suitable for data driven projects and applications such as this project.

Thus, the team believes that developing the web application with Vue will allow us to focus on implementing the functional requirement aspect rather than wasting precious development time on learning and code management or refactoring.

- Node.js and Express:

The team decided to establish a Node server with the Express framework. Similar to Vue, Express is also a very simple and minimal framework that the team can easily get started on. There are also abundant resources online which can guide us to quickly set up a structure which can communicate with both the frontend and database.

Since both Vue and Node projects are based on Javascript and can be managed by Node Package Management (npm). It takes a small amount of effort to manage, integrate and deploy the two projects together with npm.

Database tools:

- MongoDB:

A No-SQL database is suitable for this Project. As an Agile team, the software is developed in iteration. Thus, it is highly likely that the database schema is going to change very often due to change or addition of requirements. Using a No-SQL database offers us the flexibility to experiment with different database design and respond quickly to changes in the development process.

As a distributed database, MongoDB also scales well. As the amount of mobile data collected can be enormous if the number of users grows large, the high potential for scalability of MongoDB is a major reason for choosing it. With mLab, a Database-as-a-service platform, the team can be freed from setting up and configuring the environment for MongoDB. mLab also provides the service for automatic scaling of the Database.

- Firebase:

Firebase is known as a Backend as a Service (BaaS) platform provided by Google. It has a flat learning curve. With Firebase, the team is able to implement secure user authentication, database CRUD operations by simply invoking relevant Firebase APIs. It also provides excellent web-based monitoring and visualisation services for free. Thus, even though the team was very new to Firebase half way through the project, we were still able to provide reliable and secure backend support for the mobile application within a week and maintain/oversee the status of the backend with little effort. This is also the main reason why Firebase was chosen.

Moreover, in the later stage of the development, we need to integrate the Mobile application with the Web application. Firebase integrates well with both Android and Node projects which is very suitable to be used in our project as the repository for data communication between web and mobile applications. As Firebase is a remote service and does not require a local environment to run, it simplifies the environmental dependency and deployment process when we render the whole project as an integrated system.

Source code management:

- Git and Bitbucker:

Git and Bitbucket are the suggested options by the course. They certainly help substantially with team collaboration and version management with robust features (especially code snapshot and branching) and integrated well with other tools and platforms. Moreover, both the team and the client are familiar with Git, thus the client can easily access and continue work on the codebase after the handover.

- Bitbucket Issues:

As an issue tracking tool, it is integrated in Bitbucket and with Slack. Thus, it is convenient for everyone in the team to track their tasks while developing.

- Bitbucket Wiki:

It is used to document the development status and team process. This will give the client a better idea of how the application is developed when the project is handed over.

Deployment tools:

- Bitbucket pipeline:

Bitbucket pipeline is used to deploy the Mobile end application by generating the APK file in the arifect. It is a very convenient tool to automate the build and deployment process. It is similar to docker in the sense that it is environment independent. It is even better because in docker you will still need to provide your own environment to run the script. But by using bitbucket pipeline, Bitbucket will create a virtual environment temporarily when running the build process.

- Docker

Docker is used to deploy the web application as it is platform independent and substantially automates the configuration and deployment process.

c.Information search/research and discipline knowledge use and application

Use of Design pattern

The Singleton design pattern for our mobile room local database is used. It ensures that the database only needs to be initialized once during the entire program operation, instead of a new instance every time it is retrieved, and the memory usage and garbage collection frequency are optimized.

Use of localStorage in Web application

The backend server is using node.js and localStorage as the local storage variable. Only need to store the data once after the web page is opened. It can also be used as an intermediate variable, allowing the frontend of the web to obtain MongoDB data.

Password encoding with Bcrypt

It is insecure to store the password as it is in the MongoDB. To improve the resilience of the system against password exposure especially in the process of data passing from MongoDB to the Node server, we researched and found out that Bcrypt is a popular module used in Node project to create and verify hashes of password. We used becrypt to encrypt password during user register before pushing it to the Database and using the API provided to verify the hashes during user log in. It helped us to solve this security issue.

