

SMARTASTHMA
AN IOS APPLICATION DEVELOPMENT FOR INHALER USER

by

Renfan Yang

Submitted to the Graduate Faculty of

Swanson School of Engineering in partial fulfillment

of the requirements for the degree of

Master of Science in Electrical Engineering

University of Pittsburgh

2018

UNIVERSITY OF PITTSBURGH

SWANSON SCHOOL OF ENGINEERING

This thesis was presented

by

Renfan Yang

It was defended on

July 24, 2018

and approved by

Wei Gao, PhD, Associate Professor

Amro El-Jaroudi, PhD, Professor

Zhi-Hong Mao, PhD, Professor

Wei Chen, PhD, Associate Professor

Thesis Advisor: Wei Gao, PhD, Associate Professor

Copyright © by Renfan Yang

2018

SMARTASTHMA

AN APPLICATION DEVELOPMENT FOR INHALER USER

Renfan Yang, M.S.

University of Pittsburgh, 2018

According to the surveys from the U.S. Centers for Disease Control and Prevention (CDC), asthma continues to be a serious public health problem since an estimated 24.6 million people caught asthma and 6.1 million of them are children. Children are more easily than adults to be affected by asthma and 8.3% of children under age 18 years who currently have asthma. Therefore, it is necessary to find a way to make it easy and convenient that parents can understand their children's asthma situation and the teenagers can be informed with their progress on asthma treatment at real time.

With the development of smartphone that users can detect location, communicate through Bluetooth and access the internet through cellular or Wi-Fi, it is a great fit and convenient to let mobile record the daily data, such as the daily usage of inhaler, most frequent locations using inhaler and the Asthma Control Test (ACT) score history, for an inhaler user.

This thesis presents a framework of an iOS application developed by Swift 4.1 that realize the data collection, data visualization and report generation. Phone will receive the information via Bluetooth each time when inhaler is pressed. A new ACT score algorithm based on QualityMetric Inc. is created so that the score can be estimated without manually answering the questions and calculation. Users can view their score in a radar chart with suggestions or complete an exacerbation score questionnaire if they which zone (Red/Yellow/Green) they are currently in. All the data including users' accounts are backed up in the Firebase Database and the data will be back synchronized if user use a new device.

TABLE OF CONTENTS

1.0	INTRODUCTION.....	1
1.1	BACKGROUND	1
1.2	RELATED WORK.....	2
1.2.1	AsthmaMD and Propeller Health	2
1.2.2	Asthma Manager and My Asthma Pal	4
1.2.3	Asthmatic – the first asthma weather forecast	5
1.2.4	Asthma & Me	6
1.3	SMARTASTHMA	6
2.0	FRAMEWORK.....	8
2.1	DEVELOPMENT ENVIRONMENT	8
2.1.1	iOS.....	8
2.1.2	XCode	9
2.1.3	CocoaPods	11
2.2	BLUETOOTH LOW ENERGY (BLE)	11
2.2.1	Bluetooth Features.....	11
2.2.2	Peripheral and Central	12
2.2.3	Work Flow.....	13
2.3	LIBRARIES AND THIRD-PARTY FRAMEWORKS	14

2.3.1	ResearchKit.....	14
2.3.2	UIKit	16
2.3.3	MapKit.....	16
2.3.4	WebKit.....	16
2.3.5	Core Location.....	16
2.3.6	Charts	17
2.3.7	Firebase.....	17
3.0	SOFTWARE DESIGN	19
3.1	DATA FLOW.....	19
3.1.1	Data Initialization	19
3.1.2	Data Flow Diagram	20
3.2	APPLICATION WALKTHROUGH.....	22
3.2.1	Login and Sign Up	22
3.2.2	Allergies and Asthma Videos.....	22
3.2.3	Exacerbation Score Questionnaire.....	23
3.2.4	Data Visualization.....	25
3.2.4.1	Daily Usage Report	25
3.2.4.2	Most Frequent Locations.....	26
3.2.4.3	Asthma Control Test Score	27
3.2.4.4	Exacerbation Score History	30
3.2.5	Report Generation.....	31
3.2.6	Running at Background.....	33
3.3	DATABASE.....	34

3.3.1	Data Structure.....	34
4.0	TESTFLIGHT	35
5.0	CONCLUTIONS AND FUTURE WORK.....	36
5.1	CONCLUTIONS.....	36
5.2	FUTURE WORK.....	37
5.2.1	Asthma Game.....	37
5.2.2	Asthma Action Plan.....	37
5.2.3	Better Interface and Interaction Options.....	38
APPENDIX A		39
APPENDIX B		44
BIBLIOGRAPHY		61

LIST OF TABLES

Table 1. Comparisons between SmartAsthma and other asthma applications	7
Table 2. Layers in the iOS Architecture	9
Table 3. Comparison between original questions and new specific method	28

LIST OF FIGURES

Figure 1. AshtmaMD Peak Flow Meter. It is a device to measure your lung performance.	3
Figure 2. Propeller Health Bluetooth app and its cover.....	3
Figure 3. Check-in history and Asthma Action Plan in Asthma Manager	4
Figure 4. Functional portal and schedule screen in <i>My Asthma Pal</i>	5
Figure 5. Weather forecast in <i>Asthmatic</i>	5
Figure 6. Create an educational short video of a diseased airway onto a patient's video selfie.....	6
Figure 7. XCode workspace.....	10
Figure 8. XCode Interface Builder area	10
Figure 9. Content of Podfile.....	11
Figure 10. Peripheral and Central	12
Figure 11. Advertising mode	12
Figure 12. Demo running on laptop works as peripheral.....	13
Figure 13. Sample codes of the function used to get data from peripheral	14
Figure 14. Screenshots of Exacerbation Score Questionnaire	15
Figure 15. Sample codes of synchronizing data from database in function initiate().....	20
Figure 16. Data flow diagram	21
Figure 17. Login and register screens	22
Figure 18. Main screen and icon links	23

Figure 19. Asthma control zones and results	24
Figure 20. Samples codes of creating a questionnaire step	25
Figure 21. Daily Usage Report screenshots	25
Figure 22. Sample codes of function setChart()	26
Figure 23. Sample codes of function chartValuesSelected()	26
Figure 24. Locations and Log	27
Figure 25. Asthma Control Test Score Report screenshot.....	28
Figure 26. Current exacerbation score and its history.	31
Figure 27. Report shown in Web View and email sending screen	32
Figure 28. Replace the place holder with real data	32
Figure 29. Sample codes of setting up mailing details	33
Figure 30. Background modes settings	33
Figure 31. Flattened data structure example	34
Figure 32. iOS Builds on TestFlight	35
Figure 33. MR. NOSE-IT-ALL’S WORD GAME, from [27]	37
Figure 34. Asthma Action Plan details, from [28].....	38

1.0 INTRODUCTION

Due to environmental issues being more and more severe, Asthma keeps being a serious public health problem, according to the statistics from the website of the U.S. Centers for Disease Control and Prevention (CDC). It is estimated that 24.6 million people are suffering from asthma, of which 6.1 million are children. And by statistical analyze, children are more likely to be influenced by asthma. To make things worse, 8.3% of children under age 18 years among the whole is suffering from asthma. Therefore, SmartAsthma is designed to give parents an easy access to the asthma status of their child, or to facilitate adolescents to take care of themselves with better knowledge of asthma.

1.1 BACKGROUND

Data, in some situations, is more trustworthy compared to people themselves since people may be constrained by their own personalities and emotions but data is accurate and in detail. On the other hand, smartphones are more advanced than ever before with sensors to detect location, to exchange data through Bluetooth and to access the internet through cellular or Wi-Fi [1], which makes it a great fit to record our daily data, especially the health data.

With the technological development of communication and operation system on smartphone, mHealth (Mobile Health) is developed fast from eHealth (Electronic Health) [2]. On

the one hand, more and more health and fitness apps for iOS appear since the third party can access to the Health app via HealthKit, a framework provided by apple, after user provides permission to read and write health and activity data to their Health app. According to the statistic portal, there are 47,911 of mHealth apps available in the Apple App Store [3]. On the other hand, 89% of providers in both Arizona and California believed a suitable smartphone app might achieve the goal of adolescent asthma management, and 78% agreed that an app would help managing adolescent asthma in comprehensive way [4].

1.2 RELATED WORK

After some researches carried out about Apple Store, some knowledge about existing market is acquired. There are mainly 4 types of applications about asthma in the app store right now.

1.2.1 AsthmaMD and Propeller Health

AsthmaMD is a research application which support a specific device called AsthmaMD Peak Flow Meter [5]. The app is free, but users need to buy the hand-held device first. Also, the data must be typed manually after using the device. The figure below shows the meter which gauges how air being pushed out from lungs in one fast blast.



Figure 1. AshtmaMD Peak Flow Meter. It is a device to measure your lung performance.

Another app that being used corporately with a specific device is called Propeller Health [6]. It has a Bluetooth cover on the inhaler so that phone receives data each time when user press the inhaler. Propeller Health works well on the data collection and visualization, but it only shows the raw data and not provide suggestions and give users a referential current situation of asthma control based on the data.



Figure 2. Propeller Health Bluetooth app and its cover.

1.2.2 Asthma Manager and My Asthma Pal

Apps like Asthma Manager [7] and My Asthma Pal [8] are focus on providing knowledge of asthma and helping users record and remind their asthma action plan. Apps like these works as an assistant to inform you if you take your inhaler with you when leaving home or to notify you of your asthma action plan. This do make the life easier, but it will be better if the app could tell if you need to make an appoint with your doctor based on your daily inhaler usage.

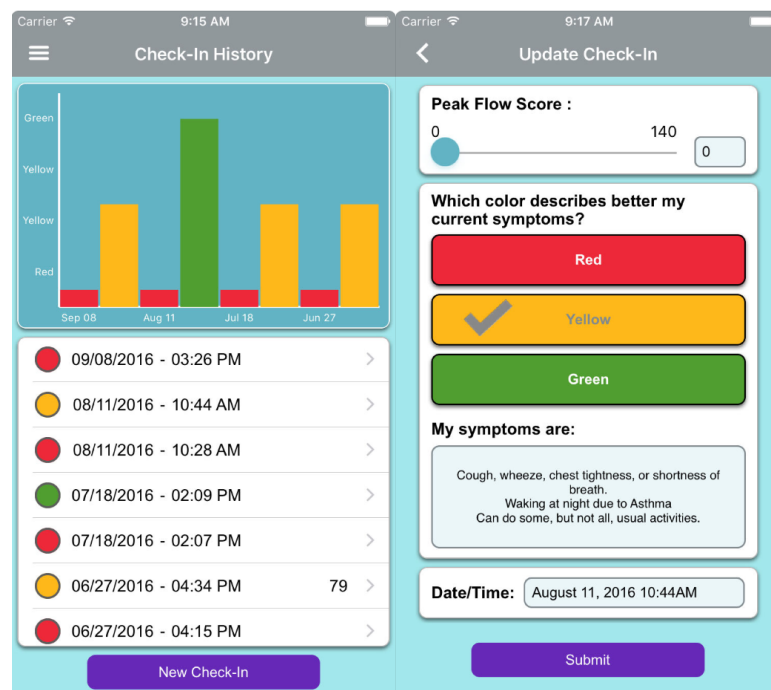


Figure 3. Check-in history and Asthma Action Plan in Asthma Manager

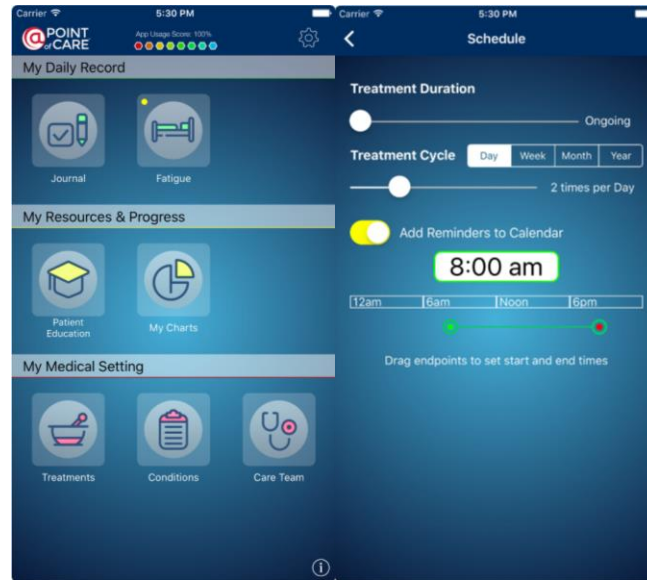


Figure 4. Functional portal and schedule screen in *My Asthma Pal*

1.2.3 Asthmatic – the first asthma weather forecast

Asthmatic [9] is an good example to prove that weather has a big impact on asthma patients, but this app is lacking data collection and analysis. Users can only have the weather forecast and prevent to go outside or reduce the change to be exposed outside when weather goes worse.

