DESE71003 – SENSING AND INTERNET OF THINGS

IMPERIAL COLLEGE LONDON

DYSON SCHOOL OF DESIGN ENGINEERING

Utilising Environment and Sleep Performance Data to Optimise User Sleep Regularity

Author: George Alexander (CID: 02039720)

Project Code and Data Repo URL:
https://github.com/g-alexander01/somna

Coursework Video URL:
https://www.youtube.com/@galexander_oceantech

Date: December 12, 2024

***Abstract*—This project aimed to address the pervasive challenge of irregular sleep patterns faced by university students, whose dynamic schedules often hinder maintaining a consistent sleep routine. A comprehensive IoT-based system was developed, integrating environmental sensors and sleep monitoring tools to analyse and optimise sleep conditions. The system uses a Raspberry Pi Zero 2 and sensor array to collect environmental data such as temperature, humidity, and air quality, while the use of a Garmin smartwatch captures sleep parameters, including sleep onset and wake times.**

These data sources are processed using Python-based algorithms to compute sleep metrics like Standard Deviation (StDev), Interdaily Stability (IS), and Social Jet Lag (SJL), providing personalised recommendations for optimal sleep and wake times. Environmental conditions are compared to established optimal values, with actionable interventions suggested for deviations. The outcomes are visualised on a dynamic web application built with Dash, offering users actionable insights and trend analysis. The project successfully demonstrates how real-time data-driven systems can support personalised sleep optimisation, with opportunities for scalability and further enhancements in efficiency and privacy.

Note: Tools such as ChatGPT have been used to support the rapid debugging and development of code, but these were always used in moderation given my motivation to use this opportunity to refine my programming skills.

I. Introduction and Objectives

University represents a pivotal stage in life, where students experience a new and exciting level of independence and responsibility. For many, this is the first time they are entirely responsible for their own schedule, balancing academics, socialising, exercise, commuting, working and everything in between. Whilst empowering, this autonomy comes with its challenges.

Maintaining a regular sleep schedule is essential for health and cognitive performance, yet university students often face challenges in maintaining it due to academic pressures, social events, and extracurricular commitments. Research shows that irregular sleep patterns are associated with poorer mental health, reduced cognitive functioning, and lower academic achievement [1] [2]. On the other hand, maintaining a consistent sleep schedule can enhance mood, memory consolidation, and physical recovery [3].

However, for students balancing late-night activities and early morning exercise, with schedules that vary week in week out, achieving a consistent sleep schedule can be challenging. This highlights the need for effective strategies that help students achieve the most regular sleep patterns that their dynamic lifestyles allow.

The aim of this project is to devise a new system that will enable students to optimise their sleep in real time, reacting to the discontinuities in their busy agendas and supporting them to practice sleep habits that are as 'regular' as they can realistically achieve. I also aim to use time-series sensor data to provide insight into the environmental trends within their sleep environment through the night, and to help them better understand how to foster an excellent sleep environment. The key objectives of the project are as follows:

- Measure and record bedroom environment indicators that have been shown to directly impact sleep quality and communicate these wirelessly to a database.
- Utilise my Garmin smartwatch to obtain data on my sleep, including precise sleep/wake times, time series heart rate data, sleep stage, and sleep restlessness.
- Research and implement approaches to use the aforementioned data sources to gain insight into my sleep patterns and approaches I can take to improve regularity.
- Develop an interactive web-app that delivers the actions I need to take each day to optimise my sleep schedule for that day, as well as display interactive plots that will help me better understand my sleep performance.

II. Section A: Sensing

The sensing functionality of this project would be achieved through 2 components: a Raspberry Pi Zero 2 (PZ2) and a Garmin wrist-based smartwatch (in this case the Fenix 6X Pro). The PZ2 was run with a 32-bit Raspberry Pi OS (pre. Raspbian), with all scripts written in Python for ease of development. SSH was used to connect to the PZ2 in headless mode during development, with the default username and password changed to mitigate security risk. It was configured to connect with my home WiFi network upon boot, allowing it to transfer files remotley. The PZ2 supports 2.4 GHz 802.11 b/g/n wireless LAN which performed brilliantly for the small files I was dealing with.

