Monitoring Bipolar Disorder using Heart Rate, Sleep, and Mood Changes

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Rushmeet Bahra

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

Professor Arshia Khan

August 2017

Acknowledgements

I would like to thank my advisor, Dr Arshia Khan for her constant support and guidance. She has helped me to see this work through and overcome the barriers encountered during the course of this thesis.

I would like to thank my committee members, Dr Douglas Dunham and Dr Ona Egbue, for taking the time to review my work and Dr Pete Willemsen, Director of Graduate Studies for his constant effort in making sure that we continuously work on our thesis.

I would like to thank the Computer Science faculty Dr. Peter Peterson, Dr. Neda Saeedloi, Dr. Haiyang Wang, Dr Ted Pedersen, Dr. Richard Maclin for their core teachings on the computer science subject matter.

I would also like to take this opportunity to thank Lori Lucia, and Clare Ford, for their constant help.

Lastly, I would like to thank my family and friends for providing their undying encouragement.

Dedication

I would like to dedicate this thesis to my family for supporting me through thick and thin.

Abstract

According to World Health Organization, unipolar depressive disorders were ranked as the third leading cause of global burden in 2004 and predicted to move into the first place by 2030[16]. Bipolar Disorder affects approximately 5.7 million adult Americans every year (NIHM)[18]. Bipolar Disorder (BD) is recognized as a chronic mental illness, which requires consistent monitoring. Due to the unpredictability of this disease, any attempts to manage BD are limited to constant tracking of the patients' vitals, monitoring and auditing the behavior and the triggers in the environment.

This thesis develops an algorithm as a solution for predicting the onset of a bipolar episode with the use of mobile technology. Several factors can lead to increase in stress levels, which is a major cause for the onset of a bipolar episode. Variables such as sleep, mood and heart rate have been reported to have a direct impact on stress levels.

The prediction algorithm has been designed and developed for the iOS mobile platform in which various variables responsible for the onset of a bipolar episode are taken into account. Wearable sensor has been used to capture the heart rate information. Mood and sleep data is being obtained via a self-reporting mechanism where the application prompts the user to provide the input. This data is then used to identify a bipolar episode and generate alerts depending upon the prediction algorithm.

Contents

List of Figures

1 Introduction

The World Health Organization Global Burden of Disease Study lists 4 mental illnesses among the 10 leading causes of disability worldwide, specifically, unipolar major depression, bipolar disorder, schizophrenia, and obsessive-compulsive disorder[12].

Depression is a chronic mental illness experienced by numerous individuals, irrespective of age, sex, caste, and creed. It is one of the most common prevalent mental illnesses where an individual experiences sadness, dullness, anger and frustration for a longer period of time[9].

The various types of Depression are listed below:

Bipolar Disorder, Postpartum Depression, Seasonal Depression, and Premenstrual Dysphoric Disorder.

Bipolar Disorder is one of the greatest challenges in modern psychiatry[22]. It is a prevalent major illness with high rates of morbidity, comorbidity, disability, and mortality[13]. Patients suffering from Bipolar Disorder may get a rush of feelings, known as an episode where they can either harm themselves or people around them. It is very difficult to figure the exact point of time when an episode will occur, therefore constant monitoring of the patient's vitals and behavior is highly essential.

Studies have proved that patients suffering from Bipolar Disorder utilizes nearly three to four times the health care resources and incurred over four times greater costs per patient as compared to the non-bipolar group. Patients suffering with this disease incur the highest healthcare costs, therefore treatment of bipolar disorder is very expensive to patients and

health insurers. Hence, there is a need to find ways to reduce and manage the overall cost for these patients.

There are a number of treatments which have proved successful over the period of time for people suffering from bipolar disorder. A few of the cures includes use of antidepressant medications, shock therapy, psychotherapy, music therapy, and most of all the will of the person to overcome the illness along with the support from their loved ones.

We hypothesized that by tracking individual vitals as heart rate, sleep patterns, and mood changes, we can potentially predict the onset of an episode, or predict the possibility of an episode so that essential measures are taken before catastrophic repercussions.

As part of this thesis, an algorithm was developed to predict a bipolar episode depending upon the values provided for the factors responsible for bipolar disorder. This thesis exercised the advantage of wireless technology to track and monitor patients suffering from bipolar disorder. The prediction algorithm has been implemented for a mobile application where an existing wearable sensor has been used to monitor patient's heart rate[4]. The application prompts the users to provide data regarding their mood and sleep. An amalgamation of self-reported data and sensor based reported data was then fed into the system and subjected to the algorithm; which uses this data and determines if the patient is susceptible to an episode. If the patient is on the verge of an outbreak, an alert gets generated indicating that the patient should seek medical assistance.

This study also provides patients suffering from bipolar disorder and their families a way to be better prepared for the episode and how to tackle it. It provides enough information about the disease and measures to be taken in case of a bipolar attack. This study showcases an example of amalgamation of two disciplines - we make use of the tools available to us by Technology to make a difference in the medical world.

2 Background

2.1 What is Depression?

Human beings are susceptible to experience changes in emotions - an individual may experience feelings of happiness, joy, sadness, anger, anxiety and more. These feelings are indispensable and everybody experiences them at one time or another. Individuals who are undergoing adverse and stressful situations are likely to feel sad and dejected. Such feelings do not typically lead to depression; sulking for a day or two does not qualify as one being under depression. If these feelings persist and linger for a long period of time then it can potentially develop into depression. Depression is much more than just sadness as defined by the American Psychological Association. It is when people feel emptiness in their lives, lose all hope and think very low of themselves. People tend to behave in a different manner as well when they are in depression. It is not that they just feel sad, they can also act out and become super aggressive, or rude.

2.2 What are the Causes of Depression?

Depression can be triggered because of any reason - it could stress at workplace or studies, some financial troubles, issues at home, and many more. There are many causes of depression and it is really difficult to figure out the exact cause for triggering of depression. If someone even thinks that they are in depression they should immediately consult a doctor

to know for sure and figure out ways to tackle that. Different causes could be:- Genetic, Environmental, Hormonal, and Situational[6].

2.3 Types of depression

There are different types of depression:

2.3.1 Postpartum Depression

Postpartum Depression is a form of depression which is prevalent in women. Women which fall into the depression hole as in when they give birth are said to be in Postpartum Depression. This is an extreme condition which is faced by pregnant women where they go into a state of shock right after the moment their child is conceived. Women who face a lot of difficulties during their pregnancies, difficulties like lack of support from the family, handling everything on their own, or if there are a lot of complications during their pregnancy; they happen to fall victims to this kind of depression.

2.3.2 Seasonal Depression

Seasonal Depression, as the same suggests is a form of depression which occurs because of seasonal changes. This form of depression could occur to anyone irrespective of age, or sex. One, People living in places where they are exposed to only one type of weather can experience this because of the sheer monotonicity of the climate. People start feeling sad because of their surrounding temperature which does not change throughout the year and thus get dull. Another, People who live in places where one season (particularly winters) persists for a longer time; implying that there is very less or no exposure to sun could fall into

a state of dullness and lethargy. If these conditions remain for a longer time, then the person could very well go into depression where he/she feels so sad that they don't even leave their beds. Surroundings can have a huge impact on people's emotions, positive surroundings inculcates positive behavior whereas a negative or dull environment can turn the boat to extreme ends where the person loses their will to live.

2.3.3 PreMenstrual Dysphoric Disorder

PMDD is another form of depression which is experienced by women. Women undergo menstruation every month. Menstruation could be very hard on a woman's body and mind. A few days before the menstruation starts the woman starts experiencing hormonal changes because of which she remains irritated, short tempered, and/or lethargic. This is known as Premenstrual Syndrome. Even during their menstruation, these feelings continues to be prevalent. Women tend to experience irritation, mood swings, and lack of strength/energy; but all these feelings are because of the hormonal changes that happen in a woman's body during that time and these feelings go away as in when the woman is done with her monthly cycle. But, there is a possibility that even when the monthly cycle is finished, the female tends to exhibit extreme tension, irritation, and/or frustration and these feelings contain themselves for a longer during then that could be symbolic of the fact that the woman has fallen prey to the PreMenstrual Dysphoric Disorder.

2.3.4 Bipolar Disorder

Bipolar Disorder is a mental illness and can occur to anyone irrespective of age or sex. This disease has two sides to it. One is Bipolar Depression, and the other one is bipolar manic. Bipolar Depression, is the one extreme conditions where an individual feels extreme sadness and it lasts for days, weeks or even months. A person suffering from bipolar depression tends to become very negative towards oneself. The person starts thinking worse about oneself, loses all its focus, and zest to live. This disease can make a person go to a point where he/she wants to kill themselves.

Bipolar Manic is the condition where the person is extremely active and doesn't rest. The person experiences extreme happiness or irritability and this behavior lasts for a long

period of time.

2.4 Consequences and the Ripple Effect

People experiencing depression should be handled with a lot of care. People suffering from this disease are very edgy, irritable, may experience insomnia, and may attempt suicide. This disease has the capacity to suck the life out of a living human being and may cause serious damage to him or people around them. These people may indulge in substance abuse as well i.e. they become heavy alcohol consumers or start using drugs just to get on with their daily schedules. Obviously, when people fall under depression, they tend to behave in a way which can intensively affect the people around them. There could be a situation where depressed people may try and assault a person who is trying to help them.

2.5 Management of Depression

Depression is a very serious illness and must not be taken lightly. Early detection and treatment can make huge difference in an individual's life. Depression can treated well and easily, with fewer medications, if it is caught at early stages. Although it is not the case that depression is not curable if caught at later stages, but the only thing is it becomes difficult to treat it soon and it may take good amount of time. There are many ways with which Depression can be cured. First, by the use of medications like antidepressants. Second, by the use of psychology as a means of refactoring the brain to change negative thoughts and make them positive. Third, by shock therapy which is, in reality, giving electric shocks to the brain of the person so as to bring the person back to his senses. All these methods may cure a person suffering from depression. It takes different amounts of time for curing

depression depending upon the type and intensity of their state of mind. With depression, every case is unique and there is no set pattern for individuals. Anything which is associated with the mind cannot have a definite path or reasoning. In order to treat depression, the most important cure is the person's own willingness and willpower to overcome this state of mind.

2.6 Bipolar Disorder

2.6.1 What is Bipolar Disorder?

Bipolar means having two extremities or relating to two poles. Bipolar, term itself is enough to insinuate the meaning of bipolar disorder. Bipolar Disorder is a chronic mental illness which is very common - more than 3 million cases are encountered every year in the United States. Bipolar Disorder comprises of Bipolar Depression and Bipolar Manic - the two extremities of the disease i.e. the two sides of the coin (Bipolar Disorder).

Bipolar Depression is the condition where in an individual feels extreme sadness and it lasts for days, weeks or even months. Here, the chances of suicide are very high and the people around the person who experiences this are under danger as well. The person experiences self-loathing, or is too tired to get out of bed.

Bipolar Manic is the condition where in the person experiences high energy levels with reduced need for sleep. The person exhibits elevated mood or irritability and this behavior continues for a long period of time.

2.6.2 Causes of Bipolar Disorder

The causes for bipolar depression are unknown but mainly it is hereditary. However, even with people who have this inherent vulnerability, doesn't necessarily mean that they develop the illness. There are many triggering factors like stress, seasonal changes, substance abuse, medication or sleep deprivation.

2.6.3 Symptoms of Bipolar Disorder

Various symptoms for this disease are:

Irritability

Inability to experience pleasure

Fatigue or loss of energy

Physical and mental sluggishness

Appetite or weight changes

Sleep problems

Concentration and memory problems

Feelings of worthlessness or guilt

Thoughts of death or suicide

2.6.4 Management of Bipolar Disorder

In order to cure bipolar depression, early detection and assistance is vital. Use of antidepressants is a very common approach towards managing depression but the use of traditional antidepressants to treat bipolar depression is considered experimental, and none are FDA-approved for that purpose. There is no research to show that they have any greater benefit than taking a mood stabilizer alone. Antidepressants can actually worsen the conditions and can increase the risk of suicidal behavior, but proper care and regular tracking by the psychiatrist can make a difference[15].