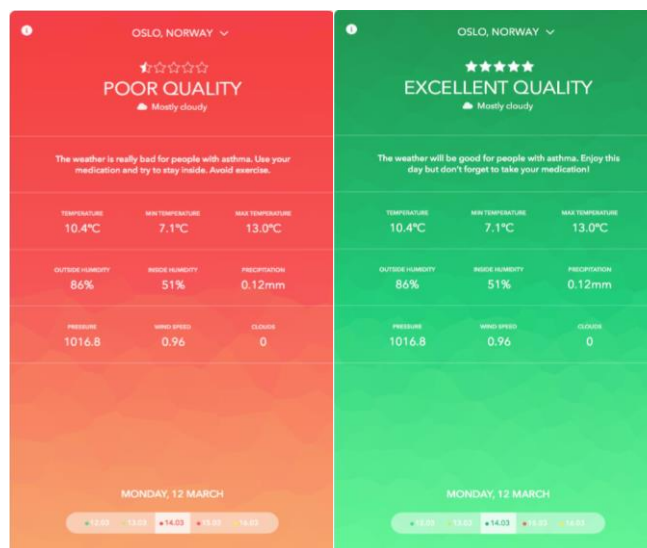


Figure 5. Weather forecast in *Asthmatic*

1.2.4 Asthma & Me

The last kind of the app is like Asthma & Me [10], which will help you test whether you have been caught asthma by a short video.

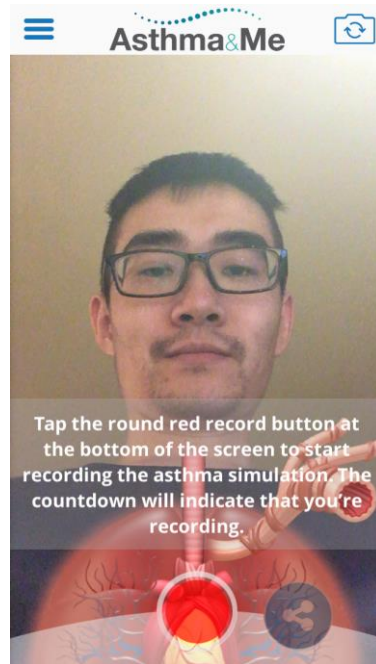


Figure 6. Create an educational short video of a diseased airway onto a patient's video selfie.

1.3 SMARTASTHMA

What I have done, SmartAsthma, is an iOS application for users who would like to track the data of their daily inhaler usage. There will be a Bluetooth cover like Propeller Health that being linked on the inhaler.

Table 1. Comparisons between SmartAsthma and other asthma applications

	AsthmaMD	Propeller Health	Asthma Manager	My Asthma Pal	Asthmatic	Asthma & Me	SmartAsthma
Free	YES	YES	YES	YES	NO	YES	YES
Additional Device	YES	YES	NO	NO	NO	NO	YES
Asthma Control Test	NO	YES	NO	YES	NO	NO	YES
Asthma Action Plan	YES	NO	YES	YES	NO	NO	NO
PDF Report	YES	NO	NO	YES	NO	NO	YES
Weather Forecast	NO	YES	NO	NO	YES	NO	NO

2.0 FRAMEWORK

In this section, the development environment and the constituent parts of the application will be discussed.

2.1 DEVELOPMENT ENVIRONMENT

The development environment is iOS platform, which is *nix based and comes directly from the development path of OS X. And the language is Swift 4.2, which is safe, fast and expressive [11].

2.1.1 iOS

Architecture is the highest level of a system design. So, knowing the architecture of iOS will help designing an application. The iOS is constructed by layers. The layers are shown below:

Table 2. Layers in the iOS Architecture

Cocoa Touch
Media
Core Services
Core OS

Cocoa Touch layer drives the UI, providing the controllers, widgets, etc. And this layer let us access to main system functions, such as Contacts, Camera and pushing notifications, etc. This layer offers high-level features in generating the interface (Controllers), handling pushing notification service and frameworks containing user interface elements, known as UIKit, as well as map interface, known as MapKit. Media layer has the Graphics libraries, Audio and Air play. This layer provides capabilities of graphics, audio and video. In this layer, developers can custom images or graphics being shown on screen or append audio and video into the app by interacting with the lower level hardware. Core Services gives access to fundamental resources needed for app. It provides the frameworks to detect location (Core Location), to access storage (Core Data) and more. Core OS processes the kernel operations. Bluetooth, USB and other accessories are in this layer [12] [13].

2.1.2 XCode

XCode is developed by Apple, which was selected to work on this project. Integrated development environment (IDE) is consist of editor, builder, compiler and debugger. The workspace could be divided into four areas: editor, debug, utility areas and navigator [14].

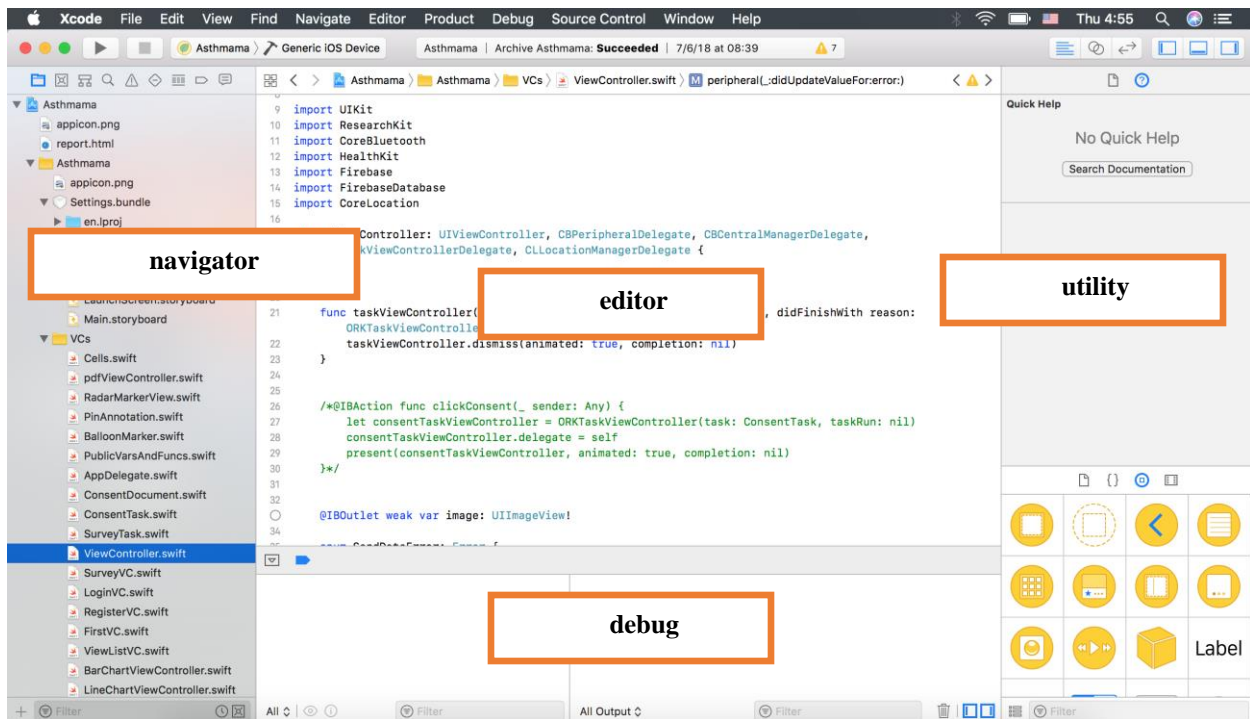


Figure 7. XCode workspace

Interface Builder (IB) is the workplace where developer designs the UI by dragging the elements into the view area. Never forget to connect the element to code area.

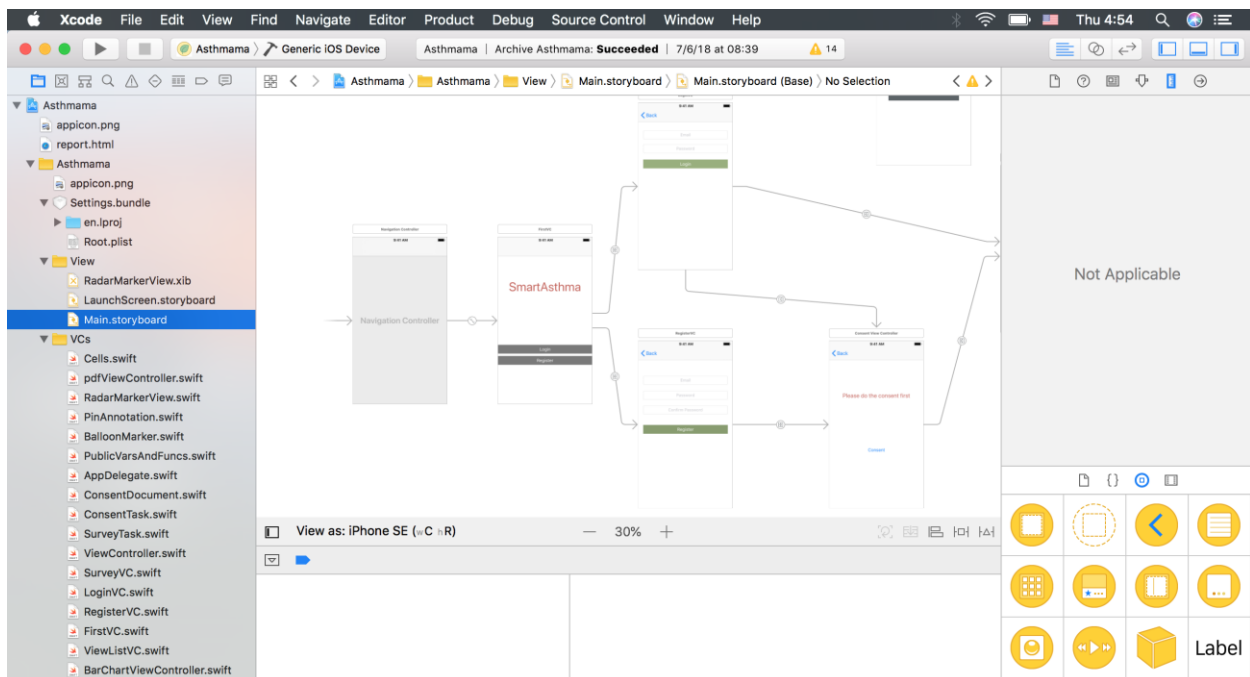


Figure 8. XCode Interface Builder area

2.1.3 CocoaPods

Developers can add third party libraries to their project via CocoaPods. Generate the Podfile and then insert the command line. The Podfile content of this project shows below.