For simplicity, the sensing components are divided into 'Environment' which refers to the sensors within the bedroom collecting environmental metrics, and 'Sleep' which refers to the data obtained through the Garmin wrist-based smartwatch.

A. Sensors

The local bedroom environment data collection system consisted of the below components. I soldered header boards onto each component and they were quickly connected on a prototype breadboard, which provided a sufficient platform for the hardware.

I^2C was chosen as the communication protocol for the majority of the environment sensor array. This is due to its efficiency and suitability for integrating multiple devices with only two wires - an SDA bus for data, and SCL bus for clock. The shared bus architecture enables a streamlined, address-based communication between the Pi Zero 2 (I^2C master) and the multiple slave sensors. The PZ2's native I^2C support allows it to read data from the buses by addressing each sensor sequentially through its unique 7-bit address. The data structure consists of a start condition, slave address, read/write bit, data bytes, and stop condition.

The sampling interval of the SHT31 and ENS160 was set at 300s. Given the slow change seen in temperature and humidity trends in an indoor environment, this sampling frequency was selected to support the display of smooth time-series data without introducing excess transmission, storage and processing requirements.

- SHT31 I2C Temperature and Humidity sensor

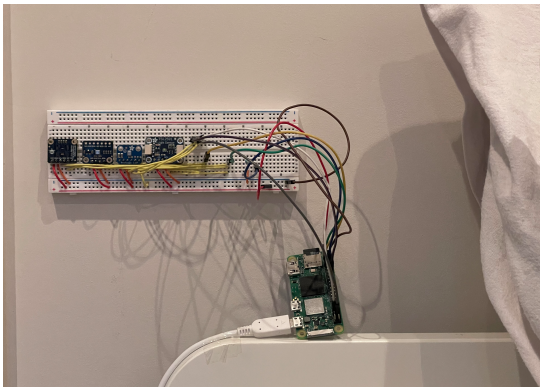

Fig. 1. Hardware System Setup by Bed

- ENS160 I2C eCO2 and TVOC sensor
- SPH06 I2S Microphone

1) Temperature and Humidity: The SHT31 is a highly versatile digital temperature and humidity sensor selected for its I^2C compatibility which simplifies integration with the PZ2 and avoids the need for an Analog to Digital Converter. The SHT31's high accuracy and resolution are particularly well-suited for monitoring bedroom environment conditions, where small changes in temperature or humidity can significantly affect sleep quality.

One drawback of the SHT31 is its sensitivity to condensation, which can temporarily disrupt readings. However, given the non-humid UK conditions and ventilation within the bedroom it was placed in this was not considered a risk. Future iterations of this project could explore more robust sensors for tropical environments to enhance its utility.

TABLE I
SPECIFICATIONS OF THE SHT31 SENSOR

Specification	**Value**
Temperature Measurement Range	-40 to 125°C
Temperature Accuracy	±0.2°C
Humidity Measurement Range	0% to 100% RH (at 25°C)
Humidity Accuracy	±2% RH
Operating Current and Voltage	<1.5mA @ 2.4 V

2) eCO2 and TVOC: The ENS160 is a digital air quality sensor specifically designed for monitoring eCO_2 (equivalent carbon dioxide) and TVOC (Total Volatile Organic Compounds). It relies on inbuilt correlations to infer CO2 from the TVOC readings, so may not be a highly accurate component, but for the purpose of this exploratory project and its budget it is a strong choice given low cost and I^2C compatibility. The sensor's <3-minute warm-up time allows it to provide accurate air quality readings quickly after being powered on, which is convenient for initiating measurements at the start of the night when the PZ2 is booted. The sensor is also efficient; with an operating current of 1.5 mA at 3.3V it supports sustained operation with low power demand.