2.7 Technology in Medicine

Technology has always been an essential part for medicine. It has been central to medicine since 1800s. Starting from scalpels, probes, stethoscope till advent of x-rays, ECGs, MRIs in the 20th century, medicine has always relied on technology. Since the 1800s doctors and scientists have invested a good amount of time to develop instruments to examine and understand the human body. Devices such as the thermometer, microscope, stethoscope revealed how a healthy or diseased a human body is. The 20th century marked the use of computers as the most important technological change in the field of medicine. They became central to medical care from the 1950s. Computerized machines in hospitals monitored patients continuously. Imaging techniques such as MRI or PET were possible because faster computers could reconstruct images of the body[21]. Technology has been giving effective solutions for treating neurological diseases like depression and bipolar disorder. Classical ways includes talk therapy, medication, peer support, and a personal wellness plan. However, people who don't respond to these traditional methods or need additional help managing their illness, hence, turns to technological alternatives. In the 1800, electrical therapy was very huge for treating depression. Also, commonly known as shock therapy or Electrotherapy - which made use of electricity. In 1930, researches discovered that applying small amount of electrical current to the brain can cause small seizures to the brain which can change the brain chemistry - which is the cause of bipolar disorder (i.e. imbalance of brain chemistry).

Few other technological treatments for bipolar disorder includes Transcranial Magnetic Stimulation (TMS), and Vagus Nerve Stimulation (VNS). TMS works by using a special electromagnetic device that is placed on the scalp and sends short bursts of energy to the brain. These pulses of energy stimulate nerve cells in the part of the brain that is associated with mood regulation - which is another cause for bipolar disorder.

As for VNS, it is delivered through a small pulse generator, which is inserted into the left chest area and connected to the vagus nerve - (vagus nerve is one of the primary communication pathways from the major organs of the body to the brain) in the left side of the neck. The pulse generator sends small pulses to the vagus nerve, which then delivers these pulses directly to the brain. This therapy targets specific areas of the brain that affect mood and other symptoms of depression. It also influences the activity of neurotransmitters, such as serotonin and norepinephrine[19].

Nowadays, mobile technology is one the most ubiquitous and dynamic technologies that have come into scope. Everything has become on the go, and on smartphones - starting from voice conversations to having to access your emails, personal records, etc[10]. Mobile technology has the potential to revolutionize the field of medicine and to some extent it has already started to impact the field. Starting from having access to the latest medical research at the point of care to being able to communicate at a moment's notice with physicians and colleagues around the world, we are practicing medicine in a technological age[20]. Various smartphones applications are being made to monitor the mental health of patients specially bipolar disorder, where it is vital to constantly track the changes in moods, heart rate, voice modulation, sleep patterns of a patient[14].

As part of this thesis, we are using technology to provide early assistance to the patients suffering from bipolar disorder so that extreme damage to the patient and to their loved ones can be avoided. By doing so the health care cost can be controlled as well. It is highly essential to constantly track and monitor the symptoms of patients suffering from bipolar disorder. According to [14], bipolar disorder has a direct correlation to mood, heart rate, and sleep. It is probable that studying and monitoring these three dimensions associated with a patient, we might be able to predict a prodrome.

A prediction algorithm has been designed which has been implemented for an iOS mobile application. The mobile application monitors the three dimensions associated with a

bipolar patient. Wireless technology is being used to feed this application with heart rate data i.e. a wearable sensor is being used to gather this information and then it is fed into the application. The application has a self-reporting vertical as well where the patients are providing data regarding their mood and sleep, and the same application poses an alert by studying the provided data when or if a bipolar episode is about to be triggered.

Below is a baseline algorithm to explain the flow of information:

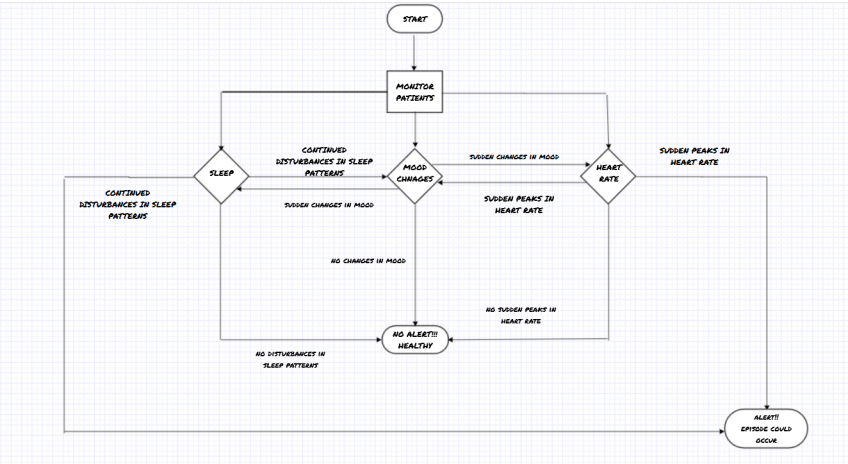

Figure 2.1: Baseline Algorithm Flowchart

According to the prediction algorithm, first and foremost the mood factor of the patient is taken account. If the mood value provided by the patient is either depressed or elated then the application validates the heart rate information. If there are huge variations for the heart rate data then the application takes into consideration the last factor i.e. sleep and checks if the patient has been sleeping for longer or shorter durations. By observing all the three variables an alert gets generated if required and which can then be used to notify clinicians and patient's family members so that they can intervene at the right instant and avoid any extreme conditions.

3 Implementation

The unpredictability of Bipolar Disorder makes it essential to constantly monitor and track the symptoms of the audience subjected to this condition. Early detection of a bipolar episode can prevent undesirable consequences that can potentially cause harm to the patient or the people around the affected individual.

The three critical factors that needs to be monitored in order to be able to predict an episode are

Heart Rate,

Mood Changes, and

Sleep Patterns.

There have been multiple studies which makes use of one or two of the above mentioned factors as a means for detecting a bipolar episode. While this work is making use of all the three factors for predicting an episode and providing early intervention so as to avoid any harmful after-effects.

Technology has provided a number of platforms for implementing medical solutions in an effective manner and obtain critical results. One of these platforms is mobile technology which makes use of portable, handheld devices connected to wireless networks for communication purposes. Use of smartphones is no longer restricted to communication purposes, nowadays they are being used as handheld computers due to their powerful on-board computing capability, spacious memories, large screens and open operating systems

that encourage application development. With the increase in the development of mobile applications, there has been a huge increase in interfacing these applications with the external devices. Because of this flexibility available in smartphones due to their computing capabilities, there is a great potential of using mobile technology in developing healthcare solutions[5].

One such solution that we have proposed as part of this thesis is a prediction algorithm, implemented for a mobile application for Monitoring Bipolar Disorder. An iOS application has been developed in the Swift programming language for constant monitoring and tracking of the patient's vital signs who are suffering from bipolar disorder. The factors being accounted for the onset of a bipolar episode in this work are:

heart rate in beats per minute,

number of hours of sleep, and

mood value from depressed to elated.

We have made use of a wearable external sensor for getting the heart rate data. As for Mood value and Sleep hours, a self reporting system is being used. Data from all these sources is clubbed together and depending upon the prediction algorithm, results are generated i.e. whether the individual is about to experience an episode or not. If the episode is about to be triggered then an alert is generated stating that the patient needs medical assistance.

Along with providing the essential features for tracking and predicting an episode, this application also provides general information about the disease and how to manage the health of an individual suffering from bipolar disorder. We have sign-on functionality to ensure the security of personal health information being provided by the individual.

Heart Rate

The data used for tracking the Heart Rate factor of the individual is obtained via wearable sensors. These wearable devices which are in close contact with the individual provides a lot of information accounting for the heart factor. The data that is captured by the heart monitor is then fed to the mobile application interface via bluetooth technology. Bluetooth is a wireless standard for exchanging data over short distances from mobile devices.

An existing heart rate monitor being used for this purpose is Polar Heart Rate Sensor H7[8]. This device is a low bluetooth energy (BLE Bluetooth Low Energy) which makes it easier for the device to interface with new mobile applications.

This device provides a number of features like:

HEART RATE MONITOR:

Provides live, accurate heart rate to compatible mobile training applications

FITNESS TRACKER:

Waterproof heart rate sensor compatible with a number of smartphones

ADVANCED TECHNOLOGY:

Gym Link connects Polar heart rate monitors and activity trackers with compatible gym equipment

CALORIE COUNTER CAPABILITIES:

For those who want to count the calories they burn and get basic heart rate-based features to keep their fitness training simple

Sleep Patterns

The data for Sleep is gathered as a form of user input. The user of the application needs to provide the time at which they are going to bed and then after they wake up, by doing so the application is going to calculate the total hours of sleep and use this information as the sleep factor for the algorithm. The algorithm makes sure that the time being entered is not

past the current time, this ways the patient has to provide true values.

Mood Changes

The data for Mood Changes factor is obtained via a self reporting mechanism where the user will provide this information to the application. The individual suffering from bipolar disorder or a family member who has access to the application will provide input regarding the emotions the individual is experiencing during the day.

Mood Changes data will be accounted based on five values - neutral, sad, depressed, happy, and elated.

This information provided by the individual is then used as one of the contributing factors for predicting a bipolar episode.

All the data given to the application is analyzed and an alert gets generated in an event of an episode so as to ensure early intervention. This data should be collected over a period of time, at-least a week(seven days) so as to get better predictions.

There are many advantages of having a mobile application taking care of all your health related information. Few of them includes -

All the data is available at all times, and on the go.

It can be used to constantly keep track of your health.

It can provide the corresponding intervention at the earliest possible instant.

3.1 The Swift Programming Language

This work makes use of Swift 3.1 as the programming language with Xcode version 8.3.2 as the development environment for the creation of the mobile application for iOS devices.

Swift is a new programming language developed by Apple Inc for iOS and OS X development. It is a general-purpose programming language built using a modern approach to safety, performance, and software design patterns[1]. It adopts the best of C and Objective-C, without the constraints of C compatibility[11]. Swift unifies the procedural and object-oriented portions of the language and is designed to make writing and maintaining correct programs easier for the developer[1]. It provides seamless access to existing Cocoa and Cocoa Touch frameworks[11].

Although inspired by Objective-C and many other languages, Swift is not itself a C-derived language. As a complete and independent language, Swift packages core features like flow control, data structures, and functions, with high-level constructs like objects, protocols, closures, and generics. Swift embraces modules, eliminating the need for headers and the code duplication they entail[3].

3.2 Core Bluetooth Framework

The application uses an external sensor to get heart rate data which connects to the application via Bluetooth technology. Apple provides a framework for communication between Bluetooth Low Energy devices and iOS applications.

Bluetooth Low Energy(BLE: Also known as Bluetooth Smart is a lightweight subset of classic Bluetooth and is intended to provide reduced power consumption and cost while maintaining a similar communication range.

Core Bluetooth Framework: It is an abstraction of the Bluetooth 4.0 specification i.e. it hides many of the low-level details of the specification from the developer making it much easier for them to develop apps that interact with Bluetooth low energy devices. It defines a set of easy-to-use protocols for communicating with BLE devices.

There are two major players involved in BLE communication: the central and the peripheral. A central uses the information served up by peripherals to accomplish some particular task. In our case central will be the iOS device and peripheral is the heart rate monitor. Advertising is the primary way that peripherals make their presence known via Bluetooth LE. The job of a central is to scan for these advertising packets, identify any peripherals it finds relevant, and connect to individual devices for more information.

Advertising packets are very small therefore they cannot contain a great deal of information. In order to get more, a central needs to connect to a peripheral to obtain all of the data available. Once the central connects to a peripheral, it needs to choose the data it is interested in. In Bluetooth LE, data is organized into concepts called services and characteristics:

A service is a collection of data and associated behaviors describing a specific function or feature of a device. For our application, a service is a heart rate monitor exposing heart

rate data from the heart rate sensor. A device can have more than one service.

A characteristic provides further details about a peripheral's service. For example, the heart rate service contains a characteristic that describes the intended body location of the heart rate sensor and another characteristic that transmits heart rate measurement data. A service can have more than one characteristic.