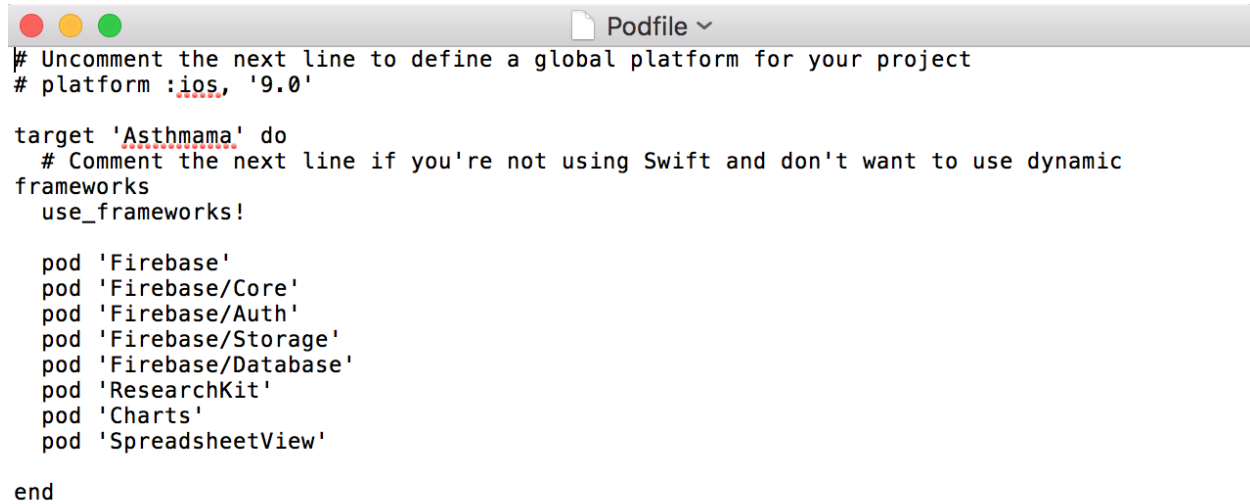


Figure 9. Content of Podfile

2.2 BLUETOOTH LOW ENERGY (BLE)

2.2.1 Bluetooth Features

Bluetooth does not rigidly match the OSI model. What should be really cared is how Bluetooth works and the different modes that BLE could work as. BLE can perfectly work within 100 meters (about 330 ft). The data rate is from 125 kbit/s to 2 Mbit/s over the air. There is a 6 ms latency when two devices connect with each other. [15]

2.2.2 Peripheral and Central

There are 2 types of devices: one is central while the other one is peripheral. Peripheral acts as the server that provides client with data. Central plays the client role that receives data.

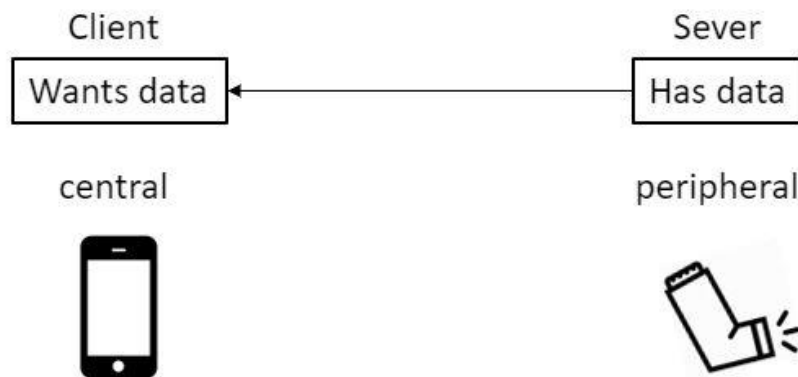


Figure 10. Peripheral and Central

There are 2 communication modes: advertising mode and connected mode. The former one is one-to-many mode, which means the peripheral will keep advertising all the time and the messages are available to all the centrals around. This is not safe and not allowed by Apple. The other mode is one-to-one mode, in which peripheral and central need to connect to each other by matching both uuid and device name and then start data transmission.

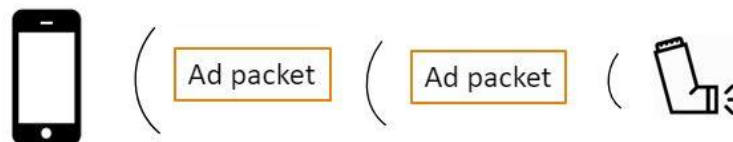


Figure 11. Advertising mode

2.2.3 Work Flow

In this project, functions in CBCentralManager Class of CoreBluetooth framework are used to set up the working cycle of Bluetooth. Firstly, check and make sure Bluetooth is turned on. Then, start scanning peripheral. After saving peripheral name and uuid, stop scan to save battery. Next, make connection between inhaler and mobile phone. In this progress, characteristics are the container of messages passed from sensors to SmartAsthma. Characteristics are checked continuously if they are updated. Figures bellow show the diagram of work flow and sample codes corresponding to it.

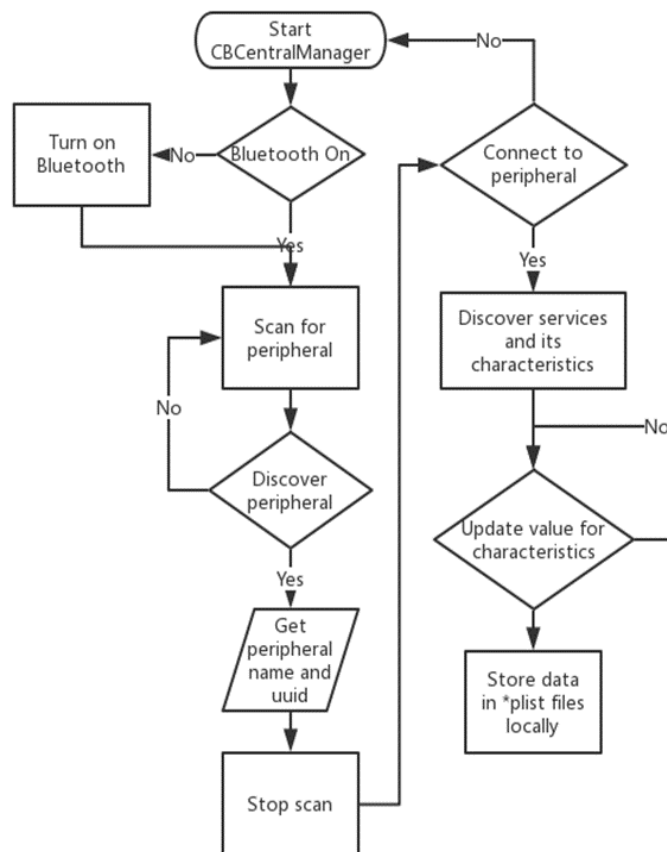


Figure 12. Demo running on laptop works as peripheral

```

/* get peripheral data */
//var ClickNumber = 0
func peripheral(_ peripheral: CBPeripheral, didUpdateValueFor characteristic: CBCharacteristic, error: Error?) {
    guard error == nil else {
        print("ERROR: \(#file, #function)")
        print(error!)
        return
    }

    if characteristic.uuid.uuidString == "C001" {
        let data = characteristic.value! as NSData
        DispatchQueue.main.async {
            let receiveStr = String(data: data as Data, encoding: .utf8)!
            let receiveAry = receiveStr.components(separatedBy: "\t")
            //print(receiveAry)
            let stampStr = receiveAry[0]
            let receiveLati = receiveAry[1]
            let receiveLong = receiveAry[2]
            let latiandlong = NSArray(objects: receiveLati, receiveLong)
            saveArray(array: latiandlong, fileName: "latiandlong.plist")
        }
    }
}

```

Figure 13. Sample codes of the function used to get data from peripheral

2.3 LIBRARIES AND THIRD-PARTY FRAMEWORKS

Some third-party libraries are added to this iOS project to save the workload and downsize the code. This section will discuss the architectures or the features of these frameworks.

2.3.1 ResearchKit

ResearchKit can easily help developers create a medical research running on iPhone, which is an open source framework provided by Apple. Developers can create the consent flows, customize survey types and questions and share the results with the community [16].

There are three reasons that choose ResearchKit. Firstly, it is an open source framework backed by Apple, and it is accessible to plenty of documentation for help. Secondly, it allows to create visual consent flows including the hand signature, and the modules of surveys are

customizable. Last but not the least, ResearchKit works seamlessly with HealthKit, it is possible to fetch the relevant data from the Health App.

In this project, Exacerbation Score Questionnaire is created by the modules in ResearchKit. Below shows some of the questionnaire screen shots.

SmartAsthma

Please complete the questionnaire to determine the risk of asthma exacerbation

Exacerbation Score Intro

Just answer a few simple questions to complete the exacerbation score

Get Started

How old is your child?

22

Next

We may not estimate the score if skipping

1 **2** **3**
ABC DEF

4 **5** **6**
GHI JKL MNO

7 **8** **9**
PQRS TUV WXYZ

0 **X**

ASTHMA SYMPTOMS
Select ALL that apply:

Symptoms present for 3 or more months of the year

Symptoms precipitated by colds ✓

Symptoms precipitated by cold air ✓

Symptoms precipitated by exercise

Symptoms precipitated by dust

Inhaled steroids:

Beclomethasone (Qvar)

Budesonide (Pulmicort)

Ciclesonide (Alvesco)

Fluticasone (Flovent, Arnuity Ellipta)

Mometasone (Asmanex) ✓

Triamcinolone (Azmacort)

PAST MEDICAL HISTORY
Select ALL that apply for your child with asthma:

You child has a history of eczema or allergic rhinitis (hay fever)

Mother or father have (or had) asthma, allergies, or hay fever ✓

Your child has been exposed to cigarette smoke (either as an infant or currently)

Figure 14. Screenshots of Exacerbation Score Questionnaire

2.3.2 UIKit

UIKit is a primary framework, which is responsible for API access to the elements in an app. It works as the infrastructure for iOS apps [13] [17]. This framework connects codes with interface as well as functions corresponding to the interaction with users.

2.3.3 MapKit

MapKit is also a framework developed and maintained by Apple. With MapKit, developers could embed and display a map or satellite interface into their own app. Custom annotations and overlays can be added to the map. In this project, a map was generated to show the most frequent locations where inhaler being used [18].

2.3.4 WebKit

Thanks to WebKit, developers can implement browser features and visit specific URL in their app. Also, it is the simplest way to use WebKit to open a pdf file. But here is a bug or something that the link will be crash if the URL has the beginning of “https”. URLs starting with “http” are fine [19].

2.3.5 Core Location

Another framework created by Apple, allowing users access to GPS to detect their location. Core Location provides service in current device location, latitude and longitude detection, and mutual

transformation of altitude and specific address [20]. In this project, Core Location works with MapKit to display the places where inhaler used the most frequently.

2.3.6 Charts

Charts for iOS is a third-party library to generate beautiful graphs. Charts supports Xcode 9.3 / Swift 4.1 and provides 8 different chart types including radar and combined chart [21].

The reason that choosing Charts to realize data visualization is that the framework is mainly updated by Swift 4. Comparing with Core Plot, which is developed by Objective-C, Charts has less crashes and more stability during running the project. Also, Charts provides more than 7 types of charts to display data.

2.3.7 Firebase

The Firebase Realtime Database is backed by Google. It is a cloud-hosted database, where data is stored as JSON tree. Unlike SQL database, there are no tables or records [22].