3) Noise: The SPH0645-MEMS noise sensor provides a digital output via the I^2S (Inter-IC Sound) protocol. A range of 50 Hz to 15 kHz supports detection of a wide range of ambient sounds. However, a sensitivity of -26 dBFS and a signal-to-noise ratio of 64 dB means the the sensor did not adequately capture low-level background noises below a threshold of approximately 30 dBA. Through testing, it was found the sensor couldn't pick-up background noise from the road by my bedroom, which is often the source of disturbance. As such, the sensor was excluded from the project for now.

TABLE II
SPECIFICATIONS OF THE ENS160 SENSOR

Specification	**Value**
Warm-up Time	<3 minutes
Operating Temperature Range	-40 to 85°C
eCO_2 Measurement Range	400 ppm to 65,000 ppm
TVOC Measurement Range	0 ppb to 65,000 ppb
Operating Current and Voltage	1.5 mA @ 3.3V

TABLE III
SPECIFICATIONS OF THE SPH06 MICROPHONE

Specification	**Value**
Frequency Range	50 Hz – 15 kHz
Sensitivity	-26 dBFS
Signal-to-Noise Ratio (SNR)	64 dB
Sampling Rate	16 kHz
Operating Current and Voltage	<1.2 mA @ 3.3 V

4) User Sleep Data: The sleep data for the project was obtained through a wrist-based wearable device, my Garmin watch. An optical heart rate sensor measures blood flow under the skin via photoplethysmography and detects the changes in blood volume as the heart pumps, in turn using the time and variability between consecutive beats to calculate heart rate (HR) and heart rate variability (HRV).

The watch remains connected via Bluetooth to my phone, syncing periodically throughout the day, and in the event of an activity such as a recorded exercise or sleep finishing. Data is uploaded to the Garmin Connect app, which contains a host of insights on my health. The app uploads data to the Garmin Connect API, which can then be accessed to obtain detailed sleep data, including restlessness levels, sleep/wake times, and sleep scores. These can be correlated with environmental readings recorded by the PZ2 sensor array to provide holistic sleep insight. The watch uses its accelerometer to detect wrist movements during sleep. By analysing the frequency and intensity of these movements, it can determine sleep and wake times, as well as restless moments in the night.

Relating these readings to HR and HRV data enables the watch to assess sleep architecture; deep sleep is marked by minimal accelerometer activity, low heart rate, and increased HRV, reflecting parasympathetic dominance. REM sleep shows heightened accelerometer readings with variable heart rate. Light sleep features moderate movement and heart rate, acting as a transition between wakefulness and deeper stage [4].

B. Data Handling

1) Environment Data: The PZ2 was configured with a 32GB SD card for its file system and onboard data storage.

Environment sensor readings were uploaded in-real time to ThingSpeak, an IoT analytics platform that facilitates data storage and visualisation in the context of this project [5].

A private channel was created on ThingSpeak with a unique ID and write and read API keys. Each sensor variable was assigned to a field within the channel, with variables uploaded to their respective fields using a HTTP POST request. The 5 fields are within the channel limit of 8, and in the 2 weeks that the project has been running every night there has been approximatley 22,000 data entries, much below the channel limit of 10 million. The system also operates below the maximum ThingSpeak channel upload limit of one request per 15s, with readings uploaded once every 5 minutes.

Readings were also written to the PZ2 onboard file storage, providing a backup data source in the event of a network disconnection or PZ2 power outage. However, this was not observed in the 2 weeks of data collection, demonstrating the robustness of the system.

2) Sleep Data: Accessing the Garmin watch data was a little more tricky. Garmin restricts access to their API unless you are a registered developer, making direct data extraction challenging. To work around this, I utilised a GitHub module developed by Tom Goetz [6], which simplifies interaction with Garmin Connect by bypassing the official API.

