Sleepify: A system towards personalizing and optimising sleeping environments

Jeremy Chan, Tsz Ho Ho, Dominic Kwok, Ho Shun Lo, Nathalie Wong
Department of Electrical and Electronic Engineering, Imperial College London, SW7 2AZ
Email: {jc4913, thh13, cyk113, hsl113, nw813}@ic.ac.uk

Abstract*— Given the rapid adoption of smartphones, mobile healthcare is an area which deals with acquiring, transporting, storing, processing and securing data from sensors or the phone to improve the health of a user. This project aims to investigate the relationship between room temperature and human sleeping quality by developing a mobile healthcare system to improve sleeping quality based on the current body and room temperature of the user. Sleepify utilizes advanced machine learning techniques to classify the user's sleep quality; this data is then fed to HomeKit compatible devices to control the heating of the user during their sleep. Both an app and web interface are offered. Initial testing and evaluation show positive feedback.

I. Introduction

Good sleep is important for both the physical and psychological health of a person. For example, sleep aids in the healing and repair of the blood vessels and heart. Studies have shown that sleep deficiency has been linked to an increased risk of stroke, high blood pressure, heart disease and diabetes [1]. The mechanism of regulating sleep is complex; there are many factors which affect sleep quality, such as the psychology of a person. In addition, the thermal environment is a key determinant in achieving good quality sleep [2]. Furthermore, disturbed sleep affects not only physical and psychological health status, but also mortality rates in the elderly [3]. This mechanism is also controlled by sleep regulation and circadian rhythm. These findings indicate that maintaining a comfortable thermal sleep environment is important for a healthy life. Several other works have also investigated on the effects of room temperature on sleeping pattern in human [2], [4].

These findings are our motivation for creating a product that improves sleep quality by monitoring vital physiological data and sleep environment of the user, and applying machine learning on these data to return an optimal and personalized room temperature.

II. Sleepify's Promise

This project aims to provide a better sleeping experience overall from having the room temperature automatically adjust to body and room temperature information. Sleepify also promises improved performance and improved machine learning classification accuracy based on prolonged usage of the app. Continued usage of Sleepify is especially important for our machine learning algorithms; thankfully the retention rate of health and fitness apps are the highest among others [5]. Finally, Sleepify promises to deliver a slick and intuitive app, and web interface for the user to use and interact with, motivating the user to continue using Sleepify regularly; this is crucial to having a low app abandonment rate [6].

III. Background

A. Sleep

Sleeping refers to a periodic physical state, featured by diminished consciousness, sensory activity, voluntary muscle control, and interactions with surroundings. The whole process can be divided into 4 distinctive stages: Three stages in Non-Rapid Eye Movement(NREM) [7] and one in Rapid Eye Movement(REM).

First, the process begins with stage 1 of NREM (NREM1) whereby the subject's eye movements are slowed down, before coming to a complete stop in NREM2. The subject then enters NREM3, which is the deep sleep stage. Finally, the subject moves on into the REM stage, in which muscles are paralyzed and biometrics such as heart rate and body temperature are uncontrolled. The sleep cycle ends with awakening.

B. Sleep and thermoregulation

The sleep-wake rhythm is strongly correlated with the circadian rhythm of the core body temperature (Tcore). Core body temperature decreases upon the onset of sleep due to the circadian rhythm; sleep further enhances this effect by keeping Tcore low[8] . The fundamental driving force behind this decrease in Tcore is due to the peripheral skin temperature. Vasodilatation near the peripheral skin allows rapid decreases in Tcore and promotes onset of sleep [4], [8]–[10]. Studies have concluded that elevated room temperature does degrade sleeping quality [11], [12], as sleep and decreases in skin temperature are related to cardiac activity, it has been suggested that the use of heart rate variability (HRV), skin temperature and galvanic skin response (GSR) can infer to the different stages of sleep and indeed this is how wearable such as Fitbit, and Jawbones detect sleeping patterns [13, p. 3].

IV. Related Work

A. Sleep Quality Evaluation

There are three widely used methods in clinical sleep quality assessment: Pittsburgh Sleep Quality Index (PSQI), Polysomnography, and Actigraphy [14]. First, PSQI is a questionnaire-based assessment focusing on subjective feedback on medium to long-term sleep quality [15]. Based on subjects' answers, it generates a score that is inversely proportional to sleep quality (lower is better). Due to the limitation of a long assessment interval, PSQI is not suitable for our real-time implementation. However, this method can, and is, used to evaluate the general performance of our system on sleep quality enhancement; the result of which will be discussed in VI.

Polysomnography analyses sleep quality by using electroencephalograms (EEG), electro-oculograms (EOG) and electromyograms (EMG) of the mentalis and libs [16]. It reflects

Sleep tracker name	Hardware	Motion Sensor	Heart rate	Sound	Breathing	Skin Temperature	GSR	Room temperature
Basis Peak	Watch	✓	✓	✗	✗	✓	✓	✗
Beddit 2.0	Bed strip	✓	✓	✗	✓	✗	✗	✗
Fitbit Blaze	Watch	✓	✓	✗	✗	✗	✗	✗
Fitbit Charge 2	Wrist band	✓	✓	✗	✗	✗	✗	✗
Fitbit Surge	Wrist band	✓	✓	✗	✗	✗	✗	✗
HeartWatch	Apple Watch	✓	✓	✗	✗	✗	✗	✗
Jawbone UP 3	Wrist band	✓	✓	✗	✓	✓	✓	✗
Microsoft Band 2	Wrist band	✓	✓	✗	✗	✓	✓	✗
Sense	Sense ball and small disc	✓	✗	✓	✗	✗	✗	✓
Sleepbot	/	✓	✗	✗	✗	✗	✗	✗
SleepCycle	/	✓	✗	✓	✗	✗	✗	✗
Sleepace reston	Bed strap	✓	✓	✗	✓	✗	✗	✗
S+	Bedside unit	✓	✗	✓	✗	✗	✗	✓
Withings Aura Smart Sleep	Bed Strap	✓	✓	✓	✓	✗	✗	✓

Figure 1: Sensor comparison

the precise proportion of each sleep stage during a 24-hours assessment interval, and hence provides the most accurate sleep quality evaluation. Despite its accuracy, it has a few critical disadvantages that prevent its application in our system. First, the sensors required are extremely intrusive to user (located around the head) and all signals require intensive processing algorithms to analyse. Secondly, the data collection process for complete analysis require at least 12 hours, making it not applicable in Sleepify's case.