When adding data to the JSON tree, it becomes a node in the existing JSON structure with an associated key. Nesting data structure should be avoided because the flatter the structure is, the faster data would be fetched.

The reason to use Firebase is very simple. Firebase is backed by Google and adequate help is provided both by Google documentation and the third-party solution platform like Stack Overflow. It is easy and convenient to find tutorial videos online and even email Google the issue whatever it has. And Firebase Database is multi-platform supportable, which makes it possible that developing an Android or Web App of SmartAsthma application by using the same

database. Therefore, it is possible for users to share data between different platform such as an iPhone and an Android watch, and they do not need to worry about losing data if they have to change their mobile from iOS to an Android one.

3.0 SOFTWARE DESIGN

SmartAsthma is an iOS mHealth application that utilizes a Bluetooth cover attached on the inhaler to record daily usage data and inform the user with their current situation. With SmartAsthma, users can register their accounts, focus on their Asthma Action Plan, track their frequent inhaler used location, complete the questionnaire to see if their asthma controlled well, and email themselves their pdf reports.

3.1 DATA FLOW

3.1.1 Data Initialization

When creating a new account, if the email has already been registered, that email will not be used as the username any more.

After login, data will be synchronized from database and saved in *plist file locally.

Figure bellow shows the sample codes of initial function, in which DailyData will be used as data source to generate the bar chart.

```

public func initiate() {
    var ref : DatabaseReference!
    ref = Database.database().reference()
    ref.child("Test").child("DailyData").observeSingleEvent(of: .value, with: {(snapshot) in
        var timelogArray = [String]()
        var addressArray = [String]()
        for child in snapshot.children {
            let snap = child as! DataSnapshot
            let timelog = snap.key
            let valueDic = snap.value as! [String : AnyObject]
            let address = valueDic["Location"] as! String
            timelogArray.append(timelog)
            addressArray.append(timelog)
            addressArray.append(address)
        }
        //print(timelogArray as NSArray)
        //print(addressArray as NSArray)
        if existFile(fileName: "TimeLog.plist") == false {
            saveArray(array: timelogArray as NSArray, fileName: "TimeLog.plist")
            if existFile(fileName: "DailyData.plist") == false {
                saveArray(array: addressArray as NSArray, fileName: "DailyData.plist")
            }
        }

        //let test1 = readPlist(fileName: "TimeLog.plist")
        //print(test1)
    })
}

```

Figure 15. Sample codes of synchronizing data from database in function initiate()

3.1.2 Data Flow Diagram

Whenever SmartAsthma receives an action message from the peripheral, a timestamp will be generated. Then the timestamp and current coordinate will be saved in journal file. After conversion and calculation, data will be stored in different files shown in **Error! Reference source not found.** Below is the data flow from peripheral to local *.plist files and database.

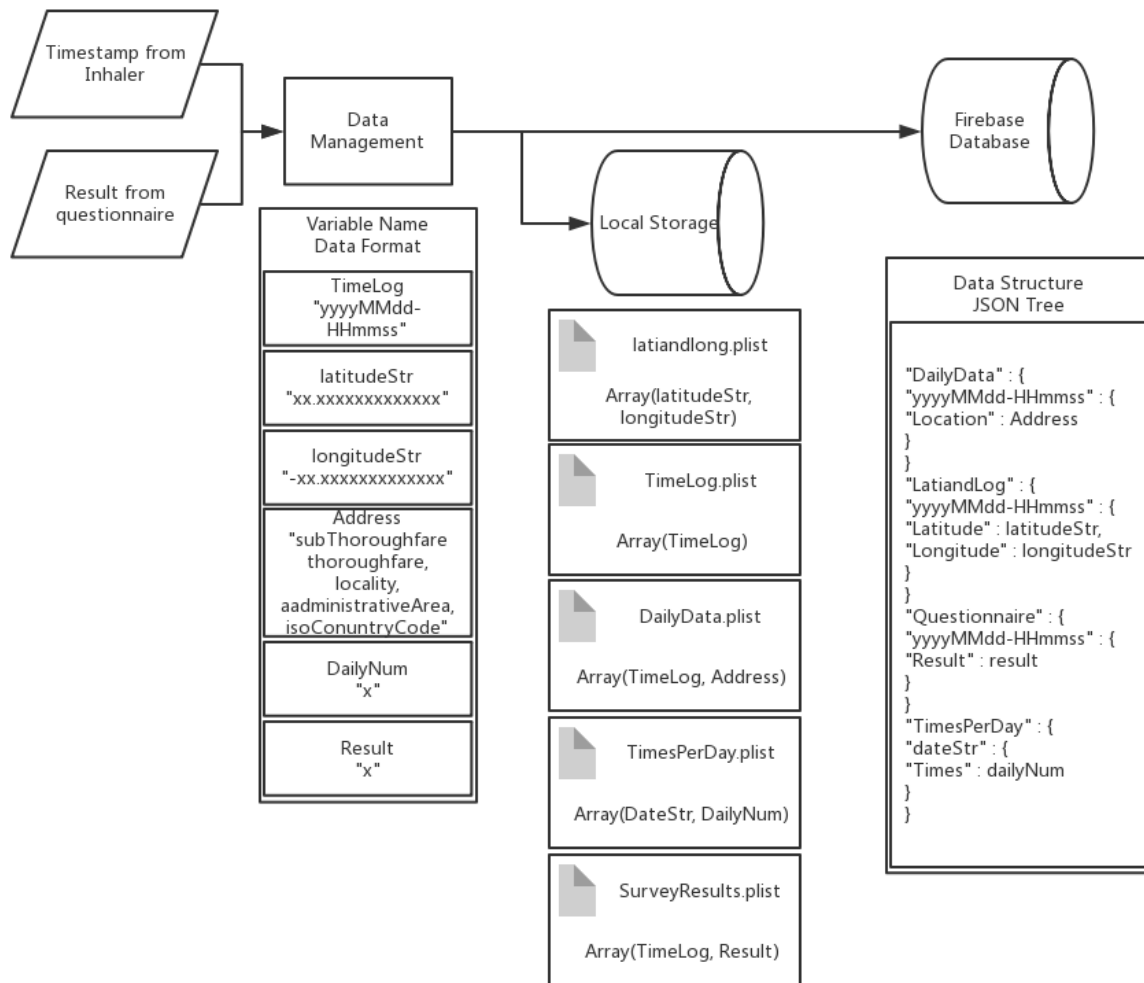


Figure 16. Data flow diagram

3.2 APPLICATION WALKTHROUGH

3.2.1 Login and Sign Up

Login is required if user kills the progress in background. Username is an email, which is used to send pdf report so that the email should not be fake. If the username has been registered, user can not register the same username any more.

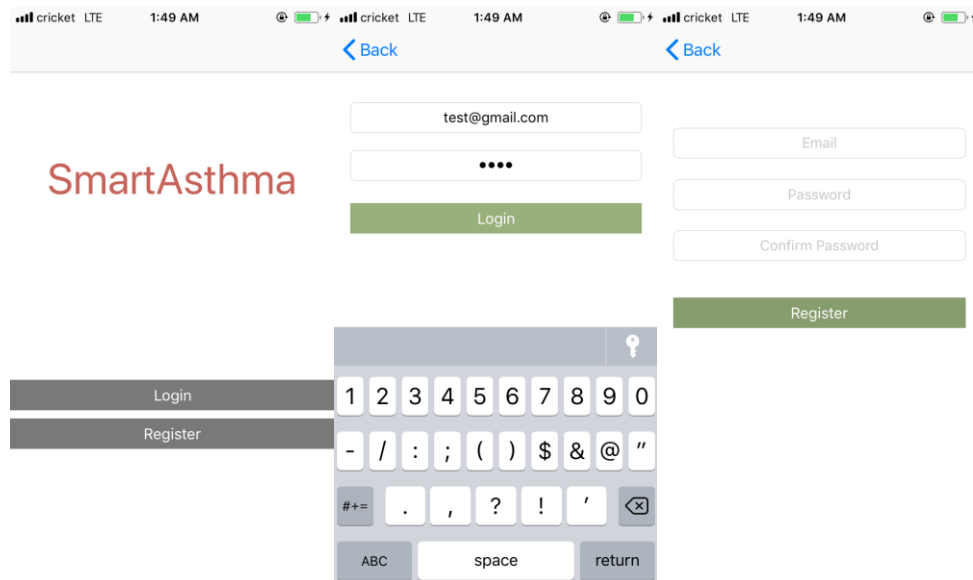


Figure 17. Login and register screens

3.2.2 Allergies and Asthma Videos

The icons in the main screen below will link user to the YouTube videos created by Children's Hospital of Pittsburgh of UPMC, in which user can learn more about Allergies and Asthma as

well as how to use inhaler. The last icon will lead user to the online resource of UPMC, where user can quickly find a doctor or explore more facts about asthma.

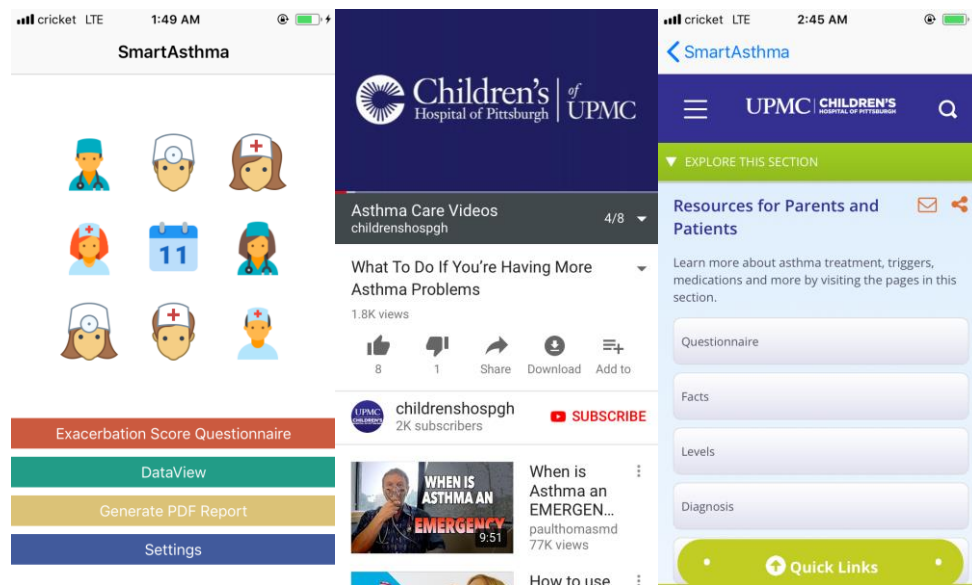


Figure 18. Main screens and icon links

3.2.3 Exacerbation Score Questionnaire

In Exacerbation Score Questionnaire, users will select their inhaled steroids, combination inhalers, asthma symptoms, medications, healthcare utilization and medical history to decide which zone (Red/Yellow/Green) they are in. After completing the questionnaire, user will be asked to add their action plan due to their current zone in the future.