The module supports the download of time-series sleep restlessness, HRV and HR. It also returns sleep scores and pre-programmed insights for a night given the value of HRV and total number of restless momements. Data is stored in JSON file format. A cron scheduler is run daily at 9 AM (a time before which I am always awake) and calls an update function that populates the database with the latest readings. The module also supports the download of activity files as well as daily, weekly and monthly summaries. In the future, or as additional scope to this project, I would be interested to explore using physiological stress metrics from the activities to further optimise sleep.

C. Basic Time-series Data

Environment and sleep data was plotted throughout testing and development to evaluate the performance of the system in relation to the objectives. MATLAB's integration with ThingSpeak supported the generation of interactive time-series graphs of the environment sensor data, and the GarminDB module was used with Jupyter Notebook to display sleep and other interesting health data.

1) Environment Data: The environment data obtained from the bedroom sensors was evaluated over nightly and weekly timescales. The weekly trend in temperature and humidity displays downward trend, attributed to outdoor weather changing toward colder and drier conditions from rainy conditions at the start of the week (given central heating schedule remained constant). These environmental changes can cause indoor conditions to deviate below optimal ranges for sleep, which are generally 15–19°C for temperature and 40–60% for humidity,

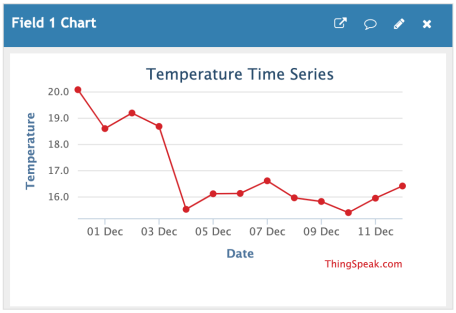

Fig. 2. Weekly Temperature Trend

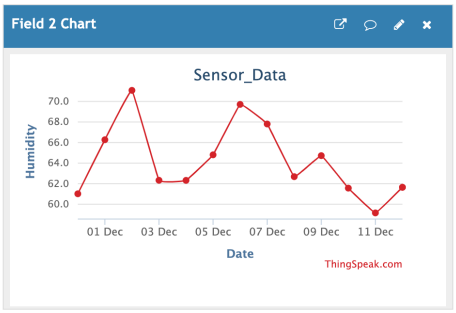

Fig. 3. Weekly Humidity Trend

especially as we enter the months of more extreme conditions [7]. As such, the final system will identify deviations between bedroom environment conditions and the optimal, and suggest methods to bridge the gap if required.

Given also the acute impact of TVOC and CO2 build-up in my (rather small) bedroom, it was crucial to understand the trend in these over the course of a night, and assess intervention opportunities. The below plots show notable spikes observed during the night, with initial theories blaming insufficient ventilation. However, given the magnitude and behaviour of the spikes, it was concluded the cause was a missed calibration step of the sensor. To rectify this, I followed the sensor's calibration procedure [8] and modified data collection script on the PZ2 to average readings over the course of 1 hour to smooth any recurring anomalies.

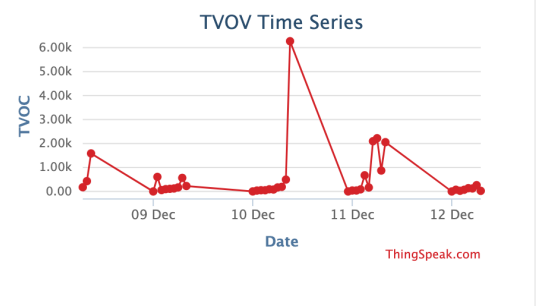

Fig. 4. Anomalous Readings in TVOC

2) Sleep and Health Data: Time-series data obtained from the GarminDB module was plotted using a graph class in Jupyter Notebooks. Argument of a date in datetime format is passed to generate the plots. Example plot of daily summary of heart rate, stress, and steps is shown in Figure 5.