Actigraphy monitors sleep quality by estimating the ratio between 'sleep' and 'awake' patterns. Conventionally, 'sleep' and 'awake' patterns are defined as minor and intense body movements during sleep by using a motion sensing device known as an actometer. The core principle is that body muscles are completely paralyzed during deep sleep stages but not in others. By extending this principle along with redefining sleep-awake patterns and combining more sensors, several actigraphy sleep quality evaluation methods have been invented. Furthermore, research by Ya-Ti Peng et.al has also shown that introducing heart rate data into normal motion tracking can improve sleep-awake pattern classification [17]. These applications act as a proof of concept for actigraphy validity, as well as the correlation between sleep quality and biometrics including body movement, heart rate, etc. Moreover, they demonstrate the method's compatibility with a mobile phone. Thus, we decided to utilise actigraphy as our detection principle. We additionally leverage machine learning for the implementation to provide short-term sleep quality evaluation, continuously.

B. Commercial Products

Mobile applications such as iSleep [18], Sleep as Android [19] and Toss 'N' Turn [20] use the mobile phone as the main sensor to collect data about sleeping noises, body movement, and background light intensity. They determine the sleep-awake ratio in each night to evaluate sleep quality with a mean accuracy over 80%. Sense [21] is a sleep tracke r 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. The small disc is placed under the pillow with an accelerometer to detect the body movement during sleep. These use low energy wireless Bluetooth connections. Finally, Z3 by a group of engineers at Imperial College, London, track light intensity to figure out sleep status from sunlight.

S+ By ResMed also includes a bedside unit to monitor the sleep [22]. 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 diffe rentiate sources that cause sleep disruption. The app would provide personal feedback based on the data collected. Basis Peak 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 [23].

From these products, Sleepify aims to fill the gap of real-time environmental changes based on user feedback. There is no product available on the market that changes heating based on sleep quality – Sleepify fits that niche.

V. System Design and Implementation

A. Overall High Level Design

Heredia et al. shows that traditionally, users pay a subscription fee instead of an upfront fee for software; however the Software-as-a-Service (SAA) model guarantees that the user will always be using the most updated version of the software as there is no 'local copy' of the software to install [24]. Following the SAAS model, Sleepify consists of a front-end and a back-end, each consisting of two parts. The front-end is what users see and use, and consists of an iOS application and the web interface. Updates through the App Store and the website ensure the user will always be using the most updated version of Sleepify. Finally, this front-end connects to the sensors for data collection and temperature adjustment. The back-end consists of the servers, databases, APIs, and machine learning modules – these both provide, and accept information from the front-end applications. The user has no information or control on how the back-end is configured; they need not to.

B. Sensors

Based on our findings from both academic and commercial sources stated in related works, our chosen wearable should provide physiological signals such as heart rate, rr-interval, skin temperature and GSR. The Data-as-a-service platform for healthy lifestyle and preventive medicine (DAPENE) has provided a comprehensive literature on existing wearable technology, and also from Figure 1, it is clear that only products from Fitbit, Jawbone and Microsoft contain the above sensors

[25]. For example, the Apple Watch does not have GSR, and its heartrate measurements have been known to have inconsistencies based on customer reports. Since the release of the first Microsoft Band, researchers have been using it for various works in different fields such as sleep tracking [26]. It can be see n from **Error! Reference source not found.** that Microsoft Band 2 provides an extensive range of suitable sensors and APIs, and therefore the Microsoft band 2 was chosen.

C. Backend (Server, Database, API)

The backend is responsible for interfacing with the front-end, in accepting and providing it with the information it needs. It consists of a server on which a database resides, and an API which allows the iOS app, web interface, and machine learning sections to communicate with the server and by extension, the database.

1) The Server

As the d atabase, the web interface, and the API all reside upon the server, a smart choice needed to be made regarding how the server should be implemented.

Our group had prior experience in setting up a server running a LAMP stack (Linux, Apache, MySQL, PHP) from scratch, but this was judged to be inadequate for Sleepify as trying to hand code PHP without a web framework when creating any sort of standard compliant web app would take an extremely long time. Laravel, and Yii, both modern PHP frameworks, were initially shortlisted as usable frameworks. However, the verbose and sometimes confusing syntax of PHP mean getting things done is emphasised more than code readability [27]. As Sleepify's development may continue in the future, reusability and code readability meant the group decided not to go with a PHP framework. Further research from Srinivasan et al. also showed PHP to suffer from more security issues compared to other web frameworks [28]. Therefore, Sleepify needed a framework with support for security features such as HTTPS, SQL injection, and Cross Site Request Forgery (CSRF), all part of the top 10 application security risks as defined by the Open Web Application Security Project (OWASP) [29].

Emphasising code readability, rapid development, and security features meant the choice was narrowed down to two frameworks in two programming languages: 'Django' in Python, and 'Ruby on Rails' in Ruby. Both offer extremely fast prototyping and development, extensive documentation, security measures against common attacks, and multiple libraries to assist development. The final decision was to use Django, the Python web framework, as the ease of use of Python (smaller learning curve compared to Ruby) and the ample documentation on Django, along the active community meant decreasing the time needed to create the MVP.

Multiple Django libraries were leveraged to add extra functionality, the most notable being: django-rest-framework, a library which provides the skeleton of the API; django-rest-auth, which extends Django's already excellent authentication system with the API; django-bootstrap3, a library which simplifies styling a website using Twitter bootstrap.

2) The Database

The main design decision when choosing the database was whether to go with a Structured Query Language (SQL) or a NoSQL database.

SQL databases are known as relational databases, where databases are linked together by keys and values [30], held in entries in database tables. In SQL databases, all the incoming data must match the format of the database table, whilst NoSQL operate on the premise that the incoming data is of a large volume and of a rapidly changing format [31].