Group processes, reflections and conclusions

Team Collaboration:

All below documents are documented on Bitbucket WIKI.

1. Group Roles
2. Group Contract
3. Meeting Minutes
4. Project Status Report
5. Slack Channel for team Communication
6. Wechat Group
7. Task allocation 1. Meeting Agenda(week2-week 7) 2. Backlog (week8- week12)
8. Issues on Bitbucket

Weekly routine

For the first half of the semester, the manager will prepare a meeting agenda before the meeting which will check the progress of previous tasks and allocate tasks for the next week. To make sure every task will be finished on time, we use Bitbucket Issues to sign tasks and remind members what needs to be done on time. Slack is also used extensively for further task clarification and progress updates. We also integrated Bitbucket issues with Slack such that all members are informed by Slack when an issue is created or completed.

For the second half of the semester, we had established a more stable weekly routine. Every Thursday,we will have our weekly client interaction in which we demonstrate user stories that we have implemented for the past one week and collect feedback from the client during the meeting. Then on Saturday, we will have a team meeting to consolidate all the new requirements and distribute tasks based on the new requirements. Lastly on the next Tuesday, we will have another meeting within the team to get an update of progress for the tasks distributed. We will be aware of what to present for the next client meeting.

Allocation of Task

For the first half of the semester, we separate into three groups includings: Mobile, Web and Back-end. Mobile group has three people because of Mobile part is the hardest part which needs to create a UI and set up sensors to collect data. The Web group needs to make a responsive web page and help researchers to retrieve data. The Back-end group is mainly responsible for processing front-end requests. However, it is not necessary to follow the group dividing all the time. Whenever a group meets some issues, others will help them to figure out the problem.

For the second half of the semester, all team members are shifted to develop the mobile application together. After the client approves that the mobile application is decent for deployment, all members are shifted to develop the web application. We believe that it is easier for members to communicate and cooperate when all members are working on the same part of the project for each iteration thus efficiency can be improved. To make sure all members are aware of the what tasks to implement individually and also the overall progress

for each iteration, we created a backlog for each week's team meeting to reflect who shall do what and what has been completed. We also assign tasks to members with Bitbucket Issues to keep track of progress and contribution.

Issues are created on bitbucket according to the allocation of tasks and their issues about past unfinished tasks or problems that need to be solved. (Evidence)

Communication with the client was done on a weekly basis. We have weekly client meetings and we primarily contact the client outside of these meetings through email. We send the client all of the important updates and documents. For example, we have sent the project scope statement (see appendix).
Evidence

Low compatibility Android API due to restrictions in newer API level

Different from web and desktop applications, mobile phones usually have limited processing power and rely on batteries to carry out daily work. Battery consumption of a mobile app has been a major concern of most mobile users. In response to that, Android has been increasingly strict on battery-draining activities such as background services and sensor listening for the newer versions. On one hand, we have to make the app power efficient which contradicts the implied object of the app to collect as much data as possible. On the other hand, due to the restrictions imposed by Android, the team has limited choices to implement the long running background service if we are targeting the latest release of Android and there is no guarantee that the service would function for the newer releases or future releases. Thus, we may have to limit the Android version we are targeting to 24, if we want to achieve guaranteed functionality, predictable and reproducible behaviour of the application.

Privacy concerns of mobile contributors

Privacy has been a huge concern for many mobile users nowadays, especially when it comes to activities or apps similar to our mobile application. Therefore, we have to limit the amount of personal information we require from the user. This would inevitably limit the potential extension of the services provided by the researcher facing web applications such as filtering on user information.

Limitations in terms of structure, design, implementation

Sacrifice flexibility for database migration
Since we are using Firebase as our mobile backend. It has a simple and clean API to use. However, it will be hard to migrate the data from Firebase to other Databases or changing the backend to other Databases such as MySQL or MongoDB. To migrate the database, the codebase has to be changed substantially.