Once a central has established a connection to a peripheral, it's free to discover the full range of services and characteristics of the peripheral and to read or write the characteristic values of the available services[7].

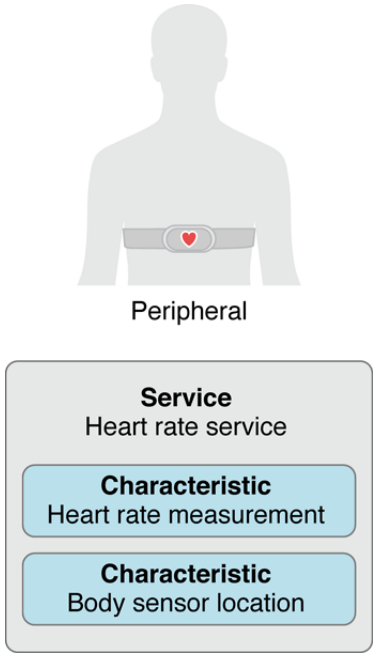

Figure 3.1: CBF - Peripheral's Services and Characteristics

3.3 Database using Core Data

In order to get the data for the three factors over multiple days, there is a need to store the past data. This data is then used to calculate the average values which in turn takes part in the prediction algorithm. Apple provides an object graph and persistence framework known as Core Data which has been used for this application. Core Data is a framework that you use to manage the model layer objects in your application[2]. It provides generalized and automated solutions to common tasks associated with object life cycle and object graph management, including persistence. It allows data organized by the relational entity attribute model to be serialized into XML, binary, or SQLite stores. The data can be manipulated using higher level objects representing entities and their relationships. Core Data manages the serialized version, providing object lifecycle and object graph management, including persistence. Core Data interfaces directly with SQLite, insulating the developer from the underlying SQL[17].

3.4 Mobile Application

As part of the solution to the problem of monitoring bipolar disorder a mobile application has been developed. The mobile application is for iOS users using Swift programming language. The developed application uses a tab bar controller to incorporate access to multiple screens using slide functionality.

This section provides screen by screen description of the mobile application developed to manage bipolar disorder.

Log-In Screen

This is the first screen that pops up when you launch the application right after the logo screen (refer Figure 3.2). It is a standard login screen with username and password as login credentials (refer Figure 3.3). If you are using the application for the first time, you need to register using the ”Register” button. Clicking on the that button will open another screen known as ”Register” where you can provide your username and password details and register for the application.

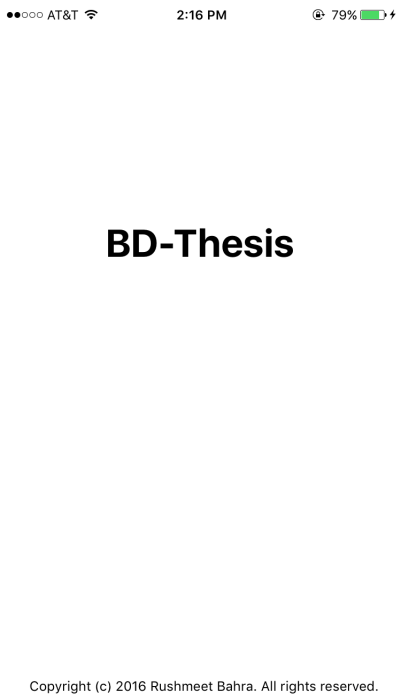

Figure 3.2: Logo Screen

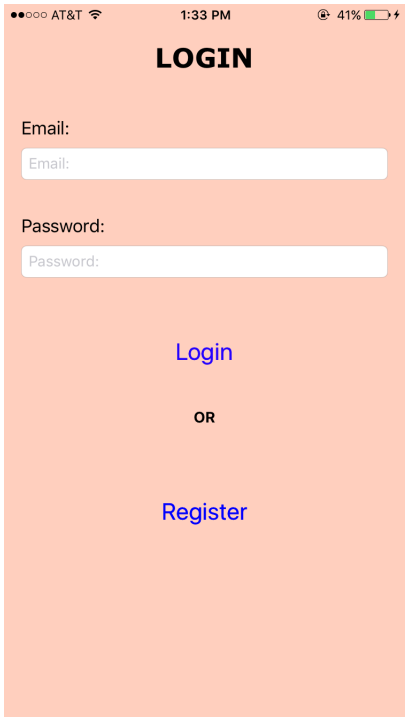

Figure 3.3: Log-In Screen

If you are not a first time user, you enter your registered credentials into the provided fields and click on ”Login” button. Upon clicking the button, application checks if the fields are not empty and then compare the values with the stored values (refer Figures 3.4 and 3.5). If the entered values match then an alert ”Login is successful” is generated and the user is logged in (refer Figure 3.6).

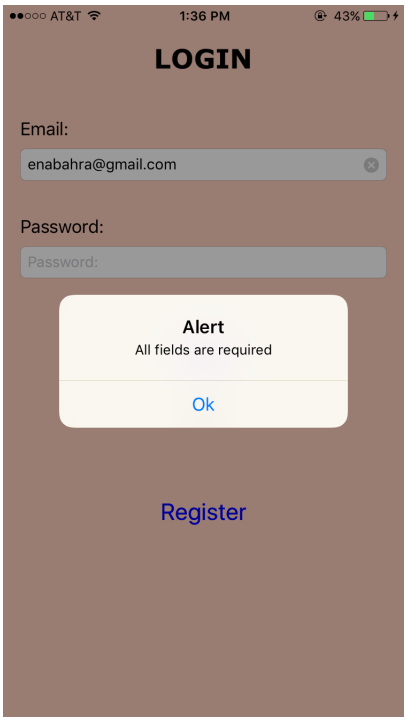

Figure 3.4: All fields required check

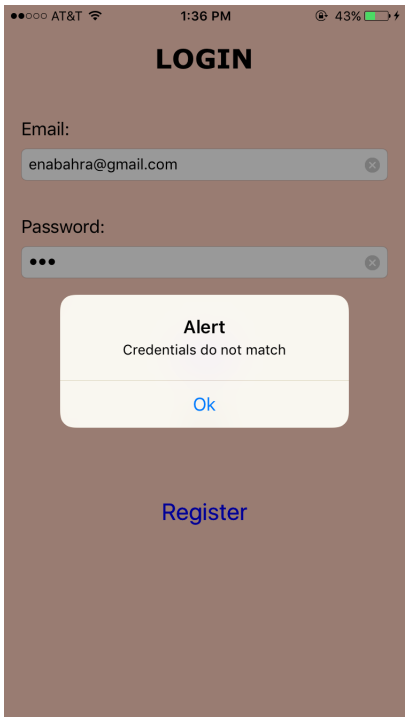

Figure 3.5: Credentials don't match check

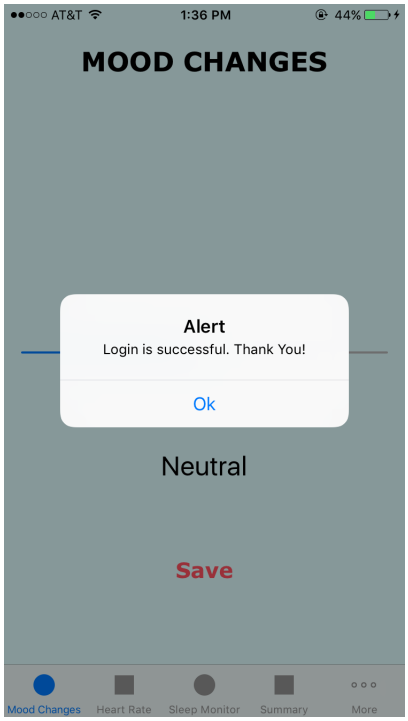

Figure 3.6: Login Successful

Registration Screen

This screen is launched when the user clicks on ”Register” button present on the Log-In screen. If the user is using the application for the first time then registration is necessary hence, this page needs to be launched; otherwise the user can directly login into the application (refer Figure 3.7).

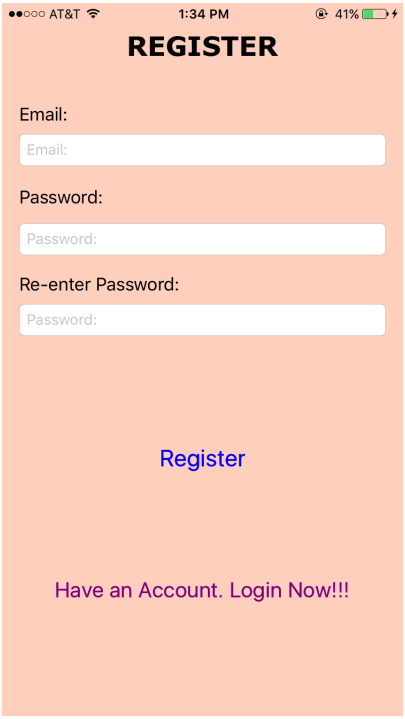

Figure 3.7: Registration Screen

As part of registration, the user needs to provide a valid email address and password. Upon entering the information, the user needs to click on Register button which will perform the below validations (refer Figure 3.8 and Figure 3.9):

1. All the fields should be non empty.
2. Values for the fields Password and Re-enter Password should match.

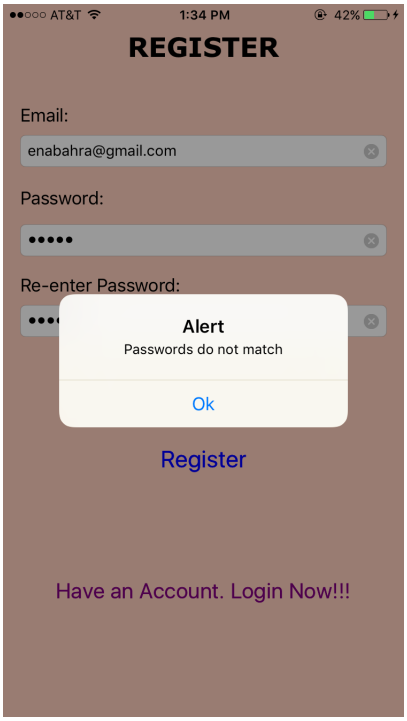

Figure 3.8: All fields required check

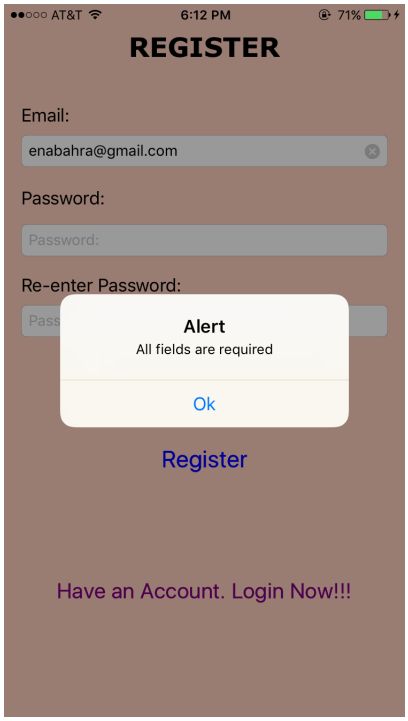

Figure 3.9: Passwords don't match check

If the above validations fail then alerts are generated stating the same or else if the validations checks out the user is directed to the Log-In screen with an alert stating "Registration is successful" (refer Figure 3.10).

There is also a "Log-In" button present on the screen which can be used to redirect the user to the login page upon their will.