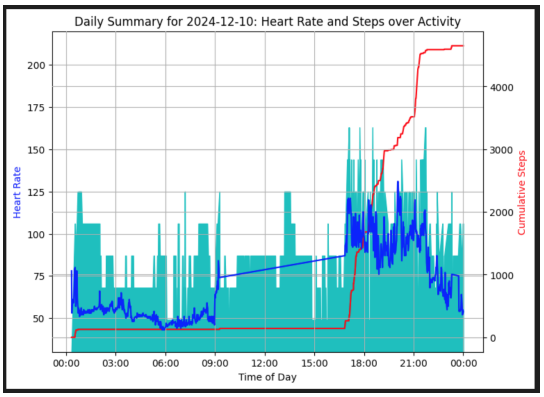

Fig. 5. Daily summary plot of Garmin data.

III. Section B: Internet of Things

The system developed is a web application designed to collect, process, analyse, and visualise sleep and environmental data. Built using Dash in Python with a Flask backend, the app incorporates HTML wrapper functions to deliver interactive features and visualizations and provides actionable insights into sleep regularity and environmental optimisation. The app is deployed on Heroku, with data processing scheduled daily at 9 AM.

A. Taking Insight from Data

As discussed in the introduction, the aim of this project is to develop a sleep tool that enables those with dynamic schedules to achieve a schedule of rest that adapts to their routine whilst supporting them to optimise their recovery.

A paper on measuring sleep regularity by D.Fischer, E.Klernman and A.Phillips provided excellent basis to achieve this [9]. The paper highlights the importance of selecting appropriate sleep regularity metrics for accurately assessing daily and multi-day sleep patterns, suggesting three metrics that can quantify and provide insight into an individuals sleep regularity. These metrics are Intra-Individual Standard Deviation (StDev), Interdaily Stability (IS), and Social Jet Lag (SJL).

1) Standard Deviation: In this project, StDev measures the variability in the sleep parameters of sleep onset time (bedtime), sleep offset time (wake) and duration. A higher standard deviation signifies greater variability across these metrics and therefore poorer sleep regularity. As such, the web-app intervention will aim to reduce StDev. It is calculated as shown in Equation 1, where X_i represents each individual data point (e.g., sleep onset, offset time or duration), $\overline{X}$ is the mean of all data points, and N is the total number of data points. The standard deviation provides a measure of variability, with lower values indicating more consistent sleep patterns and higher values reflecting greater variability.

$$\text{StDev} = \sqrt{\frac{\sum_{i=1}^{N} \left(X_i - \overline{X}\right)^2}{N-1}} \tag{1}$$

TABLE IV
StDev Function Arguments and Outputs

Specification	**Value**
Args.	date (str), callbackperiod = 7
Returns	stdevs (dict) = onset, offset, duration

2) Interdaily Stability: Interdaily Stability (IS) measures the degree to which an individual's sleep or activity pattern aligns with a consistent daily rhythm. A higher IS value (closer to 1) indicates greater regularity in sleep patterns, while lower values suggest irregular rhythms. IS is calculated using Equation 2, where N is the number of days, p is the number of epochs per day (24 for hourly resolution), $\overline{X}_h$ is the mean value at epoch h, $\overline{X}$ is the overall mean value, and X_i represents the individual sleep-wake data points. By evaluating the regularity in sleep across multiple days, the user can be provided with a quantified metric of how good their schedule is; whilst the absolute value is important, the user will be focussed toward their improvement in IS to either motivate or prompt them to use the system to improve regularity.

$$IS = \frac{N \sum_{h=1}^{p} \left(\overline{X}_h - \overline{X}\right)^2}{p \sum_{i=1}^{N} \left(X_i - \overline{X}\right)^2} \tag{2}$$

TABLE V
IS Function Arguments and Outputs

Specification	**Value**
Args.	binary$_s leep_d ate(list)$
Returns	IS (float)