Limitations in using ECMAScript 6

ECMAScript is a general-purpose programming language, standardized by Ecma International according to the document ECMA-262. This is a JavaScript standard designed to ensure the interoperability of web pages between different web browsers. However, due

to webpack compatibility issues of ECMAScript 6 (ES6, the second major version of JavaScript), some react-native libraries provide uncompiled ES6 code, and ES6 code needs to be compiled before it can be run by Jest. All these make it impossible to connect to Firebase directly. We connected to Firebase in another way, CDNjs.

One of the strengths of our implementation for the project is that little infrastructure support or environment dependency is needed to deploy the project. We extensively uses remote vender services such as Firebase to deploy mobile backends and mLab to deploy MongoDB for the web backend. Thus, we do not need to provide local infrastructure and environment support for those. To make the deployment process automatic and environment independent, we used Docker and Bitbucket pipeline.

When the team was just formed in week 2, members were generally unfamiliar with XP and the Agile development process. Thus, the quality of work was low and many required work are not properly documented. Members had problems understanding their role and did not fulfil what is required for the role. For instance, the responsibility of the tester was neglected and the test plan was only discussed verbally. However, with further research and practice of the roles, members gradually understand what is expected for each role and what else we should do other than development tasks. Documentation of work is improving as well. Members are making an effort to note down their research and document the research on the Bitbucket wiki.

The tools we use to control versions are Git and Bitbucket as mentioned above. All team members will post current issues in wiki and sign who will solve it. Once it has been solved, we change the state of this issue to resolved. Last but not least, understanding code without comments is essentially impossible. So,members need to add comments when they are coding and sign variables with a meaningful name.

To figure out how to organize achievable and successful goals, our product is determined by the content of meeting with clients. Before meeting with the client, we will prepare an outline of what we need to confirm with the last step and what we are going to do next. After meeting with the client, we will conclude bullet points about what we missed last week and design next week's plan.

References

Tridens Technology [Online image]. (2019). All about the Internet of Things (IoT).https://tridenstechnology.com/all-about-the-internet-of-things-iot/

Appendices:

A1. User stories

A2. Research, studies of similar systems

Research of similar systems

List of Potential Sensors for Data Collection

Research: Raw sensors v.s. Data from Samsung Health

Suggestion on System design and Workflow for Data delivery to client

A3. Acceptance Testing

For Provider:
Register with less information(acceptance test 1)
Using username password login(acceptance test 2)
Control which data upload(acceptance test 3)
Set data collection frequency(acceptance test 4)
Giving rewards(acceptance test 5)
Summary of data collection(acceptance test 6)
Upload Data manually(acceptance test 7)

For Researcher:

Download with csv or excel data format (acceptance test 8)
Using filter display csv file immediately(acceptance test 9)
Simply downloading the data(acceptance test 10)
Drop-down box to select sensor(acceptance test 11)
Allow user search or filter the data(acceptance test 12)

A4. Usability Testing and others.

Week2- Week12 Usability Testing Result

Week2-Week12 other Testing Result

Web Testing Demo

A6. Meeting minutes and status report

A7. Meeting Agenda

A8. Communication with the client.

BS Basem Suleiman <basem.suleiman@sydney.edu.au>
Fri 18/09/2020 3:23 PM
To: Oscar Lekovic
Cc: Abdallah Lakhdari <abdallah.lakhdari@sydney.edu.au>; Qiangsheng Gao; Tianye Tang; Linxiao Li; Mingjie Zhang; Chenyue Wang

Hi Oscar and the Team,

Thank you for the summary.

Could you please send me the updated version of the scope statement including the refinements we discussed in the last meeting.