The most well-known NoSQL database is MongoDB, and it offers several advantages over SQL databases. MongoDB claims scalability and performance improvements in [32], claiming that NoSQL databases are horizontally scalable (add more servers) instead of vertically scalable (have to make the one server more powerful). However, the flexibility of NoSQL data means there exists consistency issues when dealing with many similar data objects – unacceptable for user data. Nayak et al. compares NoSQL's data formats, showing the data being held in a binary Javascript Object Notation (JSON) object [33], which allows it to be accessed using object oriented methods. However, this advantage is nullified with Django as it has its own object oriented wrapper for any type of database. Django supports its own 'Models', which abstract away the complicated SQL statements needed to modify the database [34] in favour of treating database tables as objects, nullifying yet another advantage of NoSQL databases.

Hence, the final decision was to use SQL databases. Having a SQL database means structured, and relational relationships mean data can be linked with user profiles very easily. There are a few popular SQL databases, the most popular 3 being SQLite, MySQL, and PostgreSQL. From [35], the pros and cons were evaluated; the final decision was made to use SQLite, a SQL database that comes shipped with Django by default, with the main justification coming from portability (copy and paste the database across testing machines, committable on Git), and it supports enough features to not be considered bloated. Scalability issues have been moved down in priority as according to SQLite, they only occur at high volumes of data [36], an unrealistic target for Sleepify. Lastly, NoSQL support is not part of the official Django development effort, and is only supported via third party forks [37].

With regards to user security, the SQLite database is currently not encrypted as the database is un-reachable through the internet. However, in the future, a Django library known as django-fernet-fields can be utilised to encrypt database fields.

3) The API

Creating an API was a top priority for the back-end as it enables a consistent communication format between the front and back-ends. Sleepify's API exposes URLs in which data can be sent or retrieved, including but not limited to the following: machine learning results, raw sensor data, graphs, calendar events, and sending push notifications.

To create the API, a communication format and architectural style had to be decided. Nurseitov et al. compares the two main communication formats, eXtensible Markup Language (XML), and JSON [38]. XML follows a rigid pre-defined structure while JSON does not have any pre-defined structures, so initially it seemed XML was the way forward as health data follows a pre-defined format. However, since everything in XML is stored in strings, parsing the XML data takes relatively more processing power than that of JSON, which can have single entries or arrays of strings or integers – making JSON much more efficient, especially on mobile platforms as demonstrated by Sumaray et al. [39], making it Sleepify's choice for the data format.

To decide on the architectural style, the pros and cons of Simple Object Access Protocol (SOAP), Representational State Transfer (REST), and Remote Procedure Call (RPC) were compared based off information from [40], [41], [42].

As SOAP relied on XML, it was not chosen. Based on these results, Sleepify chose to use a mixture of REST and RPC architectures as the main API functions consist of both data management and functional commands. Data retrieval and insertion would be done using RESTful nouns such as `/user/`, `/raw_data/`, `/stats/`, while push notifications and the machine learning training would be done using RPC verbs such as `/push/`, `/migrate_features/`. This gives a good compromise between the uniformity and URL design of REST, whilst keeping the flexible functionality of RPC [43], [44].

Authentication to the API is done through sessions/cookies, and is supported through defining permission classes in the API functions (e.g. statistics only available to logged in users, user registration, log in/out, open to the public). Another style of authentication is using JSON Web Tokens (JWT), but JWT does not allow pushing notifications to logged in clients as there is no way to know whether a user is logged in or not, as opposed to authenticating using sessions [45].

D. Machine Learning

1) Clustering Analysis and Features Extraction

Given the sensors provided by Microsoft Band 2, the features generated covers 3 modalities, summarized in Table 1. As sensor data is time based, a time interval for feature generation is required. Our final windowing size is 10 minutes of data sampled at 1Hz. This decision provided the best compromise between mobile phone hardware capabilities, feature validity, and data usage, giving around 2MB's worth of data per night of sleep. We justified this by observing that the typical range of sleep stage transition time varied from 7 to 45 minutes. Thus, it would be optimal to generate features every 45 minutes. However, the window width is constrained by the memory available on mobile phone. Storing 7 sensors' readings at 1Hz for 45 minutes exceeds the available memory and it also causes long delays in the sending of the data to the server. Therefore, a smaller, 10-minute interval was chosen; it also proved to be effective in Toss 'N' Turn [20].

Features for each modality are chosen by referencing existing signal processing techniques. Previous research has shown that the exposure to extreme heat and a humid environment can affect sleep quality – Sleepify uses the mean skin temperature to capture this exposure effect on the core body temperature [46]. Moreover, as temperature tends to decrease at the night-time sleep onset but increase when awake [47], the standard deviation can be used to capture the fluctuation of body temperature within each time window. The mean and standard deviation of skin temperature were extracted.

Accelerometer readings are used to reflect users' arm movement. Instead of using the interval average acceleration over time suggested in [48], data are extracted in terms of mean and standard deviation of squared amplitude shown in Equation 1.

Equation 1: $\sqrt{X^2 + Y^2 + Z^2}$

The motivation behind this is to capture movement information every second. The suggested method is effective when the time interval is around 100 seconds, which is much smaller than the chosen 10 minutes' interval. If a similar approach is used, the excessive smoothing on 10 minutes' worth of data can remove acute magnitude fluctuations caused by sudden arm movements. Therefore, the mean and standard deviations on the mean squared amplitude are used to extract overall movement intensity and frequency.

Heart rate variability is proved to be higher in rapid eye movement sleep stage than others [49] while mean and standard deviation on RR intervals are shown to be adequate measures of HRV during sleep stages transition [50]. Additionally, instantaneous heart rate data is also analysed in a similar manner. Apart from using the mean and standard deviation, kurtosis is also used to analyse the extremes. From Moors' paper [51], kurtosis is a measure on outlier's population out of the total samples. The higher the kurtosis is, the less distributed the sample is around the statistical median. Hence, it can be a useful tool to quantify the chronic changes in RR interval and heart rate that is removed under mean and standard deviation. Lastly, we have excluded the introduced Galvanic Skin Response reading as input features because of three reasons. First, the sweat production is proven to be independent with galvanic skin response amplitude [52]. Secondly, the galvanic skin response (GSR) sensor required the band be in a calibrated and 'locked' mode to provide accurate data; this is controlled independently by the Microsoft Band 2. Additionally, from experimentation, the band must be firmly attached to skin for a reliable reading. Finally, the variance of GSR across different sleep quality is found to be nearly zero from data collected. This demonstrates its insignificance in sleep quality evaluation, and hence GSR is excluded in the final feature set.