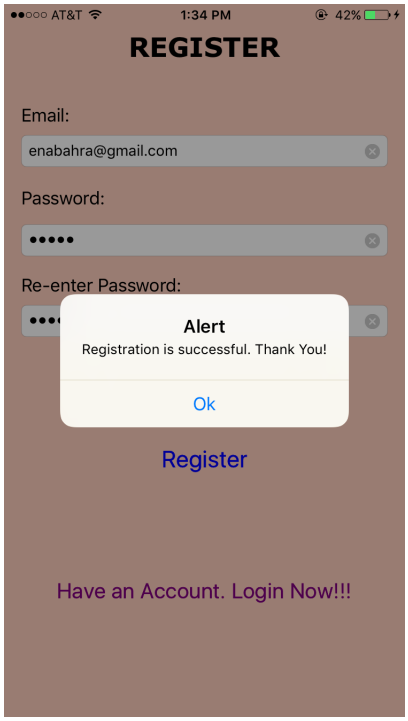

Figure 3.10: Registration Successful

Mood Changes Screen

This screen is used to get the data for mood factor for the prediction algorithm. The user provides this information with the use of a slider which has the following values (refer Figure 3.11): Depressed, Sad, Neutral, Happy, and Elated.

Once the value is selected on the slider, a corresponding numeric value is displayed in the text field just below the slider (refer Figure 3.12). Clicking on Save button records that value as the mood factor and is displayed on the Summary screen.

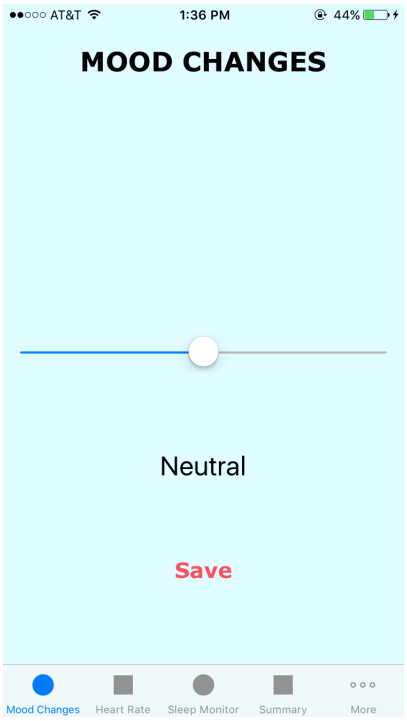

Figure 3.11: Mood Initial Screen

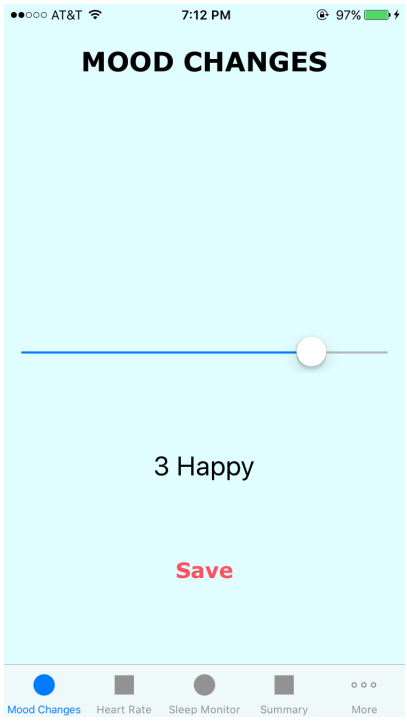

Figure 3.12: Mood Save Input

Heart Rate Screen

This screen is used to get the heart rate information from an external device. For this screen to function, the bluetooth of the device must be turned on and the external device Polar Heart Rate Monitor needs to be in use by the patient (refer Figure 3.13).

As in when you slide to this screen with the bluetooth turned on for your mobile device, the Core Bluetooth framework will be set into effect. It will start scanning for the peripheral devices. As soon as it finds the heart rate monitor, the mobile device is connected to the monitor and it starts getting the heart rate information from the external device. This information is displayed in the given text field present on the screen.

The screen is also provided with a Save button, which when clicked records the heart rate in beats per minute and displays it on the Summary screen.

Along with the heart rate information, the screen also captures some secondary infor-

mation of the heart rate device such as the Manufacturer Name, and location of the external device on the body (refer Figure 3.14).

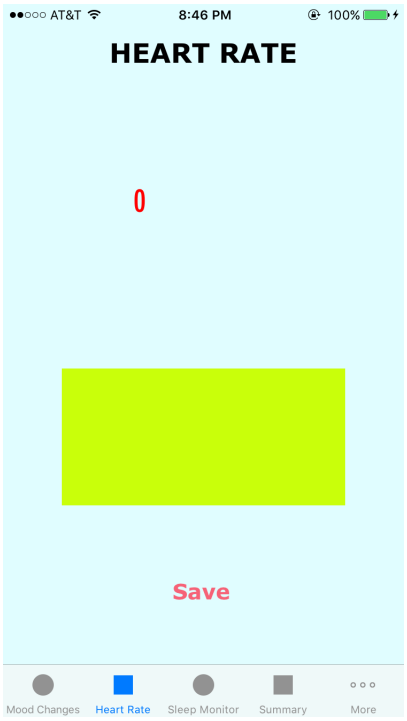

Figure 3.13: Heart Rate Monitor Initial Screen

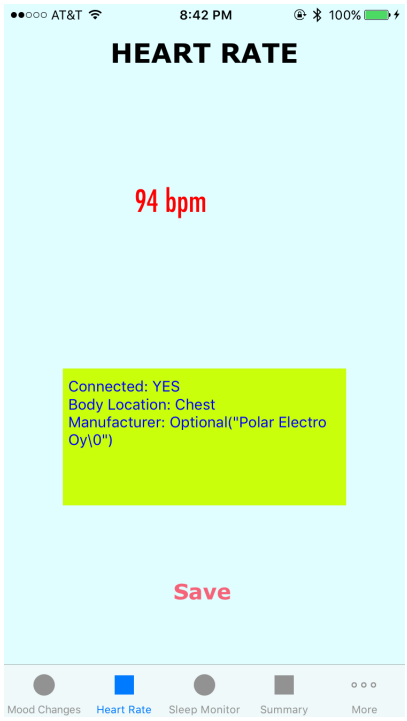

Figure 3.14: Heart Rate Monitor with Values

Sleep Monitor Screen

This screen is used to gather the sleep information. This information is provided by the user in terms of the time at which they went to bed and when they woke up (refer Figure 3.15).

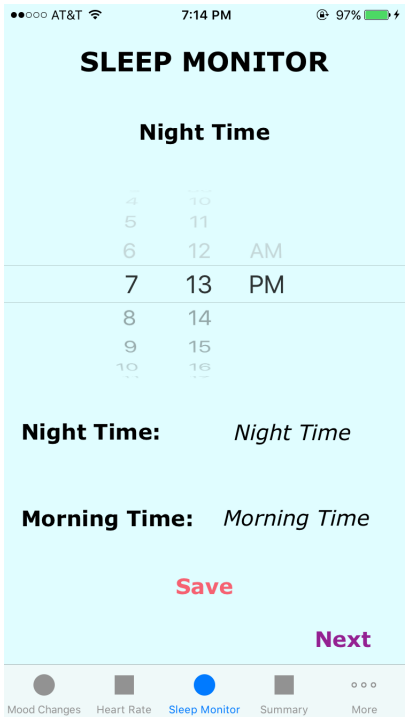

Figure 3.15: Sleep Monitor Screen

As in when you slide to the screen, the user is prompted to enter sleep time. User selects a time from the date-time picker, that time is reflected in the text field provided on the screen (refer Figure 3.16).

After that, the Next button is clicked to move to the second part of the sleep monitor (refer Figure 3.17). This is used to record the wake up time by the user so as in when the user wakes up he/she needs to record the time into the application. The time is selected from the time picker so as to be reflected into the Morning Time text field (refer Figure 3.18). Upon clicking the Save button, that value is transferred back to the first screen for the sleep monitor. Once the operation is complete, the Done button is clicked which takes the user back to the Night Time screen.

On the Night screen, both the times are reflected in the appropriate labels. Upon clicking the Save button available on the screen, the values are recorded and displayed on the Summary screen (refer Figure 3.19).

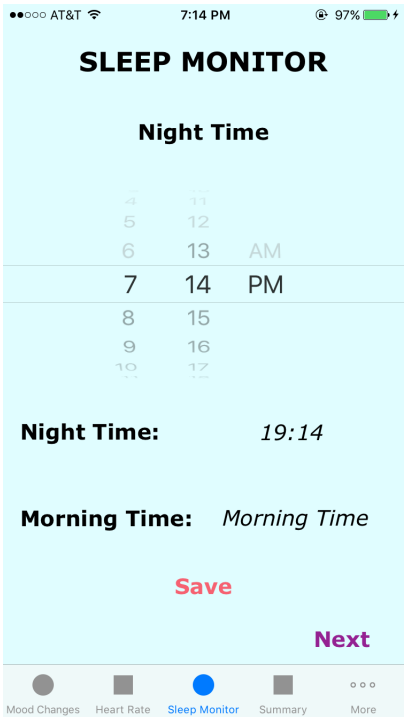

Figure 3.16: Night time selected

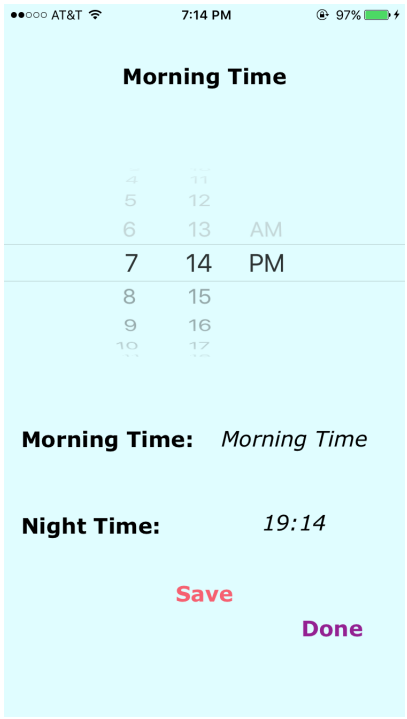

Figure 3.17: Next button clicked

Both the time values are recorded based on 24 hour clock. By doing so, we ensure that the user provides correct and continuous information without any room for error and confusion.

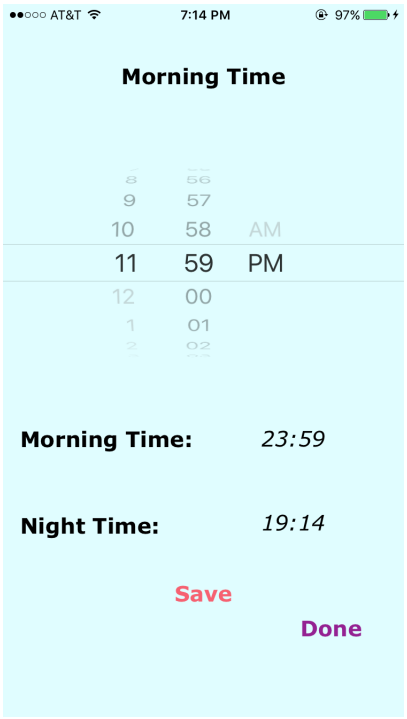

Figure 3.18: Morning time selected

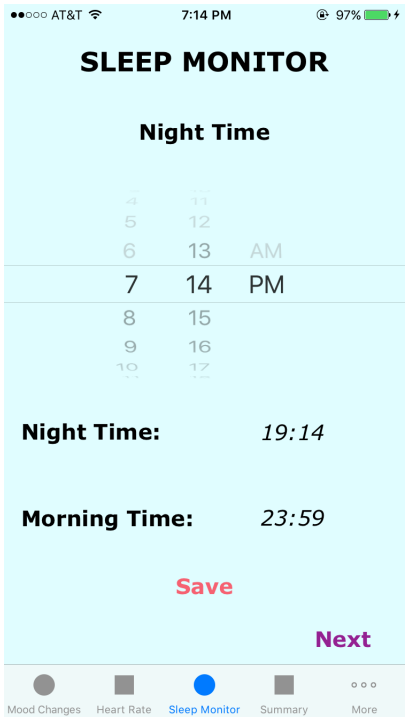

Figure 3.19: Save button clicked

Summary Screen

This screen provides the consolidated values of the all the factors responsible for predicting a bipolar episode (refer Figure 3.20). There is Save button available on the screen, which when clicked saves the values in the database. Since the values are collected over a period of time, the save button needs to be clicked everytime the user provides new values (refer Figure 3.21).