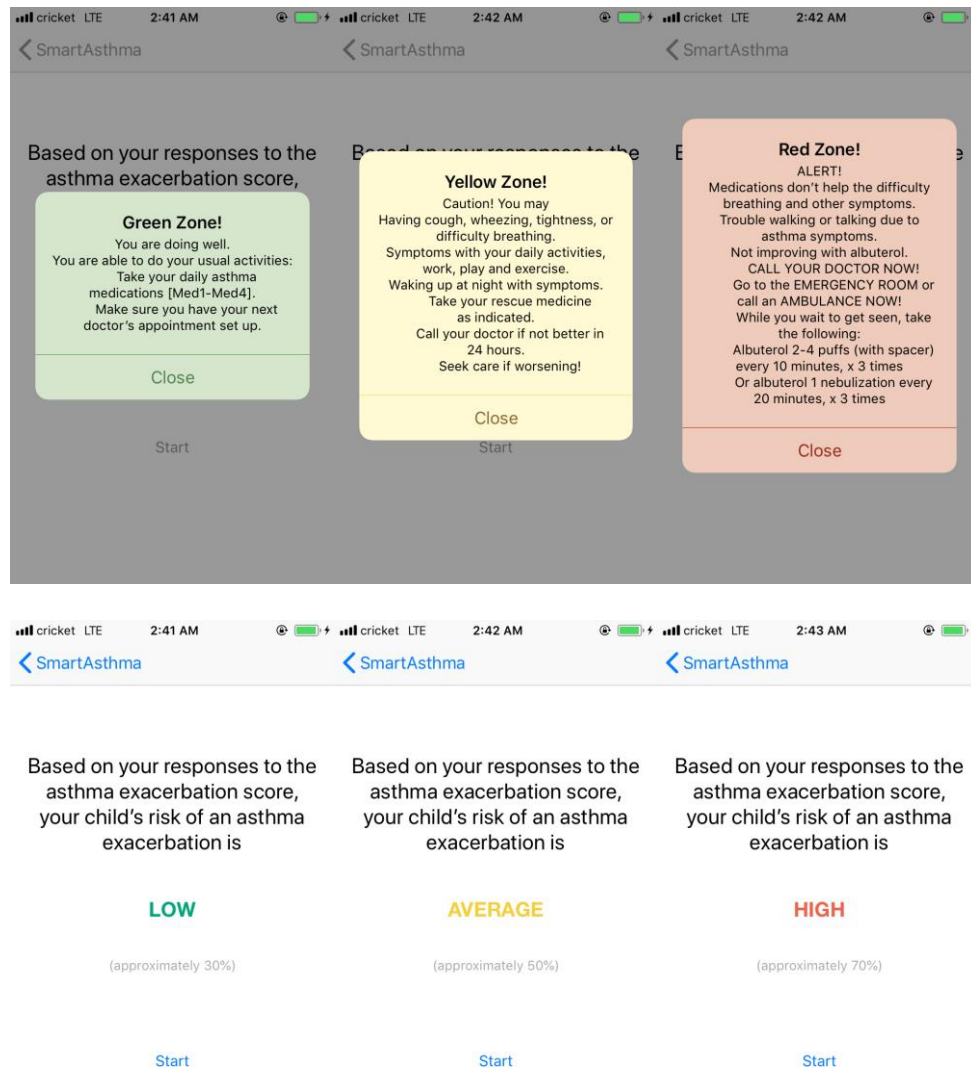


Figure 19. Asthma control zones and results

Figure below shows the sample codes that how a questionnaire step is created in ResearchKit and how to customize the questions and answer choices.

```

import Foundation
import ResearchKit

public var SurveyTask: ORKOrderedTask {

    var steps = [ORKStep]()

    let instructionStep = ORKInstructionStep(identifier: "IntroStep")
    instructionStep.title = "Exacerbation Score Intro"
    instructionStep.text = "Just answer a few simple questions to complete the exacerbation score"
    steps += [instructionStep]

```

Figure 20. Samples codes of creating a questionnaire step

3.2.4 Data Visualization

In this section, users can view their data as bar chart, line chart, radar chart, pie chart and map view.

3.2.4.1 Daily Usage Report

User can scroll back and forth to see data older and newer. Selecting the specific bar will lead user to a list of detailed inhaler usage.

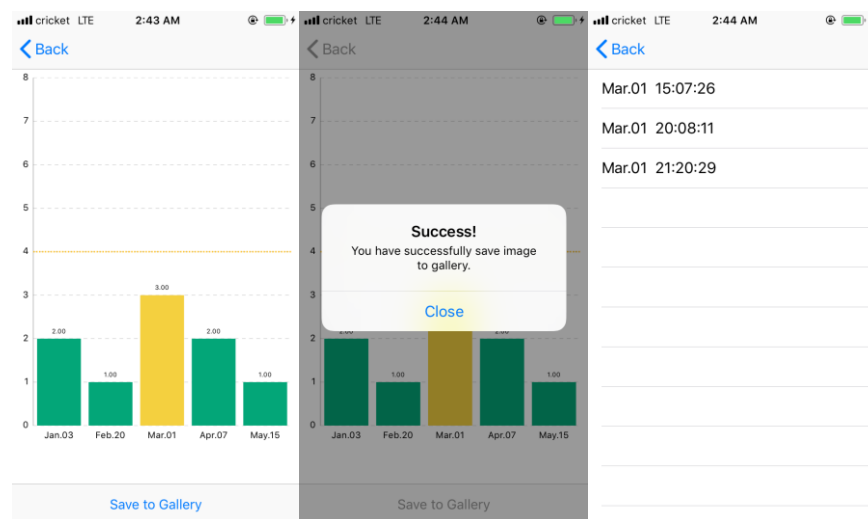


Figure 21. Daily Usage Report screenshots

Figures below shows the sample codes that how to match labels (Dates) and values (Number of inhaler used per day) together in function setChart() and how to pass the values when tap on the specific bar in function chartValueSelected().

```
func setChart(dataPoints: [String], values: [Double]) {
    chartView.noDataText = "No Data Right Now."

    var dataEntries: [BarChartDataEntry] = []

    for i in 0..

```

Figure 22. Sample codes of function setChart()

```
func chartValueSelected(_ chartView: ChartViewBase, entry: ChartDataEntry, highlight: Highlight) {

    let storyboard : UIStoryboard = UIStoryboard(name: "Main", bundle: nil)
    let vc : barPopUpVCViewController = storyboard.instantiateViewController(withIdentifier: "barPopUpVC")
    vc.a = datearray[Int(entry.x)]

    vc.b = String(entry.y)

    self.navigationController?.pushViewController(vc, animated: true)
}
```

Figure 23. Sample codes of function chartValuesSelected()

3.2.4.2 Most Frequent Locations

A pin will drop on the map each time when inhaler being used. And new action will be shown on the top of the location log. In the future, there will be more separated options in the log such as selecting by the frequency of use or displaying in selected month.

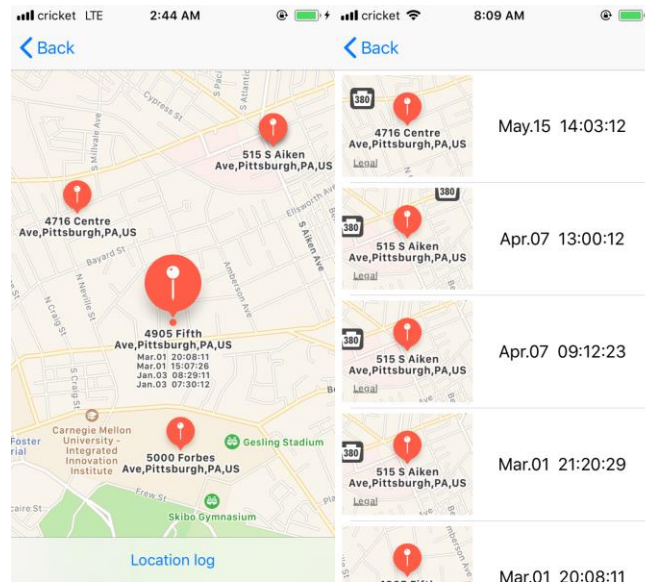


Figure 24. Locations and Log

3.2.4.3 Asthma Control Test Score

Besides the field of study, data quantity and quality as well as accurate data collection will be the foundation to convince others the value of the research [5]. In this app, it will record the location where users using inhaler and the number of times they use in a single day. After user using inhaler, a message will be advertised via Bluetooth and user's phone will record the message. All these data as well as the questionnaire history are visualized in Data View section. To speak of ACT score, it is based on the daily usage of inhaler and the questions in ACT are converted and reconstructed.

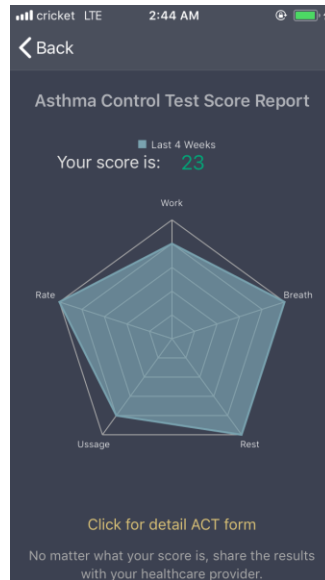


Figure 25. Asthma Control Test Score Report screenshot

Table 3. Comparison between original questions and new specific method

Score	1	2	3	4	5
xVal in Radar Chart	20	40	60	80	100
<p>In the past 4 weeks, how much of the time did your asthma keep you from getting as much done at work, school or at home?</p> <p>↓</p> <p>How many days was inhaler used during last 4 weeks</p>	<p>All the time</p> <p>↓</p> <p>more than 28</p>	<p>Most of the time</p> <p>↓</p> <p>19~28</p>	<p>Some of the time</p> <p>↓</p> <p>10~18</p>	<p>A little of the time</p> <p>↓</p> <p>1~9</p>	<p>None of the time</p> <p>↓</p> <p>0</p>

Table 3. (continued)

During the past 4 weeks, how often have you had shortness of breath? ↓ Total amount of inhaler used in last 4 weeks	More than once a day ↓ more than 28	Once a day ↓ 25~28	3 to 6 times a week ↓ 12~27	Once or twice a week ↓ 4~11	Not at all ↓ 0
During the past 4 weeks, how often did your asthma symptoms (wheezing, coughing, shortness of breath, chest tightness or pain) wake you up at night or earlier than usual in the morning? ↓ Count of hour string less than 6	4 or more nights a week ↓ more than 14	2 to 3 nights a week ↓ 6~14	Once a week ↓ 3~5	Once or twice ↓ 1~2	Not at all ↓ 0

Table 3. (continued)

During the past 4 weeks, how often have you used your rescue inhaler or nebulizer medication (such as albuterol)? ↓ Total amount of inhaler used in last 4 weeks	3 or more times per day ↓ more than 70	1 to 2 times per day ↓ 24~70	2 or 3 times per week ↓ 8~23	Once a week or less ↓ 1~7	Not at all ↓ 0
How would you rate your asthma control during the past 4 weeks? ↓ How many days was inhaler used during last 4 weeks	Not controlled at all ↓ more than 28	Poorly controlled ↓ 20~28	Somewhat controlled ↓ 16~19	Well controlled ↓ 6~15	Completely Controlled ↓ 0~5

3.2.4.4 Exacerbation Score History

The questionnaire is based on the prior studies, and these estimates are not meant to predict the exact possibility that asthma of user got exacerbation. Figure bellow shows the data visualization of the score history and suggestions given based on current questionnaire result.

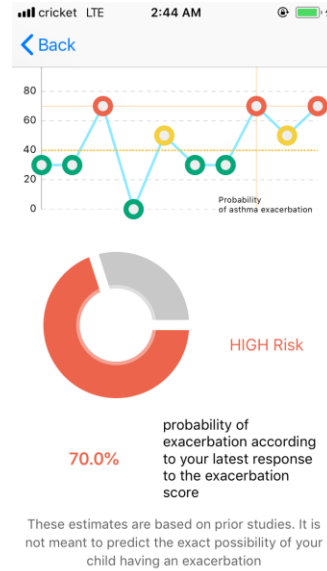


Figure 26. Current exacerbation score and its history.

3.2.5 Report Generation

The PDF report will contain the daily usage of inhaler, locations where inhaler being used and the questionnaire results history. User will get an email with the report attached.

There are generally three ways to create a pdf file: drawing rectangles and images by Core Graphics framework and Quadz2D, using PDFKit recently released by Apple, and generate pdf from the HTML template by using function `UIMarkupTextPrintFormatter()`.