Table 1: Initial Feature Set

Modality	Sensor Data	Features
Temperature	Temperature readings in Celsius	Mean, standard deviation (STD)
Movement	3-axes Accelerometer readings	Mean, STD of root mean squared amplitude

Heart rate	Optical Heart Rate readings and RR interval	Mean, STD, Kurtosis

2) Model selection

We have defined our sleep quality evaluation problem as a binary classification problem after evaluating it from the view of various sleep-awake pattern classifiers [53]. Given the final feature set, it is necessary to select an optimal classifier based on the obtained data. MATLAB's classification learner is used to perform cross-model benchmarking. From previous testing results as shown in Table 2, a user specific classifier performs better than a unified classifier and hence our final model selection process only focus on optimising models which are trained on a specific subject. To prevent a loss of generality, we carefully selected a subject that has the most uniformly distributed sleep quality. The result is shown in Table 3.

Table 2: Feature Analysis Results

Models	All Data Accuracies	Personalized Data Accuracies
Best Tree	78.1%	90.5%
Logistic Regression	78.1%	85.7%
Best SVM	75%	95.2%
Best KNN	84.4%	90.5%
Best Random Forest	75%	95.2%
Boosted Tree	81.3%	85.7%
Subspace Discriminant	60%	100%

Table 3: Model Selection Results

Models	Personalized Data Accuracies
Decision Tree	95.8%
Logistic Regression	85.7%
Best SVM	96.4%
Best Random Forest	96.3%
Boosted Tree	95.6%

From Table 2, it can be observed that either support vector machine or random forest should be chosen as the implementation model. Apart from the model accuracy, training, and testing time are also considered as model selection criteria. Thus, random forest is chosen to be implemented due to its efficient training and testing principles. Another drawback from this testing result is that it is based on feature sets generated from one specific subject over 1 week due to limited resources such as time and available hardware. Moreover, the limited subject diversity also reduced the available sample size. This is because most of data collected are from good sleepers, which causes an imbalanced sleep quality distribution. Nevertheless, the sleep quality labelling on training data assumes that overall night sleep quality can be interpolated into individual interval sleep quality. This assumption should be abandoned if more time and resources are given to perform clinical testing on overnight sleep quality monitoring with the device.

4) Machine Learning Model – Server Deployment

To integrate the machine learning module into the system smoothly, an implementation of the random forest classifier is done on the server side to provide online estimation upon requests from Sleepify's clients.

As we have chosen Python-Django for the server development platform, `Scikit-Learn` [54] (a Python library) is used for our machine learning implementation. Model persistence on server is achieved by binary serialisation and recovering using `Pickle`, a Python object serialization tool. From the preliminary study results, it is necessary that each user requires a specific classification model which is trained by personalized data. Therefore, these binary models files are linked to user entries in the database to allow user specific mapping.

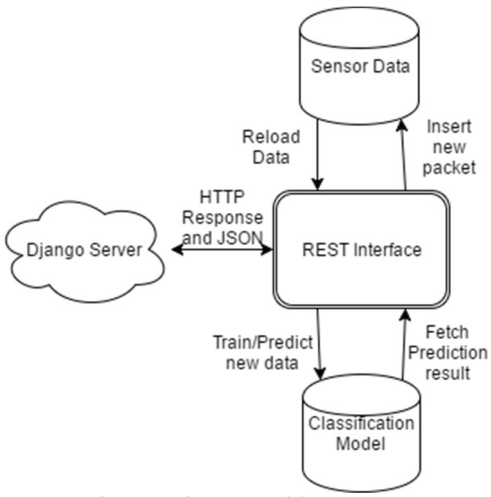

Figure 2: Server Architecture for ML model

To communicate with the mobile application, the Sleepify API is extended to offer three functions: sensor data storage, machine learning model retraining, and prediction. When the mobile application sends a packet of new sensor data to the server, this interface will first extract the date, time and user details of the packet and store the data into database under specific user accounts. After that, the packet will be pre-processed into a feature sample. Then, binary model files under the specific user are recovered, and prediction methods are called with the generated feature vector. Finally, the prediction outcome is sent back to mobile application in the form of a JSON object. Continuous learning is supported using existing models provided by `Scikit-Learn`, preventing the situation of having to retrain the model completely. A user specific data tracker is implemented to monitor the accumulated count of untrained data packets. When it exceeds a threshold, the model will be recovered from the binary file to perform online learning. The block diagram for this machine learning server infrastructure is shown in Figure 2.

5) Testing

To evaluate the model implementation, offline model testing is done to demonstrate the practical performance of random forest model provided by `Scikit-learn`. To compare with testing performed by the MATLAB classification learner, identical hyper-parameters such as tree depth and tree numbers are used. Moreover, a similar testing and training set splitting ratio is used - 0.2 and 0.8 respectively. The result shows that implemented classifier can achieve 90% accuracy, similar to the MATLAB implementation.

E. Frontend (iOS Application and Website)

1) The iOS application and Homekit

iOS has the added advantage of having the *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 [55]. However, *Homekit*'s developer sample code available on Apple's website are not yet updated to *Swift 3*. User 'OOPer' on Github [57] has converted Apple's sample code from *Swift 2.3* to *Swift 3*. This not only provides a basic framework on the linkage to the *Homekit* products, but are also used in our mobile application.

The HomeKit compatible device, Elgato Eve Room, is used to obtain the thermal, air quality, and humidity measurements of the room. As the aim of the project is to change the room temperature, Sleepify uses the Elgato Eve Energy smart switch. In the future, this can be implemented using other HomeKit compatible thermostats for more effective control of the room temperature.

2) iOS User Interface Design

Most mobile apps such as Sleepbot [58] and Sleepcycle [59] only provide an interface that allows users to view their sleep quality data. Our mobile application aims to provide a simple and clear interface so that it is intuitive enough for the users to learn to control the various hardware that are linked to the system, as well as to view the data after using it once. Therefore, it has a minimalistic design with a login page, and other main pages as follows:

(1) Login, Logout: The user would first be prompted to the login page of the mobile application upon launch. Navigations to the 'registration page' and the 'forget password' page are present for the user to register an account without having the need to access the website, enhancing the user experience when using our system. Upon successful logging-in, the user will be prompted to a tab bar controller consisting of four tabs – a control tab, a reminder tab, a data tab and a logout tab. Upon successful logging-out, the user will be prompted back to the login page, which is where the application was first launched.