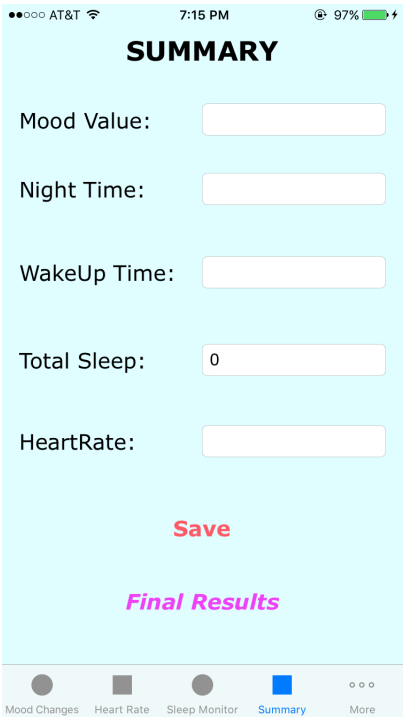

Figure 3.20: Summary initial view

The Final Results button is used as so to kick the prediction algorithm into effect. As in when the button is pressed, all the stored values are considered - their average is calculated and are then subjected to the prediction algorithm; which then generates alerts as the person being healthy or requiring medical assistance (refer Figure 3.22).

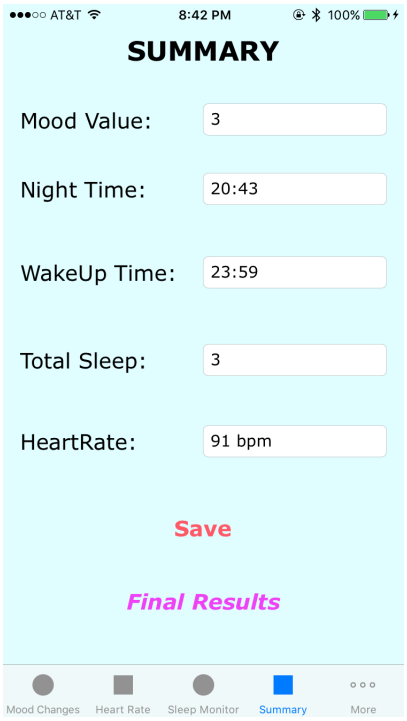

Figure 3.21: Summary with values

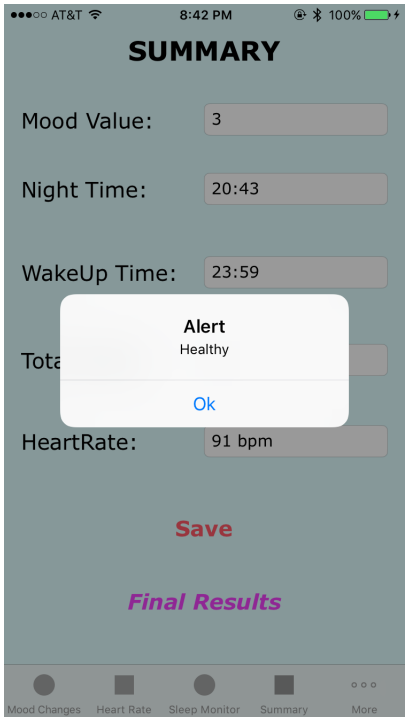

Figure 3.22: Final Results button clicked

Education Screen

This screen is used to provide information to the users regarding the disease, how to manage it and how to use the application (refer Figures 3.23, 3.24, 3.25 and 3.26).

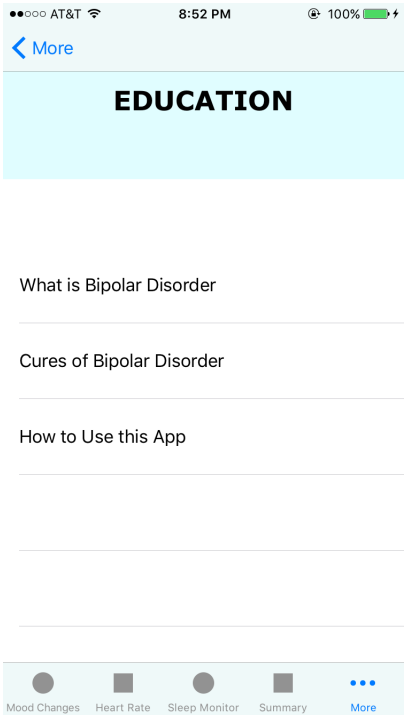

Figure 3.23: Education Screen

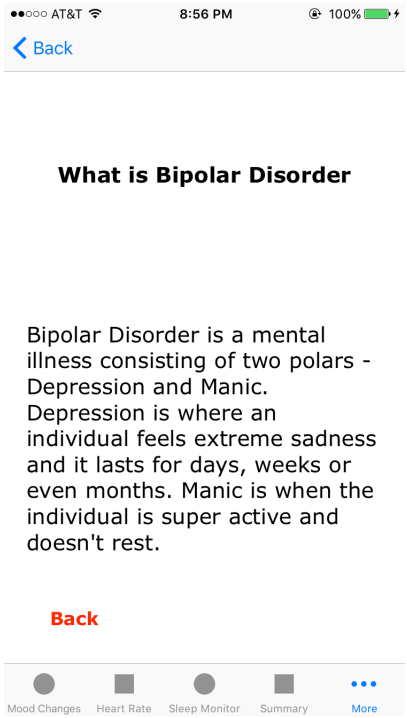

Figure 3.24: Education part 1

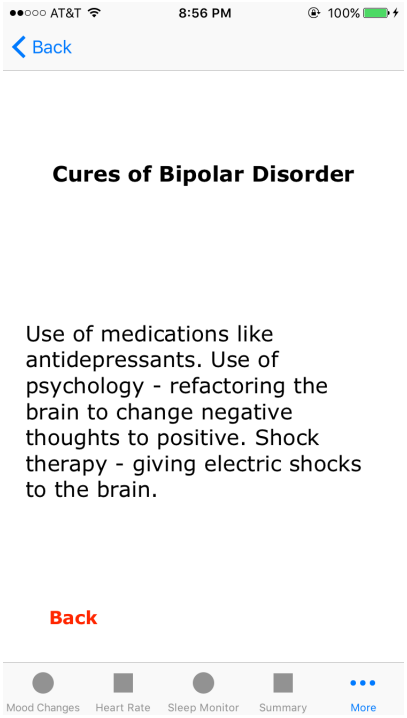

Figure 3.25: Education part 2

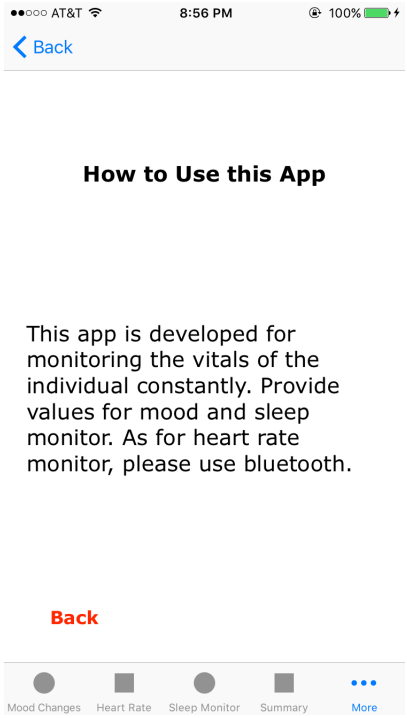

Figure 3.26: Education part 3

Sign Out Screen

This page is used to logout of the application. The user upon clicking the Sign Out button is directed back to the Login Page (refer Figure 3.27).

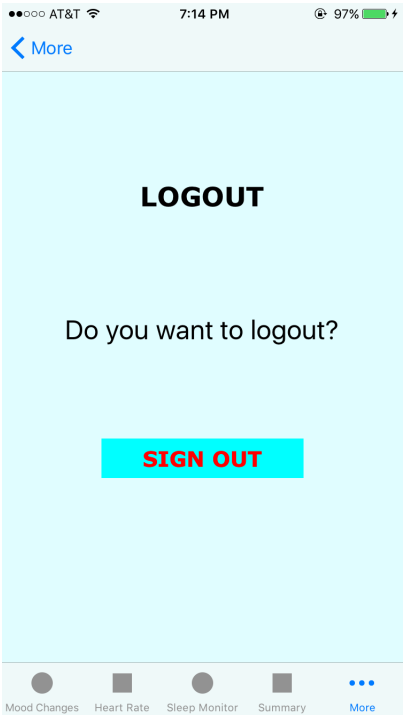

Figure 3.27: Sign Out Screen

4 Results

This section discusses the bipolar episode prediction algorithm in detail. Along with the algorithm, this section provides the description of how the application behaves when dummy data is provided and how the algorithm produces alerts based on the data provided to the application.

4.1 Prediction Algorithm Used

The prediction algorithm used as part of this work is a basic decision tree algorithm, where the average values of the three factors are compared to their threshold values and depending upon their ranges an alert is generated.

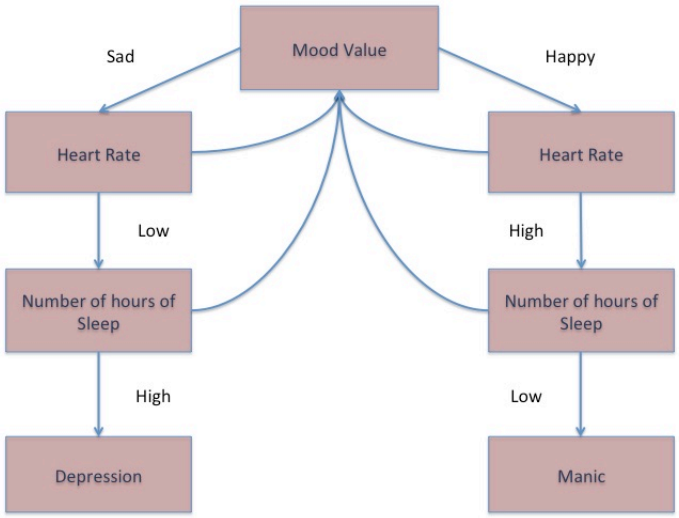

Figure 4.1: Prediction Algorithm

```
if(avgm <= -3){
  if(avgh <= 80){
    if(avgs >= 10){
                        displayalert("Consult a Doctor - Depression")
                        return;}
                    else{
                        displayalert("Healthy")
                      return;}
                  }
                else{
                      displayalert("Healthy")
                      return;}
}
else if(avgm>=3){
  if(avgh>=120){
    if(avgs<=4){
                    displayalert("Consult a Doctor - Manic")
                            return;}
          else{
                          displayalert("Healthy")
                          return;}
            }
  else{
    displayalert("Healthy")
                  return;}
}
else{
  displayalert("Healthy")
         return;
}
```

4.1.1 Depression Prediction

Firstly, the average value for the Mood factor is considered. If this value is less than or equal to -3 (Sad) then the average value for Heart Rate is considered. If this value is on the lower end i.e. the average heart beats per minute is less than or equal to 80 bpm (beats per

minute) then the average value for Sleep is considered. If this value is on the higher end i.e. the person is sleeping for a high number of hours; this value is more than or equal to 10 hours then an alert is generated stating that the person should consult a doctor as there is risk of him/her experiencing a Depression attack.

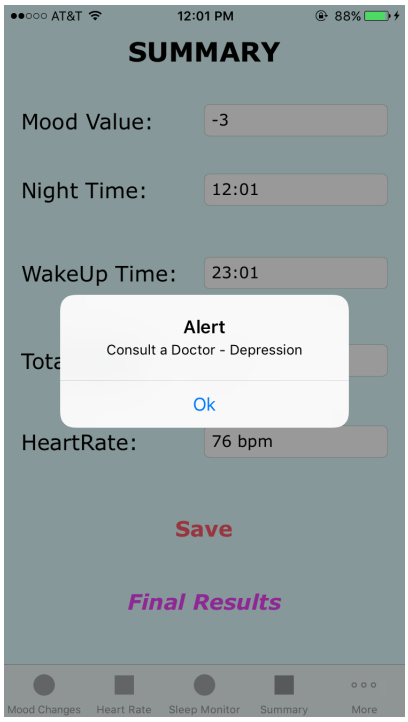

Figure 4.2: Depression Alert

4.1.2 Manic Prediction

Firstly, the average value for Mood factor is considered. If this value is more than or equal to 3 (Happy) then the average value for Heart Rate is considered. If this value is on the higher end i.e. the average heart beats per minute is more than or equal to 120 bpm (beats per minute) then the average value for Sleep is considered. If this value is on the lower end i.e. the person is either not sleeping or sleeping for a very few number of hours; this value is less than or equal to 4 hours then an alert is generated stating that the person should consult a doctor as there is risk of him/her experiencing a Manic attack.