Drawing a pdf will cost a lot of time. The margins and layout details should be considered in advanced. It is started as an empty space like a painter need to make himself a paper before creating a picture.

PDFKit is a framework created by Apple. It is convenient to call some functions to start a new page, convert images to pdf, etc. However, PDFKit is mainly used to provide developers a new way to open an existing pdf file and have more choices to handle it such as adding widgets. When embedding an image to a pdf file, the image occupies the whole page. When adding

paragraphs, it happens again like drawing the rectangles that margins and positions need to be considered.

Then, a HTML template is used in this project. Some place holders are used in the template and then they are replaced by the real pictures and data, which is shown in the figure of sample codes bellow. What need to be mentioned here is that when using `UIPrintPageRender` in Swift 4, the images will disappear if directly replacing URLs in the `` tag. Solution is that the image will be shown correctly after adding “file:///” before the URL.

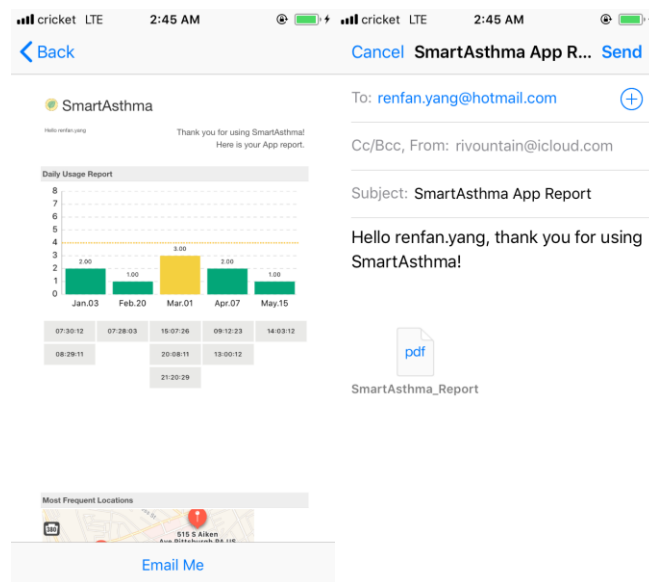


Figure 27. Report shown in Web View and email sending screen

```
var HTMLContent = try String(contentsOfFile: pathToInvoiceHTMLTemplate!)

HTMLContent = HTMLContent.replacingOccurrences(of: "#USER_NAME#", with: User_name)

HTMLContent = HTMLContent.replacingOccurrences(of: "#APP_IMAGE#", with: appImageURL.path)
```

Figure 28. Replace the place holder with real data

After generating the pdf report, users can send it to themselves via email. Figure bellow shows the codes of calling the mailing function in MessageKit framework.

```

@IBAction func emailClick(_ sender: Any) {

    reportComposer.exportHTMLContentToPDF(HTMLContent: HTMLContent)

    if MFMailComposeViewController.canSendMail() {

        let mailComposeViewController = MFMailComposeViewController()
        mailComposeViewController.mailComposeDelegate = self
        mailComposeViewController.setSubject("SmartAsthma App Report")

        mailComposeViewController.addAttachmentData(NSData(contentsOfFile: reportComp
        mailComposeViewController.setMessageBody("Hello \(User_name), thank you for us
        mailComposeViewController.setToRecipients([UserMail])

        self.present(mailComposeViewController, animated: true, completion: nil)
    }
}

```

Figure 29. Sample codes of setting up mailing details

3.2.6 Running at Background

An easier way than coding in the project is that check the background options shown in the figure below in the target project.

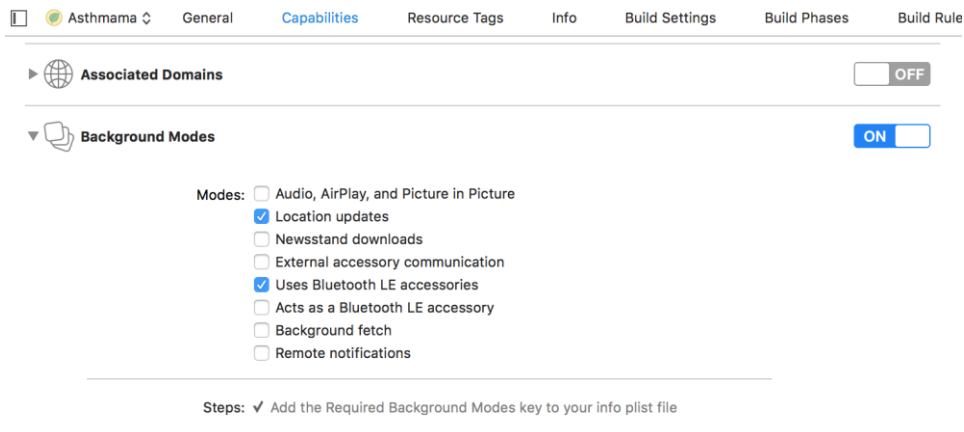


Figure 30. Background modes settings

Also, this can be achieved by coding in the delegate file, however, developers need to set the time of running in background (10 minutes at most allowed by Apple) again and again. While turning on background modes in capabilities will not suffer the same issue.

3.3 DATABASE

Firebase Database is a cloud-hosted database where data is stored in JSON tree and is synchronized each time when new data entering.

3.3.1 Data Structure

To avoid nesting data, which means data denormalization that the data is instead split into different paths, a flattened structure shows bellow. In this way, data can be efficiently downloaded in separate calls as it is needed [23] [24].

```
"DailyData" : {
  "20180103-073012" : {
    "Location" : "4905 Fifth Ave, Pittsburgh, PA, US"
  }
}
"LatandLong" : {
  "20180103-073012" : {
    "Latitude" : "40.4479845000000",
    "Longitude" : "-79.9434691999999"
  }
}
"Questionnaire" : {
  "20180103-080701" : {
    "Result" : 30
  }
}
"TimesPerDay" : {
  "20180103" : {
    "Times" : 2
  }
}
```

Figure 31. Flattened data structure example

4.0 TESTFLIGHT

TestFlight simplifies the way to invite users to test SmartAsthma and get feedback before releasing it on the App Store [25].

There are mainly two tester groups: internal testers (now called App Store Connect Users) and external testers. As for internal testers, they are the users with the role as Admin, App Manager, Legal, Developer, or Marketer to the application. It has a limit number that the developer can add 25 testers in this group at most. When it comes to external testers, they have no access to the iTunes Connect and they can only download and install the app. Developers can add at most 10,000 testers in this group [26].

iOS Builds

The following builds are available to test. [Learn more about build status and metrics.](#)

Version 1.0





Build	App Store Connect Users ?	External Testers ?	Invitations ?	Installations ?	Last 7 Days ?	Crashes ?
 1.0.6	● Testing Expires in 76 days	● Testing Expires in 76 days	10	3		
 1.0.5	● Testing Expires in 70 days	● Testing Expires in 70 days	10	2		
 1.0.4	● Testing Expires in 67 days	● Testing Expires in 67 days	10	3		
 1.0.3	● Testing Expires in 45 days	● Testing Expires in 45 days	10	4		

Figure 32. iOS Builds on TestFlight

5.0 CONCLUTIONS AND FUTURE WORK

Smartphones are more and more useful in our daily life, not only by its social position and function but also by its role of personal data collecting center. Nowadays, we trust data more than we believe in ourselves since we may be constrained by our own personalities and emotions. Therefore, SmartAsthma is designed for inhaler users who demand for accessing to their daily data the goal of SmartAsthma is to reduce the risk of severe asthma attacks.

5.1 CONCLUTIONS

With SmartAsthma, users can access to the video stuff that spreading asthma knowledge, complete the questionnaire to see if they have got exacerbation, to gather the inhaler daily usage data including the usage times per day and its places, and get the Asthma Control Test Score without answering questions or calculation, etc. Users can have all their data shown in a pdf report, which can be emailed to themselves or to their doctors selected by users. SmartAsthma cannot make decisions on asthma treatment but it can give suggestions based on prior asthma studies.

5.2 FUTURE WORK

5.2.1 Asthma Game

For the purpose of drawing children's attention and making them know more about asthma, a word game like the figure below will be added. The game not only test the vocabulary of asthma and allergies, but subtly tell what can make asthma worse.

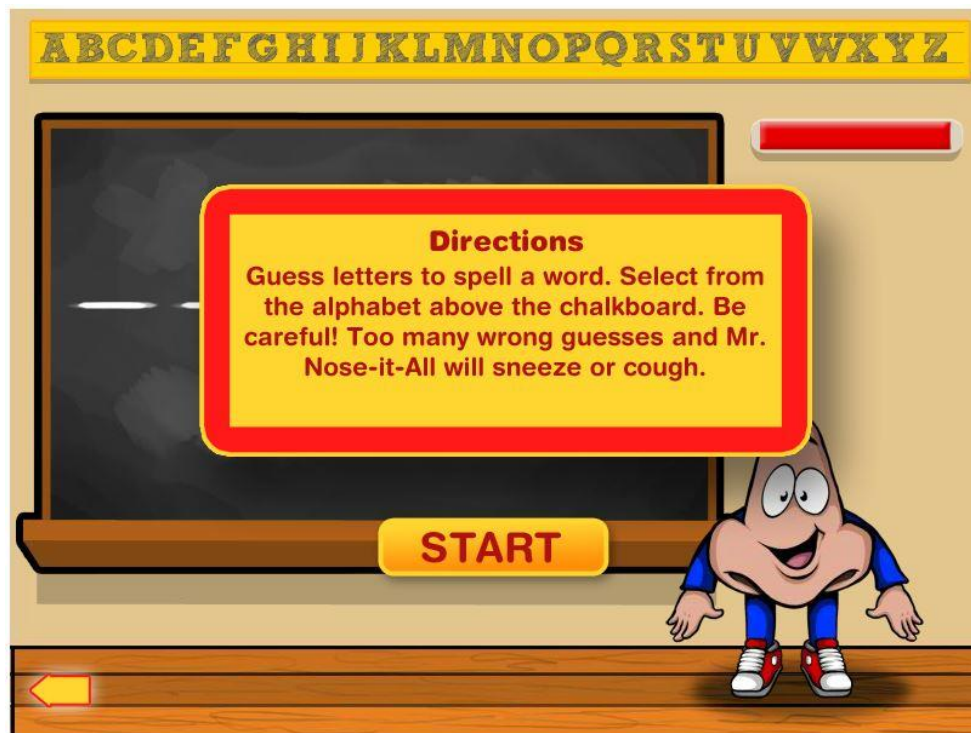


Figure 33. MR. NOSE-IT-ALL'S WORD GAME, from [27]

5.2.2 Asthma Action Plan

After completing the questionnaire, users are supposed to add their plan to the specific zone based on figure below. In this way, users can check if they do well in their asthma control.

Asthma Action Plan

For: _____ Doctor: _____ Date: _____
 Doctor's Phone Number _____ Hospital/Emergency Department Phone Number _____

GREEN ZONE

Doing Well

- No cough, wheeze, chest tightness, or shortness of breath during the day or night
- Can do usual activities

And, if a peak flow meter is used,

Peak flow: more than _____
(80 percent or more of my best peak flow)

My best peak flow is: _____

Medicine	How much to take	When to take it
_____	➡	➡

Before exercise ☐ _____ ☐ 2 or ☐ 4 puffs _____ 5 minutes before exercise

YELLOW ZONE

Asthma Is Getting Worse

- Cough, wheeze, chest tightness, or shortness of breath, or
- Waking at night due to asthma, or
- Can do some, but not all, usual activities

-Or-

Peak flow: _____ to _____
(50 to 79 percent of my best peak flow)

First

Add: quick-relief medicine—and keep taking your GREEN ZONE medicine.

