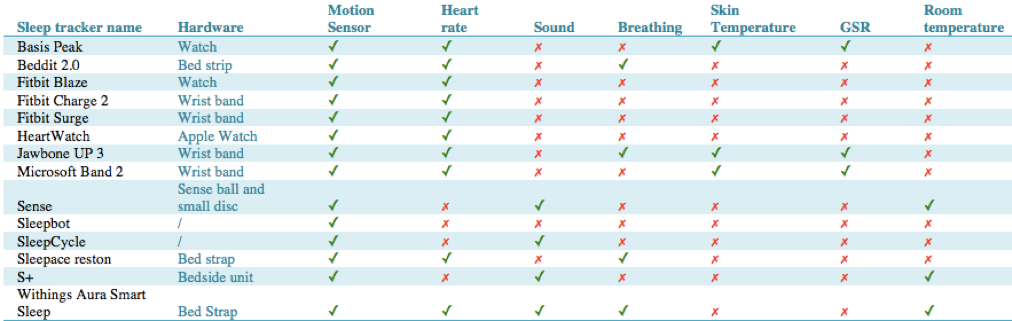
**

*B. Related work*

There are various mobile health care applications in the market that aim to record and improve the sleeping quality of the user. To classify the different sleeping stages of a user, data on the heart rate and body movement are essential. Figure 1 shows the comparison between different existing sleep trackers. Most of these sleeping applications use accelerometer and heart rate sensor to differentiate between different sleeping stages. Below highlights some interesting features of different sleep trackers.

Sleepbot[4] and Sleep cycle[5] are both mobile application that use sound in addition to the motion sensor to track sleep of a user. They have a smart alarm function to wake the user up during light sleep to reduce drowsiness in the morning. Since the phone has to be placed on the bed for motion sensing, there are reviews pointing out that the phone has warmed up over the night.

Sense[9] is a sleep tracker with a ball and a small disc. The ball is placed beside the bed to sense the temperature, humidity, ambient light, sound and air quality. It also has a speaker to play audio for the Smart alarm to wake the user up during 30-min wake up period when the user is in light sleep. The small disc is placed under the pillow with an accelerometer to detect the body movement during sleep. These are wireless

and Bluetooth low energy. The only drawback in it is it does not have a snooze function in its alarm.

S+[10] By ResMed also includes a bedside unit to monitor the sleep. It has a web-based app and uses bio-motion, which is a sonar technology that uses ultra low power radio wave and its reflection to track the users breathing and motion to analyze their sleeping pattern. It also detects light, noise and temperature in the room to differentiate sources that cause sleep disruption. The app would provide personal feedback based on the data collected.

Basis Peak[11] is a smart watch that can also track sleep using accelerometer, heart rate and GSR. However, it does not have coaching information on the sleep data collected.

Taking into account the features, pros and cons of these sleep trackers, the objective of our project is refined. Choices of different software and hardware components of Sleepify are made, which will be explained in Part III of this report.

III. IMPLEMENTATION

*A. Mobile application and choice of hardware*

While most of these mobile applications aims to improve the sleeping quality of the user by tracking and providing feedback, none of them tries to improve the sleeping envi- ronment of the user actively. Sleepify is a system that would track the sleeping quality of the user through heart rate, skin temperature, accelerometer and galvanic skin response, and provide feedback on the optimal room temperature and adjusts the heater automatically for the user.

Taking into considerations of the pros and cons of different sleep trackers, we have decided to use a fitness band rather than the mobile phone or a bedside unit to track the persons sleep. Not only does it provide more information than using the phone itself, such as data on heart rate, skin temperature and GSR, it also prevents the problem of phone overheating during charging at night.

There are various fitness bands in the market, we have decided to use Microsoft band 2 as our activity tracker. As seen in Figure 1, when compared to various fitness bands in the market, what it stands out is that it has a sensor for skin temperature and a SDK to facilitate self-development work for our mobile application. Although the SDK is not available on the official website anymore, the SDK is still available on github.

***/////***

***Choice of software and Homekit hardware***

We have chosen to develop our software using the Apple iOS platform as it is known to be more straight forward and easier to debug when compared to the Android one. Therefore, our programming language will be in *Swift 3*.

The iOS platform also has the added advantage of having a *HomeKit* framework, which accommodates the use and incorporation of other *HomeKit* compatible devices into our system. *HomeKit* allows third-party access to the home configuration database, to display, edit the accessories, and perform actions. Please note that there is a *HomeKit* Framework for developing *HomeKit* related products. [1] *Homekit*’s developer guide is available on the Apple official website, which includes the sample codes on application development. [2] They are, however, not yet updated to *Swift 3*. OOPer on github [3] has converted Apples sample code from *Swift 2.3* to *Swift 3*. These codes not only provide a basic framework on the linkage to the *Homekit* products, they are also used in our mobile application.

As the aim of the project is to change the room temperature, we have added units to control this. An ‘additional temperature sensor’ and an ‘on-off smart plug’ are needed to measure the room temperature, and to provide feedback to control the heater. We have selected *HomeKit* compatible devices, i.e. the *Elgato Eve Room* and the *Elgato Eve Energy* for easier collaboration. In addition to the measurement of room-temperature, the *Elgato Eve Room* is a wireless indoor sensor for measuring air quality and humidity; while the *Elgato Eve Energy* functions as a smart plug.