3) Social Jet Lag: Social Jet Lag (SJL) quantifies the mismatch between the average mid-sleep timing on workdays and free days (e.g., weekends). Higher SJL values (positive or negative) indicate greater misalignment, which can negatively impact sleep quality and overall health. The metric is calculated using Equation 3, where MSF is the average mid-sleep time on free days (saturday and sunday), and MSW is the average mid-sleep time on workdays. In the web-app output, I attempt to be minimise SJL over an upcoming 3 day period by taking the sleep metrics over the previous 7 days and then determining the sleep and wake time that will reduce SJL by $\frac{1}{3}$ allowing it to gradually minimise over time.

$$\text{SJL} = \text{MSF} - \text{MSW} \tag{3}$$

B. System Design

The output is delivered through the fusion of sleep scores and environment scores. Each day at 9 am the cron scheduler

TABLE VI
SJL FUNCTION ARGUMENTS AND OUTPUTS

Specification	Value
Args.	sleep$_{m}idpoints(list)$
Returns	SJL (float) in hours

updates the Garmin connect database and calculates the average readings in temperature and humidity from the previous night. These values are passed to the `diff_to_ideal` function which determines the delta between the environment conditions and the ideal. Based off the magnitude of this delta, proportional actions are recommended to the user to enhance their sleep environment (under the consideration that environmental conditions between consecutive days are similar). This information is communicated visually on the web app, with a deviation below ideal temperature indicated with blue and above with orange, as well as deviation below ideal humidity as light blue and above with dark grey.

The optimal bedtime function is called with the cron scheduler also, with the current day and default callback period of seven days as argument. The output optimal sleep and wake times are profiled at the top of the web page to tell the user the bedtime and wake time they should set that day. These optimal times are calculated by determining the which sleep and wake time for the upcoming night will best balance an improvment in StDev, IS and SJL, scaled by an adjustment factor to prevent signifcant change in sleep schedule compared to the previous night. The implementation of this in Python is as follows:

1) Sleep Classification: The `sleep_scores.py` script takes the argument of a given date using the `datetime` module. The script extracts the first and last entries of the `restless_moments` dictionary inside the Garmin DB sleep data JSON file for that date, downloaded from Garmin Connect.

The start and end points of sleep are converted into datetime format and used to generate binary sleep data for that day, which serves as the basis for calculating Standard Deviation (StDev), Interdaily Stability (IS), and Social Jet Lag (SJL). The StDev function computes the variability of sleep onset and offset times, IS measures the regularity of sleep-wake patterns across days, and SJL quantifies the mismatch between workday and free-day mid-sleep timings.

The StDev, IS, and SJL functions within `sleep_scores.py` are invoked with a default callback period of 7 days. These calculated averages for the last 7 days are accessible within the `app.py` file and displayed to the user via interactive visualisations. These metrics provide a comprehensive view of the user's sleep regularity.

The average StDev, IS, and SJL values are passed into the `optimal_bedtime` function to compute an optimised bedtime and wake time. This function uses sleep patterns and target sleep duration of approximately 8 hours to align the user's schedule with circadian rhythm regularity while minimising disruptions from Social Jet Lag. The `optimal_bedtime` function works as follows:

- Input Parameters: Takes the current date and callback period as inputs.
- Calculate Historical Sleep Metrics: It calls the st_devs, IS, and SJL functions for the past 7 days to extract sleep and wake times, as well as average variability metrics.
- Adjust for Social Jet Lag (SJL): Based on the calculated SJL, the target sleep duration (default: 8 hours) is slightly modified if significant misalignment is detected.
- Optimal Sleep Timing: An ideal midpoint is calculated based on the average of sleep and wake times, adjusted for SJL, and further fine-tuned to gradually nudge the user toward regular patterns over a 3-day adjustment window.
- Gradual Alignment: Variability is reduced by incrementally adjusting towards ideal values, incorporating StDev for sleep onset and offset times to smooth transitions and improve overall sleep quality.