Best Regards,

Dr. Basem Suleiman
The University of Sydney
Faculty of Engineering
2E- 233, School of Computer Science | The University of Sydney | NSW | 2006
+61 2 8627 6602 | +61 2 9036 0000 (fax)
Basem.Suleiman@sydney.edu.au | sydney.edu.au
CRICOS 00026A

OL Oscar Lekovic
Thu 17/09/2020 5:06 PM
To: Basem Suleiman <basem.suleiman@sydney.edu.au>
Cc: Abdallah Lakhdari <abdallah.lakhdari@sydney.edu.au>; Qiangsheng Gao; Tianye Tang; Linxiao Li; Mingjie Zhang; Chenyue Wang

Hi Basem and Team,

This is just a quick email summary of what we went over in the meeting. The meeting went well, and we really have some good leads to follow:

- Switch our key focus to mobile development - let's focus on setting up user profiles which will involve the selecting which data the user is providing (which raw sensors, which applications)
- Develop a pros and cons list for data collection from the following two sources of data:
  1. Raw Sensor data
  2. Mobile Application data
  Things to consider:
  - Processing time
  - Phone Battery
  - Scalability (multiple APIs vs reading raw data)
- Adjust user stories to make them more user centric - right now they are more app centric.
- For now, let's allow researchers to clean and filter the data how they want, we will simply just provide the user data to the researchers
- For now, focus on researchers querying one type of data (providers can still provide multiple types of data) - look to extend functionality later (data can be categorized by sensor or application)

Thanks for your time today,

Nice hearing from you Basem!

Oscar and the Team.

Basem Suleiman <basem.suleiman@sydney.edu.au>
Fri 11/09/2020 4:56 PM
To: Oscar Lekovic
Cc: Tianye Tang; Linxiao Li; Mingjie Zhang; Chenyue Wang; Qiangsheng Gao; Abdallah Lakhdari <abdallah.lakhdari@sydney.edu.au>
Hi Oscar and the Team,

Thank you for the detailed updates. That's really good to see quick update after I missed the meeting. I will try to join the meeting tomorrow, but no promise. We will set up the weekly meeting after tomorrow's meeting.

Very good progress. I have read through the documents and it looks good, it pretty much aligns with we discussed in the previous meeting. I will need to read again with focus on details. The initial mock-up looks good and it helps to visualise and think of how things could be improved. Very good approach!

For the questions:

- For the web app, we need a clarification on the filtering criteria. Can it be filtered by user info such as ages, or just filter by type/category of sensor?

>> Both where possible. There should be different filters to allow researchers/user to extract data based on their preferences.

- In terms of spreading the workload amongst the team, how soon do you expect work from a backend perspective to begin?

>> Let's probably wait a little bit on this. Once we refine the requirements of the app, we may start working on the backend. You may start envisioning the backend with initial design which may inspire/inform your ap/front design.

Hope this helps.

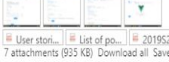
Thank you for the professional summary of progress!

Best Regards,

Dr. Basem Suleiman
The University of Sydney
Faculty of Engineering
2E- 233, School of Computer Science | The University of Sydney | NSW | 2006
+61 2 8627 6602 | +61 2 9036 0000 (fax)
Basem.Suleiman@sydney.edu.au | sydney.edu.au
CRICOS 00026A

Reply | Reply all | Forward

Oscar Lekovic
Fri 11/09/2020 4:14 PM
To: Basem Suleiman <basem.suleiman@sydney.edu.au>
Cc: Tianye Tang; Linxiao Li; Mingjie Zhang; Chenyue Wang; Qiangsheng Gao; Abdallah Lakhdari <abdallah.lakhdari@sydney.edu.au>



User stori... | List of po... | 201952_P...
7 attachments (935 KB) Download all Save all to OneDrive - The University of Sydney (Students)

Hi Basem,

It seems you have missed our scheduled meeting for this afternoon but do not worry!

Below I will provide you with a concise summary of what has been happening on our end of the project.

Report on group progress:

- We have drafted a list of user stories, and acceptance criteria based on the meeting last week (see attached pdf)
- We have discussed and chosen the developing tools for the project: Android SDK for mobile, node express for backend service, mongoDB as storage, vue framework for web application.
- We have assigned members to learn about the different dev tools and we have been learning individually for the past one week.
- We also looked at what sensor information can be easily obtained in android, and categorised. The list of sensors is in the attachment.
- We did construct a scope statement; we hope you can have a look at it in the attachment.
- We were planning to present the preliminary UI sketch of the phone app and the web application today to get some feedback.