(2) Control of HomeKit products: The control page consists of a table view where connected HomeKit devices, including the Eve energy and Eve room, are displayed. It is in a "top down" hierarchy structure, which has the advantage of reducing distraction or information overload to the user [60]. The list of homes is first listed, the user can choose to view the different accessories in each home and to change the configuration of each accessory.

(3) Data collection: The data page is the main page of the mobile application. It is where the user sends data to the server, view the real-time room and body temperature and the target body temperature for sleep optimisation.

The user can choose to start to commence data recording from the *Microsoft* band and *Eve Room* when the user is about to sleep, and to stop when the user wakes up. Throughout the period when the user is asleep, data will be uploaded to the web server every 10 minutes, to classify the sleeping quality using machine learning. Feedback from the server, consisting of the target body temperature and sleeping quality, are received in real time and displayed on the user interface. To show the user what's going on, an output console at the bottom of the screen is there to keep the user informed during the data collection process, increasing the user satisfaction with the system [61]. In case the user does not want to follow the target body temperature generated by the machine learning algorithm, the user can choose to override the desired room temperature using a circular slider, which has higher efficiency compared to other input mechanism[61]. This circular slider also increases the user interaction and make the user feel that he/she is in control which makes it more enjoyable. Lastly, the user can input whether he/she has had a good night, for record and further machine learning purposes.

3) Implementation of Communication

Stable and efficient communication between the mobile application and backend server is essential as a large amount of data is transmitted while the user is asleep. `Alamofire` is a HTTP networking library written in Swift for iOS and Mac OS X. The library has elegant connections to Apple's Foundation Common networking tasks [62]. To receive and transmit physiological data and output from the server, chainable request and response methods are used to de-serialise JSON in both the backend and frontend applications, given JSNO is the industrial standard approach in data transmission between servers, and the server's choice of data format (see server section for justification). The `NSDictionary` class in Swift is used to set key value pairs for the JSON data, as well as decode JSON received from the server.

4) Implementation of Security

To maintain usability and integrity of the network and data, unauthorized activities should be prohibited. Data encryption and user authentication are common methods to allow authorized user to perform human-to-machine interactions in order to access the connected systems and its resources [63].

Cross-site request forgery (CSRF) is an attack vector that allow unauthorized commands transmitted by sending arbitrary HTTP requests from a user that is trusted. CSRF vulnerabilities have been known since 2001 and many websites became victim of this type of attack [64]. There are several CSRF prevention techniques by embedding additional authentication data into requests to detect unauthorized usage. By implementing the CSRF token from the server in the header of all HTTP requests, similar to the implementation in [65], the server can verify that the sender is legitimate.

All data is currently sent in plaintext. For the meantime, this is fine as the server is hosted on a local PC, and not on a cloud hosting service (no chance of getting the database data without explicit authorisation). In the future, Sleepify aims to provide local encryption to data before it is sent to the server. The frameworks for AES256-CBC encryption have been set up, however they are pending verification with the server. Once the system is completed, the only remaining vector to get customer data is via the local app – Sleepify aims to solve this by leveraging Apple's on-board crypt0 engine to encrypt data files related to the app in the future.

5) Implementation of the Feedback System

As the users wake up in the morning, the mobile application will require them to provide feedback on their sleeping quality as good or bad. This allows the backend server to classify the physiological data and improve the client model. This model will be used to provide the optimal body temperature for the system to adjust the heating every ten minutes. If the user is experience bad sleep quality, this optimal body temperature will be compared with the user's current body temperature. The system will control the smart plug connected to the heater and turn it on/off depending on the difference in temperatures, and whether the override button is on. The flow of the system is shown as below:

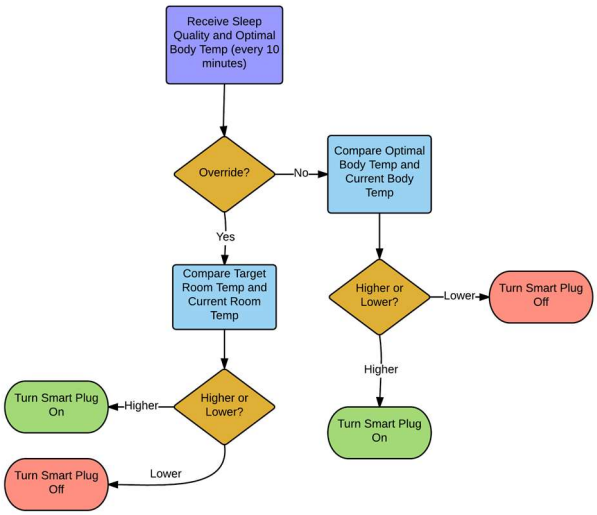

Figure 3: Flow of app

F. Web Interface

As per Sleepify's promise, the web interface should be modern, intuitive, and easy to use. The following three design principles were followed during development:

1. Compatibility with mobile devices
2. Use existing HTML styling frameworks to assist development, provide compatibility with mobile
3. Simple navigation buttons should provide directions to all parts of the website
4. Prioritise reader comfort and readability of text, use neutral colours in graphs and text

To tackle the compatibility with mobile problem, 1., responsive web design was a priority in the development of the web interface. By utilizing viewports (show different amounts of data based on the device width and display density) as described in [66], [67], a responsive web design was created as a product of extensive testing on different resolutions.

To tackle the framework problem of 2., Twitter Bootstrap was used as the library of choice as it is lightweight (only a few static files needed for setup), flexible (allows overriding of default styles with custom ones), and powerful. Bootstrap is also built for mobile first when compared to other styling frameworks such as Foundation and Skeleton [68]. Bootstrap gives predictable websites at the cost of slightly verbose HTML [69]. By following code styles and practices from Chapters 2-5 of [70], a navigation bar, a jumbotron (big heading type text at the top of every page to provide context), sidebars, and footers were created as the base template for Sleepify, thereby fulfilling 3.

No. 4 is quite subjective, however, Sleepify adhered to the design principles shown in Chapters 2,4,5 of [71], to create a flat, minimalistic, and stylish website, as opposed to other design styles such as skeuomorphism in Chapter 1 of [71]. Sleepify respects Bootstrap's grid system and whitespace, and utilises a modern font (Josefin Sans) along with short line lengths, to improve readability and usability [72].