The outputs of the `optimal_bedtime` function include:

- Optimised Bedtime: Calculated as a gradual adjustment from the average sleep time toward the ideal schedule.
- Optimised Wake Time: Adjusted similarly to bedtime, ensuring consistent sleep duration.
- Optimised Sleep Duration: Ensures alignment between the adjusted bedtime and wake time.

The function dynamically balances current sleep habits with targeted interventions, leveraging StDev, IS, and SJL metrics to provide actionable recommendations. This forms the core feature of the web application, enabling users to improve their sleep patterns direct and simple instruction.

2) Environmental Delta: The average temperature and humidity values are passed into the `diff_to_ideal` function to compute their deviation from the ideal conditions of 19°C and 60%. This function categorises the deviations and generates actionable recommendations to improve the bedroom environment for optimal sleep conditions. The `diff_to_ideal` function works as follows:

- Deviation Calculation: The function calculates the difference between the nightly average temperature and humidity and the ideal values.
- Intervention Levels: Deviations are classified into three categories:
  - No intervention: Negligible deviation.
  - Moderate intervention: Slight deviation requiring minimal action (e.g., opening a window).
  - Significant intervention: Large deviation requiring immediate action (e.g., using a humidifier or heater).
- Recommendations: Based on the deviation levels, the `recommend_action` function provides tailored advice. For example:
  - High Temperature: "Try opening the bedroom window to cool things down."
  - Low Humidity: "Consider using a humidifier to balance the dryness."

IV. Evaluation

The IoT system resulting from this project achieves most of its objectives. Appropriately spec'd sensors and collection intervals provide a breath of data that support subsequent analysis and insight, and the communication over WiFi of the PZ2 was flawless. The Garmin smartwatch was effective in providing accuracte sleep and wake times, but could have been utilised to a larger extent by potentially supporting investigation into the relatioship between bedroom environment conditions and sleep quality - although that would not have been achievable within the project timescale.

The research process and implementation of detailed sleep metrics supported the development of a novel and adaptive sleep optimisation feature, that I have enjoyed benefitting from over the past few days. A futher development could be to employ an Apple 'shortcut' tied to the web-app that sets a bedtime and waketime alarm in my phone automatically.

The developed web-app contained personalised sleep optimisation support, delivered through data containers that displayed reactive styling, as well as interactive plots that showed the link between conditions from the last night and the resulting deviation between the average and absolute environment condition values. Some features of the web app are shown below.

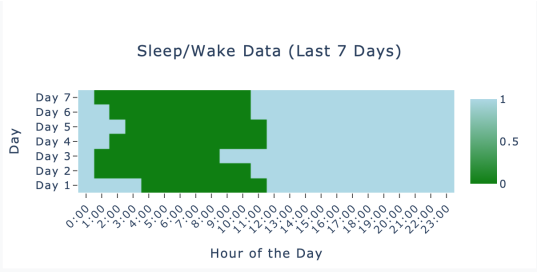

Fig. 6. Binary Sleep Heatmap Plot.

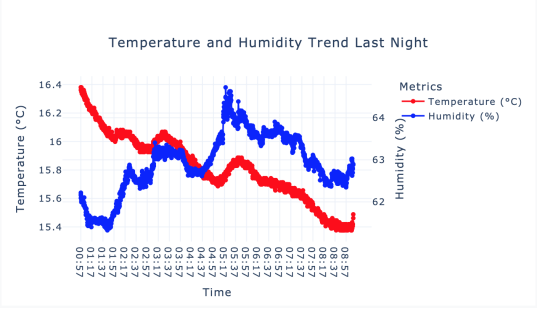

Fig. 7. Last Night Temperature and Humidity.

Fig. 8. Temperature and Humidity Deviation.

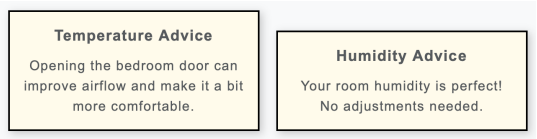

Fig. 9. Advice on Temperature and Humidity.