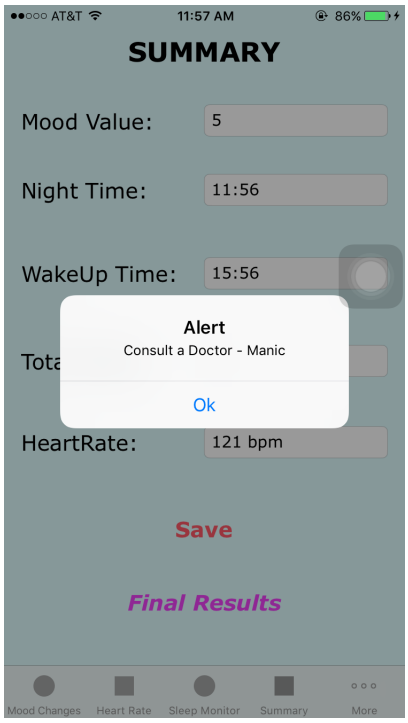

Figure 4.3: Manic Alert

4.1.3 Healthy

If none of the conditions discussed in the above sections for Manic and Depressive episodes are met then the person is healthy and an alert stating the same is generated.

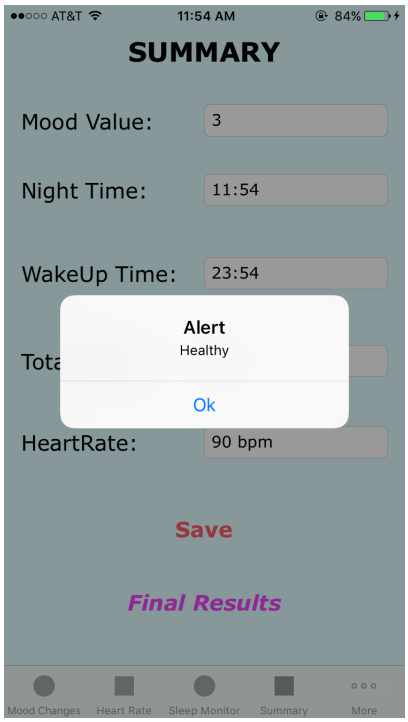

Figure 4.4: Healthy Alert

5 Conclusions and Future Work

Prediction of a bipolar disorder episode is very difficult therefore constant efforts are being made to be able to try and predict an episode. By doing so, there is a possibility of avoiding any adverse repercussions. Many studies have been performed to gather information regarding the vitals that are affected from bipolar disorder - three of them are changes in mood, disruptions in sleep, and variations in heart rate.

Many medical solutions are focussed on tackling the three given factors for bipolar disorder. One of the most essential way in which bipolar episode can be predicted is by constantly monitoring patient's vitals.

This work attempts to provide a solution for people suffering with bipolar disorder by the use of mobile technology so that it is available and accessible at all times. In this work, the values for mood and sleep factors are taken as user inputs and the value for heart rate is gathered by the use of a wearable sensor which connects to the application via Bluetooth. All the values are then fed to the baseline prediction algorithm and generate alerts.

As part of future work, the prediction algorithm needs to be modified so as to consider an individual medical history. The value for sleep factor should be taken as a "Sleep Score" provided by a wearable sensor (like Beddit 3.0) instead of calculating number of hours of sleep. By doing so we would reduce the application dependency on the user. Finally, the application should be tested on human subjects which would provide better evaluation and accuracy of the algorithm.

Bibliography

[1] Apple. *About Swift*. 2017. URL: https://swift.org/about/ (cit. on p. 18).

[2] Apple. *Core Data Framework*. 2017. URL: https://developer.apple.com/library/content/documentation/Cocoa/Conceptual/CoreData/index.html (cit. on p. 21).

[3] Apple. *GitHub Swift*. 2017. URL: https://github.com/apple/swift (cit. on p. 18).

[4] J. Blum and E. Magill. "M-psychiatry: Sensor networks for psychiatric health monitoring". In: *Proceedings of The 9th Annual Postgraduate Symposium The Convergence of Telecommunications, Networking and Broadcasting, Liverpool John Moores University*. 2008, pp. 33–37 (cit. on p. 2).

[5] M. N. K. Boulos, S. Wheeler, C. Tavares, and R. Jones. "How smartphones are changing the face of mobile and participatory healthcare: an overview, with example from eCAALYX". In: *Biomedical engineering online* 10.1 (2011), p. 24 (cit. on p. 15).

[6] J. Cohen. "Bioinformatics&Mdash;an Introduction for Computer Scientists". In: *ACM Comput. Surv.* 36.2 (June 2004), pp. 122–158. ISSN: 0360-0300. DOI: 10.1145/1031120.1031122. URL: http://doi.acm.org/10.1145/1031120.1031122 (cit. on p. 4).

[7] S. Daniel. *Core Bluetooth Framework*. 2013. URL: https://www.raywenderlich.com/52080/introduction-core-bluetooth-building-heart-rate-monitor (cit. on p. 20).

[8] P. HR. *Polar Heart Rate Sensor*. URL: https://www.polar.com/en/products/accessories/H7_heart_rate_sensor (cit. on p. 16).

[9] W. MD. *Web MD - Depression*. URL: http://www.webmd.com/depression/guide/chronic-illnesses-depressionf (cit. on p. 1).

[10] Oxford. *Oxford Journal*. URL: http://cid.oxfordjournals.org/content/47/1/117.full.pdf+html (cit. on p. 12).

[11] T. Point. *Swift Language*. URL: https://www.tutorialspoint.com/swift/ (cit. on p. 18).

[12] L. A. Pratt. *Psychological Data*. 2001 - 2004. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.218.9280&rep=rep1&&type=pdf (cit. on p. 1).

[13] ps.psychiatryonline.org. *Bipolar Disorder - PsychiatryOnline*. URL: http://ps.psychiatryonline.org/doi/pdf/10.1176/ps.2007.58.1.85 (cit. on p. 1).

[14] L. S. Talbot, S. Stone, J. Gruber, I. S. Hairston, P. Eidelman, and A. G. Harvey. "A test of the bidirectional association between sleep and mood in bipolar disorder and insomnia." In: *Journal of abnormal psychology* 121.1 (2012), p. 39 (cit. on p. 12).

[15] WebMd. *Bipolar Disorder*. URL: http://www.webmd.com/bipolar-disorder/antidepressants-for-bipolar (cit. on p. 10).

[16] WHO. *WHO Health Management Depression*. URL: http://www.who.int/mental_health/management/depression/wfmh_paper_depression_wmhd_2012.pdf (cit. on p. iii).

[17] Wikipedia. *Core Data Framework*. 2017. URL: https://en.wikipedia.org/wiki/Core_Data (cit. on p. 21).

[18] www.dbsalliance.org. *Statistics of Bipolar Disorder*. URL: http://www.dbsalliance.org/site/PageServer?pagename=education_statistics_bipolar_disorder (cit. on p. iii).

[19] www.dbsalliance.org. *Technology in Medicine*. URL: http://www.dbsalliance.org/site/PageServer?pagename=wellness%5Cdepression%5Cemerging%5Ctechnologies (cit. on p. 12).

[20] www.jmir.org. *Technology in Medicine*. URL: http://www.jmir.org/2012/5/e128/ (cit. on p. 12).

[21] www.sciencemuseum.org.uk. *Technology in Medicine*. URL: http://www.sciencemuseum.org.uk/broughttolife/themes/technologies (cit. on p. 11).

[22] www.tandfonline.com. *Bipolar Disorder*. URL: http://www.tandfonline.com/doi/full/10.3109/10673221003747955 (cit. on p. 1).

A Appendix A

This contains code for the individual screens of the application.

A.1 BipolarDisorder.swift

```
import UIKit
class BipolarDisorder: NSObject {
    var fmood: String = " "
    var fbpm: String = " "
    var nsleep: String = " "
    var msleep: String = " "
}
```

A.2 BDTabController.swift

```
import UIKit
class BDTabController: UITabBarController {
    let myreading = BipolarDisorder()
}
```

A.3 VC.swift

```
import UIKit
class VC: UIViewController {
    @IBOutlet weak var myDatePicker: UIDatePicker!
    @IBOutlet weak var mtLabel: UILabel!
    var endtime: String = " "
    @IBOutlet weak var nttime: UILabel!
```

```
    let currentDate = NSDate()
    var timeFormatter = DateFormatter()
    var start: String = " "

    @IBAction func myDatePickerAction(_ sender: Any) {
        var dateFormatter = DateFormatter()
        dateFormatter.dateFormat = "HH:mm"
        var strDate = dateFormatter.string(from: myDatePicker.date)
        self.mtLabel.text = strDate
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        myDatePicker.datePickerMode = UIDatePickerMode.time
        //myDatePicker.minimumDate = currentDate as Date
        myDatePicker.date = currentDate as Date
        nttime.text = start
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
    }


    @IBAction func doneAction(_ sender: Any) {
        [self.navigationController?.popViewController(animated: true)]
       dismiss(animated: true, completion: nil)
    }

    @IBAction func savemt(_ sender: Any) {
        endtime = mtLabel.text!
        et = endtime
    }

    override func viewWillAppear(_ animated: Bool) {
        super.viewWillAppear(animated)
    }
}
```

A.4 AppDelegate.swift

```
import UIKit
import CoreData
@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?
    func application(_ application: UIApplication, didFinishLaunchingWithOptions
    launchOptions: [UIApplicationLaunchOptionsKey: Any]?) -> Bool {
        // Override point for customization after application launch.
        let storyboard = UIStoryboard(name: "Main", bundle: nil);
        let loginVC = storyboard.instantiateViewController(withIdentifier: "LoginVC") as!
    ForthViewController
        self.window?.rootViewController = loginVC;
        return true
    }

    func applicationWillResignActive(_ application: UIApplication) {
      }

    func applicationDidEnterBackground(_ application: UIApplication) {
   }

    func applicationWillEnterForeground(_ application: UIApplication) {
            }

    func applicationDidBecomeActive(_ application: UIApplication) {
       }

    func applicationWillTerminate(_ application: UIApplication) {
        self.saveContext()
    }

    // MARK: - Core Data stack
   lazy var applicationDocumentsDirectory: NSURL = {
        let urls = FileManager.default.urls(for: .documentDirectory, in: .userDomainMask)
        return urls[urls.count-1] as NSURL
    }()
```

```
lazy var managedObjectModel: NSManagedObjectModel = {
    let modelURL = Bundle.main.url(forResource: "BD-Thesis", withExtension: "momd")!
    return NSManagedObjectModel(contentsOf: modelURL)!
}()

lazy var persistentStoreCoordinator: NSPersistentStoreCoordinator = {
    let coordinator = NSPersistentStoreCoordinator(managedObjectModel: self.
managedObjectModel)
    let url = self.applicationDocumentsDirectory.appendingPathComponent("BD-Thesis.
sqlite")
    var failureReason = "There was an error creating or loading the application's
saved data."
    do {
        try coordinator.addPersistentStore(ofType: NSSQLiteStoreType,
configurationName: nil, at: url, options: nil)
    } catch {
        // Report any error we got.
        var dict = [String: AnyObject]()
        dict[NSLocalizedDescriptionKey] = "Failed to initialize the application's
saved data" as AnyObject
        dict[NSLocalizedFailureReasonErrorKey] = failureReason as AnyObject

        dict[NSUnderlyingErrorKey] = error as NSError
        let wrappedError = NSError(domain: "YOUR_ERROR_DOMAIN", code: 9999, userInfo:
dict)
        // abort() causes the application to generate a crash log and terminate.
        NSLog("Unresolved error \(wrappedError), \(wrappedError.userInfo)")
        abort()
    }
    return coordinator
}()

lazy var managedObjectContext: NSManagedObjectContext = {
    // Returns the managed object context for the application
    // (which is already bound to the persistent store coordinator for the application
.)
    let coordinator = self.persistentStoreCoordinator
    var managedObjectContext = NSManagedObjectContext(concurrencyType: .
mainQueueConcurrencyType)
```

```
        managedObjectContext.persistentStoreCoordinator = coordinator
        return managedObjectContext
    }()

    // MARK: - Core Data Saving support
    func saveContext () {
        if managedObjectContext.hasChanges {
            do {
                try managedObjectContext.save()
            } catch {
                // abort() causes the application to generate a crash log and terminate.
                let nserror = error as NSError
                NSLog("Unresolved error \(nserror), \(nserror.userInfo)")
                abort()
            }
        }
    }
}
```