VI. User Evaluation Criteria and Setup

To evaluate the effectiveness of Sleepify on users, the testing was split into sleep quality testing, and the user interface testing (app, web interface). Due to limited hardware, sleep quality testing was done on Sleepify group members, whilst the user interface testing was done on a group of 14 non-Sleepify users. Care was taken to ensure this group had a mixture of gender, technical background, and hardware (iOS versions, phone, PC).

A. Sleep Quality

To evaluate the improvement on sleep quality brought by Sleepify, we performed a controlled experiment with 3 subjects over 2 weeks. 3 subjects are selected to cover various sleeping habits and health; Subject 1 was a healthy sleeper with a fixed sleeping schedule, subject 2 had irregular sleeping periods but healthy sleeping behaviour, while subject 3 is a bad sleeper. To demonstrate sleep quality improvement, three subjects are invited to sleep overnight in a room with a static heating system maintained at 24 degree Celsius for one week. After that, their sleep qualities are assessed by using Pittsburgh Sleep Quality Index (PSQI). PSQI cover factors including sleeping duration, midnight behaviours, and subjective feedback to provide sleep scoring which is inversely proportional to sleep quality (lower is better, >5 is bad quality sleep). Then, the three subjects performed the same experiment with Sleepify running, connected to the EvoHome and a portable heater. The static heating systems are then disabled during this period. To ensure the optimal temperature profile is used during experiment, an optimal heater setting is collected with subject personally.

B. App, Web Interface

To evaluate Sleepify's app, the group of 14 non-Sleepify users was asked to rank the app out of 5 for the following questions:

1. Is the application easy to navigate? Is the control part of the application intuitive to use?
2. What do you think about the design?
3. Has it shown enough information that you want to know? (if bad – what else would you want to see?)

To evaluate the web interface, the following questions were used (Yes/No, comments on why):

1. Are the features of the website easy to find?
2. Do you feel in control of your personal data?
3. How is the visual appeal of the website?

VII. Results

A. Sleep Quality

The result showed that subject 1 and 3 experienced comfortlessness due to thermal environment. The result shown in Table 4 and Table 5 showed that subject 1 experienced a small improvement of sleep quality in general without any issue with thermal setting. Subject 2 had sleep quality slightly worse

than before but the thermal condition problem disappeared. However, although subject 3 still found that the room too cold, their sleep quality increased.

Table 4: Sleep quality before using Sleepify (all other PSQI scores hidden)

Before			
User	1	2	3
Felt too cold	0	2	5
Felt too hot	1	1	1
Overall PSQI Score	1.5	5	12

Table 5: Sleep quality after using Sleepify (all other PSQI scores hidden)

After			
User	1	2	3
Felt too cold	0	0	4
Felt too hot	1	0	1
Overall PSQI Score	1	5.5	8.5

B. App, Web Interface

100% of the respondents agree that the application is easy navigate, and intuitive enough to use and control without prior learning, praising the *'clear tab system at the bottom".* 93% of the respondents find the design visually appealing, especially the *'professional splash, login, and ring slider'*. However, one suggests that it would be better if users can customize their own colour scheme of different pages. 71% or the respondents are happy with the current information shown on the data page of the application. 4 of them pointed out that there could be more information shown to reflect the sleeping quality, for example adding graphs or data on the other various measurement such as heart rate would make it more interesting.

For the web interface, Sleepify achieved an average of 85.7% positive feedback. 100% of testers found the features of the website easy to find, and 93% found they website appealing to look at, praising the *'sleek look of the graphs'*, and the *'good mixture between visual aid and* text'. However, some constructive feedback was had, such as '*text too small from a distance'*, and '*homepage has too much left/right margin whitespace'*; these were taken into consideration in the final design. A big area of future improvement is the user's control of personal data. Only 65% of testers felt confident in their control over their personal data, citing *'a better dashboard to display data would be beneficial*', and *'there should be a big delete all button as opposed to deleting by date or single entries'*.

VIII. Discussion

This experimental result demonstrated that Sleepify can improve the thermal condition for sleeping to a certain extent. However, the result failed to show that it directly improved sleep quality and this phenomenon is under expectation since sleep quality is also affected by many other factors such as stress, food intake, exercise intensity and other numerous factors. There are too many uncontrollable factors for test subjects - nevertheless, there exists an unneglectable inconsistency in optimal thermal environment assurance for sleep as the sample size is too small to provide a general performance benchmark with similar product on market.

The app and web interface had generally positive comments, however the next iteration of the front-end should consider user feedback, especially regarding user data on both the web and app. The team notes that Sleepify already has this functionality in its API, and implementation should be straightforward.

IX. Conclusion

X. References

[1] K. N. Mims and D. Kirsch, 'Sleep and Stroke', *Sleep Med. Clin.*, vol. 11, no. 1, pp. 39–51, Mar. 2016.

[2] S. S. Gilbert, C. J. van den Heuvel, S. A. Ferguson, and D. Dawson, 'Thermoregulation as a sleep signalling system', *Sleep Med. Rev.*, vol. 8, no. 2, pp. 81–93, Apr. 2004.

[3] A. B. Neikrug and S. Ancoli-Israel, 'Sleep Disorders in the Older Adult – A Mini-Review', *Gerontology*, vol. 56, no. 2, pp. 181–189, Mar. 2010.

[4] K. Okamoto-Mizuno and K. Mizuno, 'Effects of thermal environment on sleep and circadian rhythm', *J. Physiol. Anthropol.*, vol. 31, no. 1, p. 14, May 2012.

[5] 'Enter the Matrix: App Retention and Engagement', *Flurry Blog*. [Online]. Available: http://flurrymobile.tumblr.com/post/144245637325/appmatrix. [Accessed: 19-Mar-2017].

[6] 'App Retention Improves - Apps Used Only Once Declines to 20%'. [Online]. Available: http://info.localytics.com/blog/app-retention-improves. [Accessed: 19-Mar-2017].

[7] H. Schulz, 'Rethinking Sleep Analysis', *J. Clin. Sleep Med. JCSM Off. Publ. Am. Acad. Sleep Med.*, vol. 4, no. 2, pp. 99–103, Apr. 2008.

[8] J. Barrett, L. Lack, and M. Morris, 'The sleep-evoked decrease of body temperature', *Sleep*, vol. 16, no. 2, pp. 93–99, Feb. 1993.