***Front-end development - user interface***

Most of the mobile apps such as Sleepbot[4] and Sleepcycle[5] only provides an interface that allows the user to only read data on sleep quality. Our mobile application aims to provide a simple and intuitive interface for the users to control the various hardware that are linked to the system, as well as to read the important data, such as target temperature and the quality of sleep. It has a minimalistic design with a login page and four other main pages as follows:

*(1) Login:* The user would first be prompted to the login page of the mobile application upon launch. Buttons are also present to navigate to the registration page and the ‘forget password’ page for the user to register an account without having the need to access the website. Upon successful logging-in, the user will be prompted to a tab bar controller, where a study has shown that 80% of the participants would prefer a tab view over a list view. [6] The tab bar controller consists of four tabs – a control tab, a reminder tab, a data tab and a logout tab.

*(2) Control of HomeKit products*: The control page consists of a table view where the database of the configuration of *HomeKit* products is automatically loaded. It is in a hierarchy structure, where the list of homes is first listed, the user can choose to view the different accessories in each of the homes, and choose to show the configurations of each accessory. This “top-down” hierarchy has the advantage of reducing distraction or information overload to the user. [7] The accessories listed includes the *Eve Room* and *Eve Energy* that are being used in the temperature control module of our product. Also, the user can perform actions easier on the different products – he/she no longer has to leave the application to view or change the configurations at the ‘Home’ page.

*(3) Data collection*: The data page is the main page of the mobile application. It is where the user can send the data unto the server, read the real time room temperature, body temperature and the target body temperature for sleep optimization. The user interface will have a ‘start’ and a ‘stop’ button for the user to commence recording data from the *Microsoft* band and *Eve Room* when about to sleep, and to end when the user wakes up. Throughout the period when the user is asleep, data will be uploaded to the web server every 10 minutes in a *json* format, to classify the sleeping quality using machine learning. Feedback of the target body temperature and sleeping quality will be sent back to the mobile application in real time and displayed on the user interface. The output textbox at the bottom of the screen is essential to keep the user informed of what is happening during the data collection process. [8] In case the user does not want to follow the target body temperature generated by the machine learning algorithm, there is an ‘override’ button for the user to set his/her own desired target room temperature intuitively using the circular slider on the top of the page. This is an alternate input mechanism which allows the user to have a direct manipulation of the value and can also increase the efficiency when compared to using a text input [8]. Lastly, a simple switch is present for the user to input whether he/she has had a good night, for further machine learning purposes.

*(4) Calendar integration:* The reminder page is an override function in addition to the automatic calendar integration. This is more convenient as it allows the user to set a reminder to sleep early. There is a ‘date-picker’ in the middle to input the date and time where the user wishes the reminder to be set. The reminder content can be entered in the text field below the ‘date-picker’. A reminder can then be set easily by clicking the ‘set reminder’ button.

*(5) Logout:* The logout page has a ‘logout’ button. Upon successful logging-out, the user will be prompted back to the login page, which is where the application was first launched.

[1] HomeKit Framework, *Apple Developer*. Available from: https://developer.apple.com/reference/homekit

[2] HomeKit Developer Guide, *Apple Developer.* Available from: https://developer.apple.com/library/content/documentation/Networking Internet/Conceptual/HomeKitDeveloperGuide/Introduction/Introduction.html

[3] OOPer Github, 3rd ed. Available from: https://github.com/ooper-shlab/HMCatalog-Swift3

[4] SLeepbot, *Jane Chu, Edison Wang* Available from: http://mysleepbot.com/about [Accessed 23th Feb 2017]

[5] SleepCycle, *Northcube AB* Available from: https://www.sleepcycle.com [Accessed 23th Feb 2017]

[6] Evaluation of User Interface Design and Input Methods for Applications on Mobile Touch Screen Devices http://download.springer.com/static/pdf/515/chp%253A10.1007%252F978-3-642-03655-2\_30.pdf?originUrl=http%3A%2F%2Flink.springer.com%2Fchapter%2F10.1007%2F978-3-642-03655-2\_30&token2=exp=1490048183~acl=%2Fstatic%2Fpdf%2F515%2Fchp%25253A10.1007%25252F978-3-642-03655-2\_30.pdf%3ForiginUrl%3Dhttp%253A%252F%252Flink.springer.com%252Fchapter%252F10.1007%252F978-3-642-03655-2\_30\*~hmac=97ab87748929ef587dada6a4be977857b0df004430f98f17132aaa914aae8d8f

[7] GUIDELINES FOR HANDHELD MOBILE DEVICE INTERFACE DESIGN Jun Gong 1 Peter Tarasewich https://personal.cis.strath.ac.uk/sotirios.terzis/classes/52.504/2010/GuidelinesGongTarase.pdf

[8] Design patterns for user interface for mobile applications <http://dx.doi.org/10.1016/j.advengsoft.2009.01.017>

[9] Sense, *Hello*, 3rd ed. Available from: https://hello.is/ [Accessed 23th Feb 2017]

[10] S+, *ResMed* Feb 2017]

Available from: https://sleep.mysplus.com

[11] Basis Peak, *Basis*. Available from: http://www.mybasis.com