_____ ☐ 2 or ☐ 4 puffs, every 20 minutes for up to 1 hour
(short-acting beta₂-agonist) ☐ Nebulizer, once

Second

If your symptoms (and peak flow, if used) return to GREEN ZONE after 1 hour of above treatment:

☐ Continue monitoring to be sure you stay in the green zone.

-Or-

If your symptoms (and peak flow, if used) do not return to GREEN ZONE after 1 hour of above treatment:

☐ Take: _____ ☐ 2 or ☐ 4 puffs or ☐ Nebulizer
(short-acting beta₂-agonist)

☐ Add: _____ mg per day For _____ (3–10) days
(oral steroid)

☐ Call the doctor ☐ before/ ☐ within _____ hours after taking the oral steroid.

RED ZONE

Medical Alert!

- Very short of breath, or
- Quick-relief medicines have not helped, or
- Cannot do usual activities, or
- Symptoms are same or get worse after 24 hours in Yellow Zone

-Or-

Peak flow: less than _____
(50 percent of my best peak flow)

Take this medicine:

☐ _____ ☐ 4 or ☐ 6 puffs or ☐ Nebulizer
(short-acting beta₂-agonist)

☐ _____ mg
(oral steroid)

Then call your doctor NOW. Go to the hospital or call an ambulance if:

- You are still in the red zone after 15 minutes AND
- You have not reached your doctor.

DANGER SIGNS

- Trouble walking and talking due to shortness of breath
- Lips or fingernails are blue

➡

Take ☐ 4 or ☐ 6 puffs of your quick-relief medicine AND

Go to the hospital or call for an ambulance _____ NOW!
(phone)

Figure 34. Asthma Action Plan details, from [28]

5.2.3 Better Interface and Interaction Options

In the future, there will be more separated options in the log such as selecting by the frequency of use or displaying in selected month. And the accuracy will be considered like more accurate pins will show up after scaling the map.

More specific functionality will be noted under the icon in the main screen. And the icon will have some change after being tapped.

APPENDIX A

HTML TEMPLATE

```
<!DOCTYPE html>

<html>

<head>

  <meta content="text/html; charset=utf-8" http-equiv="content-type">

  <title>#DATE#</title>

</head>

<body>

  <div class="invoice-box">

    <table cellpadding="0" cellspacing="0">

      <tbody>

        <tr class="top">

          <td colspan="2">

            <table>

              <tbody>

                <tr>
```

```

        <td class="title">
             SmartAsthma
        </td>
    </tr>
</tbody>
</table>
</td>
</tr>
<tr class="information">
    <td colspan="2">
        <table>
            <tbody>
                <tr>
                    <td>Hello #USER_NAME#</td>
                    <td>Thank you for using SmartAsthma!
                        <br>Here is your App report.
                    </td>
                </tr>
            </tbody>
        </table>
    </td>
</tr>
<tr class="heading">

```

```

        <td>Daily Usage Report</td>

</tr>

<tr class="details">

    <td></td>

</tr>

<tr class="details">

    <td></td>

</tr>

<tr class="heading">

    <td> Most Frequent Locations </td>

</tr>

<tr class="details">

    <td></td>

</tr>

<tr class="heading">

    <td> Exacerbation Score </td>

</tr>

<tr class="details">

    <td colspan="2">

        <table>

            <tbody>

                <tr>

                    <td></td>

```

<td> These estimates are based on prior studies.

It is not meant to predict the exact possibility of your child having an
exacerbation

</td>

</tr>

</tbody>

</table>

</td>

</tr>

<tr class="heading">

<td> Asthma Control Test Score </td>

</tr>

<tr class="details">

<td colspan="2">

<table>

<tbody>

<tr>

<td></td>

<td> Your ACT score is #ACT_SCORE#

 No matter what your score is, share the results with your healthcare provider.

</td>

</tr>

```
</tbody>
</table>
</td>
</tr>
</tbody>
</table>
</div>
</body>
</html>
```

APPENDIX B

SOME CODES IN SWIFT

B.1 PUBLIC VARIABLES AND FUNCTIONS

```
import UIKit

import Foundation

import CoreLocation

import Firebase

public var mamaKid : [Int : String] = [1 : "DailyData", 2 : "TimesPerDay"]

public var userName : String!

public var userMail : String!

public var stampStr : String!

//public var tempTimeStr : String!

public var tempDailyNum : Int!

public var barimage : UIImage!

public var lineimage : UIImage!

public var mapimage : UIImage!
```

```

/* data save and read */

// if file not exist, write array to plist; if file exist, read array from plist to tempArray and then
write new array to tempArray and finally write tempArray to plist

public func saveArray(array : NSArray, fileName : String) {

    let manager = FileManager.default

    let filePath : String = NSHomeDirectory() + "/Documents/\(userName)/\(fileName)"

    let exist = manager.fileExists(atPath: filePath)

    if !exist {

        array.write(toFile: filePath, atomically: true)

    } else {

        var tempArray = readPlist(fileName: fileName)

        if tempArray.count == 1 {

            let superTempArray = [[]] as NSArray

            superTempArray.addingObjects(from: tempArray as! [Any])

            superTempArray.write(toFile: filePath, atomically: true)

        }

        else {

            tempArray = tempArray.addingObjects(from: array as! [Any]) as NSArray

            tempArray.write(toFile: filePath, atomically: true)

        }

    }

}

public func saveArray2(array : NSArray, fileName : String) {

```

```

let manager = FileManager.default

let filePath : String = NSHomeDirectory() + "/Documents/\(userName)/\(fileName)"

let exist = manager.fileExists(atPath: filePath)

if !exist {

    array.write(toFile: filePath, atomically: true)

} else {

    var tempArray = readPlist(fileName: fileName)

    tempArray = tempArray.addingObjects(from: array as! [Any]) as NSArray

    tempArray.write(toFile: filePath, atomically: true)

}

}

public func readPlist(fileName : String) -> NSArray {

    var tempArray : NSArray?

    let manager = FileManager.default

    let filePath : String = NSHomeDirectory() + "/Documents/\(userName)/\(fileName)"

    let exist = manager.fileExists(atPath: filePath)

    if exist {

        tempArray = NSArray(contentsOfFile: filePath)!

    } else {

        print("No Such File \(fileName) Exist")

    }

    return tempArray!

}

```



```

/* get current date */

public func getDate() -> String {

    let now = Date()

    let dateFormatter = DateFormatter()

    dateFormatter.dateFormat = "yyyyMMdd-HH:mm:ss"

    let convertedDate = dateFormatter.string(from: now)

    return convertedDate

}

public func existFile(fileName : String) -> Bool {

    let manager = FileManager.default

    let filePath : String = NSHomeDirectory() + "/Documents/^(userName)/^(fileName)"

    let exist = manager.fileExists(atPath: filePath)

    if exist {

        return true

    } else {

        return false

    }

}

public func getPath(imageName: String) -> URL {

    let fileMgr = FileManager.default

    let dirPath = fileMgr.urls(for: .documentDirectory, in: .userDomainMask)[0]

    let writeURL = dirPath.appendingPathComponent(imageName)

    return writeURL

```

```

}

public func deleteFile(fileName : String) {

    let manager = FileManager.default

    let filePath : String = NSHomeDirectory() + "/Documents/(userName)/(fileName)"

    try! manager.removeItem(atPath: filePath)

}

public func initiate() {

    var ref : DatabaseReference!

    ref = Database.database().reference()

    ref.child("Test").child("DailyData").observeSingleEvent(of: .value, with: {(snapshot) in

        var timelogArray = [String]()

        var addressArray = [String]()

        for child in snapshot.children {

            let snap = child as! DataSnapshot

            let timelog = snap.key

            let valueDic = snap.value as! [String : AnyObject]

            let address = valueDic["Location"] as! String

            timelogArray.append(timelog)

            addressArray.append(timelog)

            addressArray.append(address)

        }

        //print(timelogArray as NSArray)

        //print(addressArray as NSArray)
    })
}

```

```

if existFile(fileName: "TimeLog.plist") == false {

    saveArray(array: timelogArray as NSArray, fileName: "TimeLog.plist")

    if existFile(fileName: "DailyData.plist") == false {

        saveArray(array: addressArray as NSArray, fileName: "DailyData.plist")

    }

}

//let test1 = readPlist(fileName: "TimeLog.plist")

//print(test1)

})

ref.child("Test").child("Survey").observeSingleEvent(of: .value, with: {(snapshot) in

    var resultArray = [AnyObject]()

    for child in snapshot.children {

        let snap = child as! DataSnapshot

        let timelog = snap.key

        let valueDic = snap.value as! [String : AnyObject]

        let result = valueDic["AsthmaChance"] as! Int

        //print(timelog)

        //print(result)

        resultArray.append(timelog as String as AnyObject)

        resultArray.append(result as Int as AnyObject)

    }

    //print(resultArray)

    if existFile(fileName: "SurveyResults.plist") == false {

```

```

        saveArray(array: resultArray as NSArray, fileName: "SurveyResults.plist")
    }

    //let test2 = readPlist(fileName: "SurveyResults.plist")

    //print(test2)
})

ref.child("Test").child("TimesPerDay").observeSingleEvent(of: .value, with: {(snapshot) in

    var timesperdayArray = [AnyObject]()

    var dailydatadateArray = [String]()

    for child in snapshot.children {

        let snap = child as! DataSnapshot

        let dateStr = snap.key as String

        let valueDic = snap.value as! [String : AnyObject]

        let dailyNum = valueDic["Times"] as! Int

        //print(dateStr)

        //print(dailyNum)

        dailydatadateArray.append(dateStr as String)

        timesperdayArray.append(dateStr as String as AnyObject)

        timesperdayArray.append(dailyNum as Int as AnyObject)

    }

    if existFile(fileName: "TimesPerDay.plist") == false {

        saveArray(array: timesperdayArray as NSArray, fileName: "TimesPerDay.plist")

    }

    if existFile(fileName: "DailyDataDate.plist") == false {

```

```

        saveArray(array: dailydatadateArray as NSArray, fileName: "DailyDataDate.plist")
    }

    //let test3 = readPlist(fileName: "TimesPerDay.plist")

    //print(test3)
})

ref.child("Test").child("LatiandLong").observeSingleEvent(of: .value, with: {(snapshot) in

    var latiandlongArray = [String]()

    for child in snapshot.children {

        let snap = child as! DataSnapshot

        let valueDic = snap.value as! [String : String]

        let latiStr = valueDic["Latitude"]

        let longStr = valueDic["Longitude"]

        latiandlongArray.append(latiStr!)

        latiandlongArray.append(longStr!)

    }

    if existFile(fileName: "latiandlong.plist") == false {

        saveArray(array: latiandlongArray as NSArray, fileName: "latiandlong.plist")

    }

    //let test4 = readPlist(fileName: "latiandlong.plist")

    //print(test4)

})
}