A.5 SignOutViewController.swift

```
import UIKit
class SignOutViewController: UIViewController {

    @IBAction func logout(_ sender: AnyObject) {
        let storyboard = UIStoryboard(name: "Main", bundle: nil)
        let loginVC = storyboard.instantiateViewController(withIdentifier: "LoginVC")
        let appDelegate = UIApplication.shared.delegate as! AppDelegate
        appDelegate.window?.rootViewController = loginVC
    }

    override func viewDidLoad() {
        super.viewDidLoad()
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
    }
```

```
}
```

A.6 FirstViewController.swift

```
import UIKit
class FirstViewController: UIViewController {

    @IBOutlet weak var slider: UISlider!

    @IBOutlet weak var MVal: UILabel!
    var moodval: String = " "
    var myreading = BipolarDisorder()

    @IBAction func MChange(_ sender: AnyObject) {
        let currentValue = Int(slider.value)
        var Mood: String
        switch currentValue {
        case 0:
            Mood = "Neutral"
            moodval = "0";
        case -5:
            Mood = "Super Sad"
            moodval = "-5";
        case 5:
            Mood = "Super Happy"
            moodval = "5";
        case -3:
            Mood = "Sad"
            moodval = "-3";
        case 3:
            Mood = "Happy"
            moodval = "3";
        default:
            Mood = "Try Again"
            moodval = "0"
        }
      MVal.text = "\(currentValue)" + " " + "\(Mood)"
    }
```

```
    @IBAction func savem(_ sender: Any) {
        myreading.fmood = moodval

    }

    override func viewWillAppear(_ animated: Bool) {
        super.viewWillAppear(animated)
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        let tbc = tabBarController as! BDTabController
        myreading = tbc.myreading
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
    }
}
```

A.7 Summary.swift

```
import UIKit
import CoreData
var results: [NSManagedObject] = []
class Summary: UIViewController {
    var myreading = BipolarDisorder()


    @IBOutlet weak var fm: UITextField!
    @IBOutlet weak var nt: UITextField!
    @IBOutlet weak var wt: UITextField!
    @IBOutlet weak var thrs: UITextField!
    @IBOutlet weak var fhr: UITextField!
    var heartrate: String = " "

    override func viewDidLoad() {
```

```
    super.viewDidLoad()
    myreading = (tabBarController as! BDTabController).myreading
}

override func viewWillAppear(_ animated: Bool) {
    fm.text = myreading.fmood
    nt.text = myreading.nsleep
    wt.text = myreading.msleep
    fhr.text = myreading.fbpm
    var token = fhr.text?.components(separatedBy: " ")
    heartrate = (token?[0])!
    var s: Int = (nt.text! as NSString).integerValue
    var e: Int = (wt.text! as NSString).integerValue
    var total: Int = abs(s - e)
    thrs.text = String(total)
}

override func didReceiveMemoryWarning() {
    super.didReceiveMemoryWarning()
}

@IBAction func resaction(_ sender: Any) {
    let appDelegate =
        UIApplication.shared.delegate as? AppDelegate
    let managedContext = appDelegate?.managedObjectContext
    let entity = NSEntityDescription.entity(forEntityName: "Result", in:
managedContext!)
    let res = NSManagedObject(entity: entity!, insertInto: managedContext)
    res.setValue(fm.text, forKeyPath: "mood")
    res.setValue(thrs.text, forKey: "numhrs")
    res.setValue(heartrate, forKey: "heartrt")

    do {
        try managedContext?.save()
        results.append(res)
        } catch let error as NSError {
        print("Could not save. \(error), \(error.userInfo)")
    }
 }
```

```
@IBAction func final(_ sender: Any) {
    var summ: Int = 0
    var sums: Int = 0
    var avgm: Int = 0
    var avgs: Int = 0
    var sumh: Int = 0
    var avgh: Int = 0
    for i in 0  ..< results.count  {
        summ += abs((results[i].value(forKey: "mood") as! NSString).integerValue)
        sums += (results[i].value(forKey: "numhrs") as!  NSString).integerValue
        sumh += (results[i].value(forKey: "heartrt") as! NSString).integerValue
    }

    avgm = summ/results.count
    avgs = sums/results.count
    avgh = sumh/results.count

    print(avgm)
    print(avgs)
    print(avgh)

    //the algorithm
    if(avgm <= -3)
    {
        if(avgh <= 80)
        {
            if(avgs >= 10)
            {
                displayalert("Consult a Doctor - Depression")
                return;
            }
            else
            {
                displayalert("Healthy")
                return;
            }

        }
```

```
        else
        {
            displayalert("Healthy")
            return;
        }
    }
    else if(avgm >= 3)
    {
            if(avgh >= 120)
            {
                if(avgs <= 4)
                {
                    displayalert("Consult a Doctor - Manic")
                    return;
                }
                else
                {
                    displayalert("Healthy")
                    return;
                }
        }
        else
            {
                displayalert("Healthy")
                return;
        }
    }
    else
    {
        displayalert("Healthy")
        return;
    }
}

func displayalert(_ usrMessage: String)
{
    let myAlert = UIAlertController(title: "Alert", message: usrMessage,
preferredStyle: UIAlertControllerStyle.alert);
    let okAction = UIAlertAction(title: "Ok", style: UIAlertActionStyle.default,
```

```
    handler:nil);
        myAlert.addAction(okAction);
        self.present(myAlert, animated: true, completion: nil);
    }
}
```

A.8 SecondViewController.swift

```
//https://www.raywenderlich.com/52080/
//introduction-core-bluetooth-building-heart-rate-monitor

import UIKit
import CoreBluetooth
import QuartzCore
class SecondViewController: UIViewController, CBCentralManagerDelegate,
    CBPeripheralDelegate {

    let POLARH7_HRM_DEVICE_INFO_SERVICE_UUID = "180A"
    let POLARH7_HRM_HEART_RATE_SERVICE_UUID = "180D"
    let POLARH7_HRM_MEASUREMENT_CHARACTERISTIC_UUID = "2A37"
    let POLARH7_HRM_BODY_LOCATION_CHARACTERISTIC_UUID = "2A38"
    let POLARH7_HRM_MANUFACTURER_NAME_CHARACTERISTIC_UUID = "2A29"

    var centralManager: CBCentralManager!
    var polarH7HRMPeripheral: CBPeripheral!

    var myreading = BipolarDisorder()

    // Properties for your Object controls
    @IBOutlet weak var heartImage: UIImageView!
    @IBOutlet weak var deviceInfo: UITextView!

    // Properties to hold data characteristics for the peripheral device
    var connected:String = ""
    var bodyData:String = ""
    var manufacturer:String = ""
    var polarH7DeviceData:String? = ""
    var heartRate: UInt16 = 0
```

```
// Properties to handle storing the BPM and heart beat
var heartRateBPM: UILabel!
var pulseTimer: Timer!


override func viewDidLoad() {
    super.viewDidLoad()
    let centralManager = CBCentralManager(delegate: self, queue: nil)
    self.centralManager = centralManager
    self.polarH7DeviceData = " "
    self.heartImage.image = UIImage.animatedImageNamed("Heart Image", duration: 2.2)

    // Clear out textView
    self.deviceInfo.text = " "
    self.deviceInfo.textColor = UIColor.blue
    self.deviceInfo.isUserInteractionEnabled = false

    // Create your Heart Rate BPM Label
    self.heartRateBPM = UILabel(frame: CGRect(x: 55, y: 30, width: 90, height: 50))
    self.heartRateBPM.textColor = UIColor.red
    self.heartRateBPM.text = "\(0)"
    self.heartRateBPM.font = UIFont(name: "Futura-CondensedMedium", size: 28)
    self.heartImage.addSubview(self.heartRateBPM)

    // Scan for all available CoreBluetooth LE devices
    self.centralManager = centralManager
    let tbc = tabBarController as! BDTabController
    myreading = tbc.myreading
}

override func didReceiveMemoryWarning() {
    super.didReceiveMemoryWarning()
}

// MARK: - CBCentralManagerDelegate
// method called whenever you have successfully connected to the BLE peripheral
func centralManager(_ central: CBCentralManager, didConnect peripheral: CBPeripheral)
{
```

```
    peripheral.delegate = self
    peripheral.discoverServices(nil)
    self.connected = "Connected: \(peripheral.state == .connected ? "YES" : "NO")"
    print("\(self.connected)")
}

// CBCentralManagerDelegate - This is called with the CBPeripheral class as its main
input parameter.
//This contains most of the information there is to know about a BLE peripheral.
func centralManager(_ central: CBCentralManager, didDiscover peripheral: CBPeripheral,
 advertisementData: [String: Any], rssi RSSI: NSNumber) {
    var localName: String? = (advertisementData[CBAdvertisementDataLocalNameKey] as?
String)
    if (localName?.characters.count ?? 0) > 0 {
        print("Found the heart rate monitor: \(String(describing: localName))")
        self.centralManager.stopScan()
        self.polarH7HRMPeripheral = peripheral
        peripheral.delegate = self
        self.centralManager.connect(peripheral, options: nil)
    }
}

// method called whenever the device state changes.
func centralManagerDidUpdateState(_ central: CBCentralManager) {
    // Determine the state of the peripheral
    if central.state == .poweredOff {
        print("CoreBluetooth BLE hardware is powered off")

    }
    else if central.state == .poweredOn {
        print("CoreBluetooth BLE hardware is powered on and ready")
        let serviceUUIDs = [CBUUID(string: POLARH7_HRM_HEART_RATE_SERVICE_UUID),
CBUUID(string: POLARH7_HRM_DEVICE_INFO_SERVICE_UUID)]
        let lastPeripherals = centralManager.retrieveConnectedPeripherals(withServices
: serviceUUIDs)
        if lastPeripherals.count > 0{
            let device = lastPeripherals.last! as CBPeripheral;
            let connectingPeripheral = device;
            centralManager.connect(connectingPeripheral, options: nil)
```

```
        }
        else {
            centralManager.scanForPeripherals(withServices: serviceUUIDs, options: nil
)
        }
    }
    else if central.state == .unauthorized {
        print("CoreBluetooth BLE state is unauthorized")
    }
    else if central.state == .unknown {
        print("CoreBluetooth BLE state is unknown")
    }
    else if central.state == .unsupported {
        print("CoreBluetooth BLE hardware is unsupported on this platform")
    }
}

// MARK: - CBPeripheralDelegate
// CBPeripheralDelegate - Invoked when you discover the peripheral's available
services.
func peripheral(_ peripheral: CBPeripheral, didDiscoverServices error: Error?) {
    for service: CBService in peripheral.services! {
        print("Discovered service: \(service.uuid)")
        peripheral.discoverCharacteristics(nil, for: service)
    }
}


// Invoked when you discover the characteristics of a specified service.
func peripheral(_ peripheral: CBPeripheral, didDiscoverCharacteristicsFor service:
CBService, error: Error?) {
    if service.uuid.isEqual(CBUUID(string: POLARH7_HRM_HEART_RATE_SERVICE_UUID)) {
        for aChar: CBCharacteristic in service.characteristics! {
            // Request heart rate notifications
            if aChar.uuid.isEqual(CBUUID(string:
POLARH7_HRM_MEASUREMENT_CHARACTERISTIC_UUID)) {
                self.polarH7HRMPeripheral.setNotifyValue(true, for: aChar)
                print("Found heart rate measurement characteristic")
            }
```

```
            else if aChar.uuid.isEqual(CBUUID(string:
POLARH7_HRM_BODY_LOCATION_CHARACTERISTIC_UUID)) {
                self.polarH7HRMPeripheral.readValue(for: aChar)
                print("Found body sensor location characteristic")
            }
        }
    }

// Retrieve Device Information Services for the Manufacturer Name
if service.uuid.isEqual(CBUUID(string: POLARH7_HRM_DEVICE_INFO_SERVICE_UUID)) {
    for aChar: CBCharacteristic in service.characteristics! {
        if aChar.uuid.isEqual(CBUUID(string:
POLARH7_HRM_MANUFACTURER_NAME_CHARACTERISTIC_UUID)) {
            self.polarH7HRMPeripheral.readValue(for: aChar)
            print("Found a device manufacturer name characteristic")
            }
        }
    }
}

// Invoked when you retrieve a specified characteristic's value
func peripheral(_ peripheral: CBPeripheral, didUpdateValueFor characteristic:
CBCharacteristic, error: Error?){
    // Updated value for heart rate measurement received
    if characteristic.uuid.isEqual(CBUUID(string:
POLARH7_HRM_MEASUREMENT_CHARACTERISTIC_UUID)) {
        // Get the Heart Rate Monitor BPM
    self.getHeartBPMData(characteristic,error: nil)
    }
    // Retrieve the characteristic value for manufacturer name received
    if characteristic.uuid.isEqual(CBUUID(string:
POLARH7_HRM_MANUFACTURER_NAME_CHARACTERISTIC_UUID)) {
        self.getManufacturerName(characteristic)
    }
    else if characteristic.uuid.isEqual(CBUUID(string:
POLARH7_HRM_BODY_LOCATION_CHARACTERISTIC_UUID)) {
        self.getBodyLocation(characteristic)
    }
    // Add your constructed device information to your UITextView
```

```
    self.deviceInfo.text = "\(self.connected)\n\(self.bodyData)\n\(self.manufacturer)\
n"
}


// Instance method to get the heart rate BPM information
func getHeartBPMData(_ characteristic: CBCharacteristic, error: Error?) {
    // Get the Heart Rate Monitor BPM
    var data: Data? = characteristic.value
    var reportData = data?.withUnsafeBytes {[UInt8](UnsafeBufferPointer(start: $0,
count: (data?.count)!))}
    var bpm: UInt16 = 0
    if ((reportData?[0])!&0x01) == 0 {
    // Retrieve the BPM value for the Heart Rate Monitor
        bpm = UInt16(reportData![1])

    } else {
        bpm = CFSwapInt16LittleToHost(UInt16((reportData?[1])!))
    }
    print (bpm)

    // Display the heart rate value to the UI if no error occurred
    if ((characteristic.value != nil)) || !(error != nil) {
        self.heartRate = bpm
        self.heartRateBPM.text = "\(bpm) bpm"
        self.heartRateBPM.font = UIFont(name: "Futura-CondensedMedium", size: 28)
    }
    return

}

// Instance methods to grab device Manufacturer Name
func getManufacturerName(_ characteristic: CBCharacteristic) {
    let manufacturerName = String(data: characteristic.value!, encoding: String.
Encoding.utf8)
    manufacturer = "Manufacturer: \(String(describing: manufacturerName))"
    return
}
```

```
    //Instance method to get the body location of the device
    func getBodyLocation(_ characteristic: CBCharacteristic) {
        var sensorData: Data? = characteristic.value
        var bodyData = sensorData?.withUnsafeBytes {[UInt8](UnsafeBufferPointer(start: $0,
     count: (sensorData?.count)!))}
    if bodyData != nil {
            let bodyLocation: UInt8 = bodyData![0]
            self.bodyData = "Body Location: \(bodyLocation == 1 ? "Chest" : "Undefined")"
        }
        else {
            self.bodyData = "Body Location: N/A"
        }
        return
    }

    @IBAction func saveac(_ sender: Any) {
        myreading.fbpm = heartRateBPM.text!
    }
}
```