[9] K. Kräuchi, C. Cajochen, E. Werth, and A. Wirz-Justice, 'Functional link between distal vasodilation and sleep-onset latency?', *Am. J. Physiol. Regul. Integr. Comp. Physiol.*, vol. 278, no. 3, pp. R741-748, Mar. 2000.

[10] 'Acute finger temperature changes preceding sleep onsets over a 45-h period (PDF Download Available)', *ResearchGate*.

[11] E. J. W. Van Someren, 'Mechanisms and functions of coupling between sleep and temperature rhythms', *Prog. Brain Res.*, vol. 153, pp. 309–324, 2006.

[12] 'Effects of mild heat exposure on sleep stages and body temperature in older men', *ResearchGate*.

[13] 11.4.14 and J. Donahue, 'UP3: A deep dive into sleep tracking', *The Jawbone Blog*, 04-Nov-2014. .

[14] S. M. G. P. Togeiro, A. K. Smith, and R. B. P. S. P. Braz, 'Diagnostics methods for sleep disorders', *Suppl*, vol. 27, pp. 8–15, 1999.

[15] D. J. Buysse, C. F. R. T. H. Monk, S. R. Berman, and D. J. Kupfer, 'The Pittsburgh sleep quality index: A new instrument for psychiatric practice and research', *Psychiatry Res*, vol. 28, p. , 1989.

[16] R. B. Berry, R. Brooks, C. E. Gamaldo, S. M. Harding, C. L. Marcus, and B. V. Vaughn, 'The AASM manual for the scoring of sleep and associated events', *Rules Terminol. Tech. Specif. Darien Ill. Am. Acad. Sleep Med.*, 2012.

[17] Y. T. Peng, C. Y. Lin, M. T. Sun, and C. A. Landis, 'Multimodality Sensor System for Long-Term Sleep Quality Monitoring', *IEEE Trans. Biomed. Circuits Syst.*, vol. 1, no. 3, pp. 217–227, Sep. 2007.

[18] T. Hao, G. Xing, and G. Zhou, 'iSleep: unobtrusive sleep quality monitoring using smartphones', in *in Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, p. 4.

[19] U. Team, *Sleep as Android Unlock*. Urbandroid Team, 2016.

[20] J.-K. Min, A. Doryab, J. Wiese, S. Amini, J. Zimmerman, and J. I. Hong, 'Toss "n" turn: smartphone as sleep and sleep quality detector', 2014, pp. 477–486.

[21] 'Sense by Hello', *Hello*. [Online]. Available: https://hello.is/. [Accessed: 22-Mar-2017].

[22] 'S+ sleep monitoring device | ResMed.com'. [Online]. Available: http://www.resmed.com/us/en/consumer/s-plus.html. [Accessed: 22-Mar-2017].

[23] 'Basis | Basis'. [Online]. Available: https://www.mybasis.com/. [Accessed: 22-Mar-2017].

[24] A. Heredia, R. Colomo-Palacios, and A. de Amescua, 'Software Business Models from a Distribution Perspective: A Systematic Mapping Study', *Procedia Comput. Sci.*, vol. 64, pp. 395–402, Jan. 2015.

[25] Gonzalo Bailador del Pozo and Ignacio Mendizabal Vázquez (UPM), Carmen Sánchez Ávila (UPM), Javier Guerra Casanova (UPM), Miguel Arriaga Gómez (UPM), Lino García Morales, 'D4.1 State of the Art -Wearable Sensors'. Data-as-a-service platform for healthy lifestyle and preventive medicine.

[26] Jeff Huang, 'Extracting My Data from the Microsoft Band'. [Online]. Available: http://jeffhuang.com/extracting_my_data_from_the_microsoft_band.html.

[27] S. Das, 'Which is Better, PHP or Python? A Developer's Take', *LinkedIn Pulse*, 11-Jun-2015. [Online]. Available: https://www.linkedin.com/pulse/which-better-php-python-developers-take-srikrishna-das. [Accessed: 19-Mar-2017].

[28] S. M. Srinivasan and R. S. Sangwan, 'Web App Security: A Comparison and Categorization of Testing Frameworks', *IEEE Softw.*, vol. 34, no. 1, pp. 99–102, Jan. 2017.

[29] Open Web Application Security Project, 'OWASP Top 10 - 2013'. OWASP.

[30] tutorialspoint.com, 'SQL RDBMS Concepts', *www.tutorialspoint.com*. [Online]. Available: https://www.tutorialspoint.com/sql/sql-rdbms-concepts.htm. [Accessed: 19-Mar-2017].

[31] 'NoSQL Databases Explained', *MongoDB*. [Online]. Available: https://www.mongodb.com/nosql-explained. [Accessed: 19-Mar-2017].

[32] 'MongoDB at Scale', *MongoDB*. [Online]. Available: https://www.mongodb.com/mongodb-scale. [Accessed: 19-Mar-2017].

[33] A. Nayak, A. Poriya, and D. Poojary, 'Type of NOSQL Databases and its Comparison with Relational Databases', *Int. J. Appl. Inf. Syst.*, vol. 5, no. 4, Mar. 2013.

[34] 'Models | Django documentation | Django'. [Online]. Available: https://docs.djangoproject.com/en/1.10/topics/db/models/. [Accessed: 20-Mar-2017].

[35] 'SQLite vs MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems', *DigitalOcean*. [Online]. Available: https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems. [Accessed: 20-Mar-2017].

[36] 'Implementation Limits For SQLite'. [Online]. Available: https://www.sqlite.org/limits.html. [Accessed: 20-Mar-2017].

[37] 'NoSqlSupport – Django'. [Online]. Available: https://code.djangoproject.com/wiki/NoSqlSupport. [Accessed: 20-Mar-2017].