```

B.2 BLUETOOTH FUNCTION

```
class ViewController: UIViewController, CBPeripheralDelegate, CBCentralManagerDelegate,
ORKTaskViewControllerDelegate, CLLocationManagerDelegate {

    func taskViewController(_ taskViewController: ORKTaskViewController, didFinishWith
reason: ORKTaskViewControllerFinishReason, error: Error?) {

        taskViewController.dismiss(animated: true, completion: nil)

    }

    @IBOutlet weak var image: UIImageView!

    enum SendDataError: Error {

        case CharacteristicNotFound

    }

    var centralManager: CBCentralManager!

    // globle variable

    var connectPeripheral: CBPeripheral!

    var charDictionary = [String: CBCharacteristic]()

    override func viewDidLoad() {

        super.viewDidLoad()

        self.navigationItem.setHidesBackButton(true, animated: true)

        // Do any additional setup after loading the view, typically from a nib.

        let queue = DispatchQueue.global()

        // #1 method

        centralManager = CBCentralManager(delegate: self, queue: queue)

    }
```

```

/* #1 method */

func centralManagerDidUpdateState(_ central: CBCentralManager) {

    // if bluetooth open

    guard central.state == .poweredOn else {

        return

    }

    centralManager.scanForPeripherals(withServices: nil, options: nil)

}

/* #2 method */

func centralManager(_ central: CBCentralManager, didDiscover peripheral: CBPeripheral,
advertisementData: [String : Any], rssi RSSI: NSNumber) {

    print("find Bluetooth device: \(String(describing: peripheral.name))")

    guard peripheral.name != nil else {

        return

    }

    guard peripheral.name?.range(of: "Mantou") != nil else {

        return

    }

    central.stopScan()

    // store UUID

    let user = UserDefaults.standard

    user.set(peripheral.identifier.uuidString, forKey: "KEY_PERIPHERAL_UUID")

    user.synchronize()

```

```

connectPeripheral = peripheral

connectPeripheral.delegate = self

// #3 method

centralManager.connect(connectPeripheral, options: nil)
}

/* #3 method */

func centralManager(_ central: CBCentralManager, didConnect peripheral: CBPeripheral) {

    charDictionary = [:]

    // #4 method

    peripheral.discoverServices(nil)

}

/* #4 method */

func peripheral(_ peripheral: CBPeripheral, didDiscoverServices error: Error?) {

    guard error == nil else {

        print("ERROR: \(#file, #function)")

        return

    }

    for service in peripheral.services! {

        // #5 method

        connectPeripheral.discoverCharacteristics(nil, for: service)

    }

}

/* #5 method */

```



```

func peripheral(_ peripheral: CBPeripheral, didDiscoverCharacteristicsFor service:
CIService, error: Error?) {
    guard error == nil else {
        print("ERROR: \(#file, #function)")
        return
    }
    for characteristic in service.characteristics! {
        let uuidString = characteristic.uuid.uuidString
        charDictionary[uuidString] = characteristic
        print("find: \(uuidString)")
        peripheral.setNotifyValue(true, for: characteristic)
    }
}

/* get peripheral data */

func peripheral(_ peripheral: CBPeripheral, didUpdateValueFor characteristic:
CBCharacteristic, error: Error?) {
    guard error == nil else {
        print("ERROR: \(#file, #function)")
        print(error!)
        return
    }
    if characteristic.uuid.uuidString == "C001" {
        let data = characteristic.value! as NSData

```

```

DispatchQueue.main.async {

    let receiveStr = String(data: data as Data, encoding: .utf8)!

    let receiveAry = receiveStr.components(separatedBy: "\\t")

    let stampStr = receiveAry[0]

    let receiveLati = receiveAry[1]

    let receiveLong = receiveAry[2]

    let latiandlong = NSArray(objects: receiveLati, receiveLong)

    saveArray(array: latiandlong, fileName: "latiandlong.plist")

    // stamp to date and time

    let IntStamp = Int(stampStr)

    let timeInterval : TimeInterval = TimeInterval(IntStamp!)

    let time = NSDate(timeIntervalSince1970: timeInterval)

    let dformatter = DateFormatter()

    dformatter.dateFormat = "yyyyMMdd-HH:mm:ss"

    dformatter.locale = Locale.current

    let timeLog = dformatter.string(from: time as Date)

    let timeAry = timeLog.components(separatedBy: "-")

    let dateStr = timeAry[0]

    saveArray2(array: [timeLog], fileName: "TimeLog.plist")

    // latitude and longitude to address

    let    currentLoc    =    CLLocation(latitude:    Double(receiveLati)!,    longitude:
Double(receiveLong)!)

```

```

        CLGeocoder().reverseGeocodeLocation(currentLoc, completionHandler:
{(placemarks, error) -> Void in

    if error != nil {

        print("Failed")

        return

    }

    if (placemarks?.count)! > 0 {

        let pm = placemarks![0] as CLPlacemark?

        let address = (pm?.subThoroughfare)! + " " + (pm?.thoroughfare)! + "," +
(pm?.locality)! + "," + (pm?.administrativeArea)! + "," + (pm?.isoCountryCode)!

        // Save to local file

        // save DailyData.plist

        let dailyData = NSArray(objects: timeLog, address)

        saveArray(array: dailyData, fileName: "DailyData.plist")


        //let testDailyData = readPlist(fileName: "DailyData.plist")

        // save to DailyDataDate.plist for counting timesPerDay

        let dailyDataDate = [dateStr] as NSArray

        if existFile(fileName: "DailyDataDate.plist") {

            saveArray2(array: dailyDataDate, fileName: "DailyDataDate.plist")

        } else {

            let tempDlyDate = [dateStr] as NSArray

```

```

        let filePath : String = NSHomeDirectory() +
"/Documents/\\(userName)/DailyDataDate.plist"

        tempDlyDate.write(toFile: filePath, atomically: true)
    }

    /* upload to Firebase Database */

    // upload to DailyData

    var ref : DatabaseReference!

    ref = Database.database().reference()

    ref.child(mamaKid[1!]).child("\\(userName)\\(timeLog)").setValue(["Location" :
address])

    let dailyNum = 1

    let tempDlyDate = readPlist(fileName: "DailyDataDate.plist")

    //print(tempDlyDate)

    let x = tempDlyDate.count

    if x == 1 {

        let timesPerDay = NSArray(objects: dateStr, dailyNum)

        saveArray(array: timesPerDay, fileName: "TimesPerDay.plist")

        ref.child(mamaKid[2!]).child("\\(userName)\\(dateStr)").setValue(["Times" :
dailyNum])

        UserDefaults.standard.set(1, forKey: "\\(userName)tempDailyNumber")
    } else if x >= 2 {

        let a = tempDlyDate[x-1] as! String

        let b = tempDlyDate[x-2] as! String

```

```

if Int(a) == Int(b) {

    // overwrite

    tempDailyNum = tempDailyNum + 1

    var timesPerDay = [AnyObject]()

    timesPerDay = readPlist(fileName: "TimesPerDay.plist") as [AnyObject]

    timesPerDay.removeLast()

    timesPerDay.append(tempDailyNum as AnyObject)

    let    filePath    :    String    =    NSHomeDirectory()    +

"/Documents/\\(userName)/TimesPerDay.plist"

    let timesPerDayNSArray:NSArray = timesPerDay as NSArray

    timesPerDayNSArray.write(toFile: filePath, atomically: true)

    ref.child(mamaKid[2]!).child("\\(userName)/\\(a)").setValue(["Times"
tempDailyNum])

    UserDefaults.standard.set(tempDailyNum, forKey:

"\\(userName)tempDailyNumber")

} else {

    let timesPerDay = NSArray(objects: a, tempDailyNum)

    saveArray(array: timesPerDay, fileName: "TimesPerDay.plist")

    ref.child(mamaKid[2]!).child("\\(userName)/\\(a)").updateChildValues(["Times" : dailyNum])

    UserDefaults.standard.set(1, forKey: "\\(userName)tempDailyNumber")

}

}

```

```
    }  
  })  
}  
}  
}
```

BIBLIOGRAPHY

- [1] Chavez, M.D., *HeartMate: A Competitive and Motivational Fitness Application for iOS Devices*. 2016, ProQuest Dissertations Publishing.
- [2] Adibi, S., *Introduction*, in *Mobile Health: A Technology Road Map*, S. Adibi, Editor. 2015, Springer International Publishing: Cham. p. 1-7.
- [3] Statista. *Number of mHealth apps available in the Apple App Store from 2nd quarter 2015 to 1st quarter 2018*. 2018; Available from: <https://www.statista.com/statistics/779910/health-apps-available-ios-worldwide/>.
- [4] Couch, H.C., *Providers' Acceptance of Smartphone Applications as a Supportive Strategy for Adolescent Asthma*. 2017, ProQuest Dissertations Publishing.
- [5] AsthmaMD. *Why AsthmaMD?* 2018; Available from: <https://www.asthmamd.org/>.
- [6] Health, P. *Start by Connecting Your Medications*. 2018; Available from: <https://www.propellerhealth.com/>.
- [7] Care, P.O. *Asthma Manager*. 2018; Available from: <https://itunes.apple.com/us/app/my-asthma-manager/id930678066?mt=8>.
- [8] Center, C.s.M. *My Asthma Pal*. 2018; Available from: <https://itunes.apple.com/us/app/my-asthma-pal/id1040235651?platform=iphone&preserveScrollPosition=true#platform/iphone>.
- [9] Neyghem, S.V. *Asthmatic - the first asthma weather forecast*. 2018; Available from: <https://itunes.apple.com/us/app/asthmatic-the-first-asthma-weather-forecast/id1102570869?mt=8>.
- [10] Training Systems Design, I. *Asthma & Me*. 2018; Available from: <https://www.mydiversepatients.com/app-asthma.html>.
- [11] Inc., A. *Welcome to Swift.org*. 2018; Available from: <https://swift.org/>.
- [12] Baker, R. *iOS Architecture*. 2018; Available from: <https://intellipaat.com/tutorial/ios-tutorial/ios-architecture/>.

- [13] Liu, C., et al., *Status and trends of mobile-health applications for iOS devices: A developer's perspective*. The Journal of Systems & Software, 2011. **84**(11): p. 2022-2033.
- [14] Ohland, B. and J. Varma, *Xcode 7 essentials: step up your iOS development with the power and wealth of features of Xcode 7*. Second ed. 2016, Birmingham: Packt Publishing.
- [15] Gomez, C., J. Oller, and J. Paradells, *Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology*. Sensors, 2012. **12**(9): p. 11734-11753.
- [16] Apple, I. *ResearchKit*. 2018; Available from: <http://researchkit.org/index.html>.
- [17] Apple, I. *UIKit*. 2018; Available from: <https://developer.apple.com/documentation/uikit>.
- [18] Apple, I. *Framework MapKit*. 2018; Available from: <https://developer.apple.com/documentation/mapkit?changes=8>.
- [19] Apple, I. *Framework WebKit*. 2018; Available from: <https://developer.apple.com/documentation/webkit>.
- [20] Apple, I. *Core Location*. 2018; Available from: <https://developer.apple.com/documentation/corelocation?changes=5>.
- [21] Daniel Cohen Gindi, P.J. *Charts*. 2018; Available from: <https://github.com/danielgindi/Charts>.
- [22] Database, F.R. *Installation & setup on iOS*. 2018; Available from: <https://firebase.google.com/docs/database/ios/start?authuser=0>.
- [23] Yahiaoui, H., *Firebase Cookbook*. 1 ed. 2017, Birmingham: Packt Publishing, Limited.
- [24] Firebase. *Structure Your Database*. 2018; Available from: <https://firebase.google.com/docs/database/ios/structure-data?authuser=0>.
- [25] Apple, I. *Beta Testing Made Simple*. 2018; Available from: <https://developer.apple.com/testflight/>.
- [26] Rozen, R. *TestFlight Tutorial: iOS Beta Testing*. May 23, 2018; Available from: <https://www.raywenderlich.com/190493/testflight-tutorial-ios-beta-testing-2>.
- [27] American Academy of Allergy, A.I. *Mr. Nose-It-All's Word Game*. 2012; Available from: <https://www.aaaai.org/conditions-and-treatments/just-for-kids/mr-nose-it-alls-word-game>.

- [28] Publication, N. *Asthma Action Plan*. April, 2007; Available from: <https://www.aaaai.org/conditions-and-treatments/just-for-kids/mr-nose-it-alls-word-game>.