A.9 FifthViewController.swift

```
import UIKit
var myIndex = 0
let list = ["What is Bipolar Disorder", "Cures of Bipolar Disorder", "How to Use this App"
    ]
class FifthViewController: UIViewController, UITableViewDelegate, UITableViewDataSource {
 public func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) ->
    Int
    {
        return list.count
    }

    public func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) ->
    UITableViewCell
    {
        let cell = tableView.dequeueReusableCell(withIdentifier: "cell", for: indexPath)
        cell.textLabel?.text = list[indexPath.row]
```

```
        return cell
    }

    func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
        myIndex = indexPath.row
        performSegue(withIdentifier: "segue", sender: self)
   }
}
```

A.10 ViewController.swift

```
import UIKit
class ViewController: UIViewController {

    @IBOutlet weak var myLabel: UILabel!
    @IBOutlet weak var myText: UITextView!

    override func viewDidLoad() {
        super.viewDidLoad()
     myLabel.text = list[myIndex]
   }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
    }

    @IBAction func back(_ sender: Any) {
        [self.navigationController?.popViewController(animated: true)]
        dismiss(animated: true, completion: nil)
    }
}
```

A.11 RegisterPageViewController.swift

```
import UIKit
class RegisterPageViewController: UIViewController, UITextFieldDelegate {
```

```
@IBOutlet weak var userEmail: UITextField!
@IBOutlet weak var userPassword: UITextField!
@IBOutlet weak var passRepeat: UITextField!

@IBAction func registerButton(_ sender: AnyObject) {
    let usrEm = userEmail.text;
    let usrPass = userPassword.text;
    let usrRepeat = passRepeat.text;

    //Check for Empty Fields
    if ((usrEm?.isEmpty)! || (usrPass?.isEmpty)! || (usrRepeat?.isEmpty)!)
    {
        //Display Alert
        displayalert("All fields are required")
        return;
    }

    //Check for same passwords
    if(usrPass != usrRepeat)
    {
        //Display Alert
        displayalert("Passwords do not match")
        return;
    }

    //Store Data
    UserDefaults.standard.set(usrEm, forKey: "usrEm");
    UserDefaults.standard.set(usrPass, forKey: "usrPass");
    UserDefaults.standard.synchronize();

    //Display alert message with confirmation
    let myAlert = UIAlertController(title: "Alert", message: "Registration is
successful. Thank You!", preferredStyle: UIAlertControllerStyle.alert);

    let okAction = UIAlertAction(title: "Ok", style: UIAlertActionStyle.default)
        {
            action in self.dismiss(animated: true, completion: nil);
    }
```

```
        myAlert.addAction(okAction);
      self.present(myAlert, animated: true, completion: nil);
}

 func displayalert(_ usrMessage: String)
    {
        let myAlert = UIAlertController(title: "Alert", message: usrMessage,
    preferredStyle: UIAlertControllerStyle.alert);
        let okAction = UIAlertAction(title: "Ok", style: UIAlertActionStyle.default,
    handler:nil);
        myAlert.addAction(okAction);
        self.present(myAlert, animated: true, completion: nil);
    }

    @IBAction func hasLogin(_ sender: AnyObject) {
        let storyboard = UIStoryboard(name: "Main", bundle: nil)
        let loginVC = storyboard.instantiateViewController(withIdentifier: "LoginVC")
        let appDelegate = UIApplication.shared.delegate as! AppDelegate
        appDelegate.window?.rootViewController = loginVC
    }

 override func viewDidLoad() {
        super.viewDidLoad()
        self.userEmail.delegate = self
        self.userPassword.delegate = self
        self.passRepeat.delegate = self
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
    }

    func textFieldShouldReturn(_ textField: UITextField) -> Bool {
        textField.resignFirstResponder()
        return true
    }
}
```

A.12 ThirdViewController.swift

```
import UIKit
var et: String = " "
class ThirdViewController: UIViewController {

    @IBOutlet weak var mrngtime: UILabel!
    @IBOutlet weak var ntDatePicker: UIDatePicker!
    @IBOutlet weak var ntLabel: UILabel!
    var starttime: String = " "
    var myreading = BipolarDisorder()
    let currentDate = NSDate()
    @IBAction func nextAction(_ sender: Any) {
         performSegue(withIdentifier: "next", sender: self)
[self.navigationController?.popViewController(animated: true)]
       }

    @IBAction func ntDatePicker(_ sender: Any) {
        var dateFormatter = DateFormatter()
        dateFormatter.dateFormat = "HH:mm"
        var strDate = dateFormatter.string(from: ntDatePicker.date)
        self.ntLabel.text = strDate
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        ntDatePicker.datePickerMode = UIDatePickerMode.time
        //ntDatePicker.minimumDate = currentDate as Date
        ntDatePicker.date = currentDate as Date
        let tbc = self.tabBarController  as! BDTabController
        myreading = tbc.myreading
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
    }

    override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
        var secondvc = segue.destination as! VC
```

```
        secondvc.start = ntLabel.text!
    }

    @IBAction func savest(_ sender: Any) {
        starttime = ntLabel.text!
        myreading.nsleep = starttime
        myreading.msleep = et
        mrngtime.text = et
    }

    override func viewWillAppear(_ animated: Bool) {
        super.viewWillAppear(animated)
    }
}
```

A.13 ForthViewController.swift

```
import UIKit
class ForthViewController: UIViewController, UITextFieldDelegate {

    @IBOutlet weak var userEmail: UITextField!
    @IBOutlet weak var userPassword: UITextField!

    override func viewDidLoad() {
        super.viewDidLoad()
        self.userEmail.delegate = self
        self.userPassword.delegate = self
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
    }

    func textFieldShouldReturn(_ textField: UITextField) -> Bool {
        textField.resignFirstResponder()
        return true
    }
```

```
@IBAction func loginButton(_ sender: AnyObject) {

    let usrEm = userEmail.text;
    let usrPass = userPassword.text;

    let userEmailStored = UserDefaults.standard.string(forKey: "usrEm");
    let userPasswordStored = UserDefaults.standard.string(forKey: "usrPass");

    if ((usrEm?.isEmpty)! || (usrPass?.isEmpty)!)
    {
        //Display Alert
        displayalert("All fields are required")
        return;
    }

    //Check for same passwords
    if(userEmailStored != usrEm || userPasswordStored != usrPass)
    {
        //Display Alert
        displayalert("Credentials do not match")
        return;
    }

    if(userEmailStored == usrEm)
    {
        if(userPasswordStored == usrPass)
        {
            //Login is Successful
            UserDefaults.standard.set(true, forKey: "isUserLoggedIn");
            UserDefaults.standard.synchronize();
            let myAlert = UIAlertController(title: "Alert", message: "Login is
successful. Thank You!", preferredStyle: UIAlertControllerStyle.alert);
            let okAction = UIAlertAction(title: "Ok", style: UIAlertActionStyle.
default)
                {
                    action in self.dismiss(animated: true, completion: nil);
            }
            myAlert.addAction(okAction);
            self.present(myAlert, animated: true, completion: nil);
```

```
                let storyboard = UIStoryboard(name: "Main", bundle: nil)
                let tabBarController = storyboard.instantiateViewController(withIdentifier
    : "TabBarController") as!UITabBarController
                let appDelegate = UIApplication.shared.delegate as! AppDelegate
                appDelegate.window?.rootViewController = tabBarController
            }
        }
    }

    func displayalert(_ usrMessage: String)
    {
       let myAlert = UIAlertController(title: "Alert", message: usrMessage, preferredStyle
    : UIAlertControllerStyle.alert);
       let okAction = UIAlertAction(title: "Ok", style: UIAlertActionStyle.default,
    handler:nil);
       myAlert.addAction(okAction);
        self.present(myAlert, animated: true, completion: nil);
    }
}
```