[38] N. Nurseitov, M. Paulson, R. Reynolds, and C. Izurieta, 'Comparison of JSON and XML Data Interchange Formats: A Case Study', Department of Computer Science Montana State University – Bozeman Bozeman, Montana, 59715, USA.
[39] A. Sumaray and S. K. Makki, 'A Comparison of Data Serialization Formats for Optimal Efficiency on a Mobile Platform', in *Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication*, New York, NY, USA, 2012, p. 48:1–48:6.
[40] 'REST vs XML-RPC vs SOAP'. [Online]. Available: http://effbot.org/zone/rest-vs-rpc.htm. [Accessed: 20-Mar-2017].
[41] 'REST vs XML-RPC vs SOAP – pros and cons : Max Ivak Personal Site'. .
[42] 'How REST replaced SOAP on the Web: What it means to you', *InfoQ*. [Online]. Available: https://www.infoq.com/articles/rest-soap. [Accessed: 20-Mar-2017].
[43] 'Understanding REST And RPC For HTTP APIs', *Smashing Magazine*, 20-Sep-2016. [Online]. Available: https://www.smashingmagazine.com/2016/09/understanding-rest-and-rpc-for-http-apis/. [Accessed: 20-Mar-2017].
[44] 'Do you really know why you prefer REST over RPC? | API Handyman'. [Online]. Available: /do-you-really-know-why-you-prefer-rest-over-rpc/. [Accessed: 22-Mar-2017].
[45] R. Golwalkar, 'Pros and cons in using JWT (JSON Web Tokens)', *Rahul Golwalkar*, 21-Aug-2016. .
[46] O.-M. K, M. K, M. S, M. A, and I. S, 'Effects of humid heat exposure on human sleep stages and body temperature.', *Sleep*, vol. 22, no. 6, pp. 767–773, Sep. 1999.
[47] H. J. Burgess, A. L. Holmes, and D. Dawson, 'The relationship between slow-wave activity, body temperature, and cardiac activity during nighttime sleep', *Sleep*, vol. 24, pp. 343–349, 2001.
[48] M. J. B. Rogers, K. Hrovat, K. McPherson, M. E. Moskowitz, and T. Reckart, 'Accelerometer Data Analysis and Presentation Techniques', Sep. 1997.
[49] D. Zemaityte, G. Varoneckas, and E. Sokolov, 'Heart rhythm control during sleep', *Psychophysiology*, vol. 21, no. 3, pp. 279–289, May 1984.
[50] E. Vanoli, P. B. Adamson, null Ba-Lin, G. D. Pinna, R. Lazzara, and W. C. Orr, 'Heart rate variability during specific sleep stages. A comparison of healthy subjects with patients after myocardial infarction', *Circulation*, vol. 91, no. 7, pp. 1918–1922, Apr. 1995.
[51] J. J. A. Moors, 'The Meaning of Kurtosis: Darlington Reexamined', *Am. Stat.*, vol. 40, no. 4, pp. 283–284, Nov. 1986.
[52] R. Edelberg, 'INDEPENDENCE OF GALVANIC SKIN RESPONSE AMPLITUDE AND SWEAT PRODUCTION', *J. Invest. Dermatol.*, vol. 42, pp. 443–448, Jun. 1964.
[53] W. Karlen, 'Adaptive wake and sleep detection for wearable systems', 2009.
[54] 'scikit-learn: machine learning in Python — scikit-learn 0.18.1 documentation'. [Online]. Available: http://scikit-learn.org/stable/. [Accessed: 22-Feb-2017].
[55] Apple, 'HomeKit Framework, Apple Developer'. [Online]. Available: https://developer.apple.com/reference/homekit. [Accessed: 21-Mar-2017].
[56] 'HomeKit Developer Guide, Apple Developer'. [Online]. Available: https://developer.apple.com/library/content/documentation/Networking Internet/Conceptual/HomeKitDeveloperGuide/Introduction/Introduction.html. [Accessed: 21-Mar-2017].
[57] OOPer Github, 'OOPer Github, 3rd ed'. [Online]. Available: https://github.com/ooper-shlab/HMCatalog-Swift3. [Accessed: 21-Mar-2017].
[58] SleepBot, *SleepBot - Sleep Cycle Alarm*. SleepBot, 2013.
[59] 'Sleep Cycle alarm clock on the App Store', *App Store*. [Online]. Available: https://itunes.apple.com/gb/app/sleep-cycle-alarm-clock/id320606217?mt=8. [Accessed: 01-Feb-2017].
[60] J. Gong and P. Tarasewich, 'Guidelines for handheld mobile device interface design', in *In Proceedings of the 2004 DSI Annual Meeting*, 2004.
[61] E. G. Nilsson, 'Design patterns for user interface for mobile applications', *Adv. Eng. Softw.*, vol. 40, no. 12, pp. 1318–1328, Dec. 2009.
[62] 'Alamofire Reference'. [Online]. Available: http://cocoadocs.org/docsets/Alamofire/4.3.0/. [Accessed: 21-Mar-2017].
[63] 'What is user authentication? - Definition from WhatIs.com', *SearchSecurity*. [Online]. Available: http://searchsecurity.techtarget.com/definition/user-authentication. [Accessed: 21-Mar-2017].
[64] 'Cross-Site Request Forgery (CSRF) Prevention Cheat Sheet - OWASP'. [Online]. Available: https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)_Prevention_Cheat_Sheet. [Accessed: 21-Mar-2017].
[65] B. Chen, P. Zavarsky, R. Ruhl, and D. Lindskog, 'A Study of the Effectiveness of CSRF Guard', in *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on*, 2011, pp. 1269–1272.
[66] C. Sharkie and A. Fisher, *Jump Start Responsive Web Design*, 1 edition. Collingwood, VIC, Australia: SitePoint, 2013.
[67] T. Firdaus, *Responsive Web Design by Example*. Birmingham: Packt Publishing, 2013.
[68] N. Jain, 'Review of different responsive CSS Front-End Frameworks', *J. Glob. Res. Comput. Sci.*, vol. 5, no. 11, pp. 5–10, 2015.
[69] 'What are the pros and cons of using Bootstrap in web development? - Quora'. [Online]. Available: https://www.quora.com/What-are-the-pros-and-cons-of-using-Bootstrap-in-web-development. [Accessed: 21-Mar-2017].
[70] D. Cochran, *Twitter Bootstrap Web Development How-To*. Birminghan, UK: Packt Publishing, 2012.
[71] A. Pratas, *Creating Flat Design Websites*. Birmingham, UK: Packt Publishing, 2014.
[72] J. Ling and P. van Schaik, 'The influence of font type and line length on visual search and information retrieval in web pages', *Int. J. Hum.-Comput. Stud.*, vol. 64, no. 5, pp. 395–404, May 2006.

To check for missed headings, delete at end

Table of Contents