Fig. 10. Optimal Sleep and Wake Times.

Admittedly, as a busy university student myself, other projects and deliverables have restricted the extent to which I could make use of extensive data that can be obtained from Garmin Connect, as well as additional avenues of local environemnt data from an API such as

1) Personalisation: While the system effectively leverages user-specific environmental data to deliver tailored recommendations to improve sleep, its adaptive approach may be limited by the variety of data it collects. Tracking and learning deviations from optimal settings (19°C and 60% humidity) ensures practical interventions in temperature and humidity, but for some users, metrics of sound or light may be more important. As such, an improved system would contain a wider array of sensors, and potentially utilise deeper analysis methods such as regression analysis to determine holistic optimal conditions.

2) Scalability: The system is reasonably scalable in its current form. The sensor array is easily implemented, although could be designed onto a custom PCB for mass manufacture. Assuming users have access to a smartwatch, the web app could be updated to take input of the users login and password for their equivalent of Garmin Connect to access health data. However this approach raises security challenges.

3) Complexity: The system attempts to balance complexity and usability, but this balance may not be maintained as the system scales. While the modular function design reduces redundancy, it does not sufficiently address the risks of integration issues as new features are added.

4) Efficiency: Efficiency is achieved through a streamlined pipeline that processes nightly data each morning using a cron schedule to call `update_data` which downloads Garmin Connect data and environment sensor data. Interactive visualisations in `app.py` allow users to understand their environment and the actions to improve their sleep quickly. Transitioning to a database solution such as SQLite could significantly enhance both processing speed and reliability.

5) Security and Privacy: Although the project addresses basic security concerns by anonymising user data and removing sensitive information (e.g., IP addresses, authentication tokens) before publication, these measures are insufficient for robust privacy protection. Encrypting data in transit and at rest remains an omission at current. Furthermore, the absence of

user authentication creates a risk of unauthorised access to visualisations and insights.

6) Usability: The system was designed to run automatically as soon as the PZ2 was booted. This made it highly usable and easily included in my bedtime routine, however one improvement lies in modifying the sensor objects such that their LEDs are not enabled as they sometimes made falling asleep challenging - demonstrated in Figure 11.

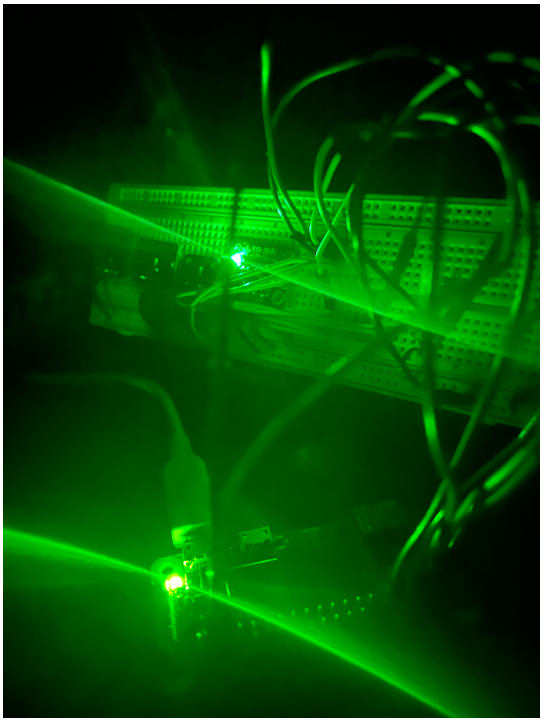

Fig. 11. Sensor LEDs.

V. Further Opportunities

Provided more time, I look to expand the data intake of the system to consider the exercise load I have experienced in a given day, supporting a more tailored sleep duration recommendation based off a recovery requirement. I also am interested to assess the performance of the system over time, as have only had 1 week to experience its fully functionality. Further scope may also lie in allowing users to change the optimal bedroom environment conditions dependant on the season, or perhaps adjust the proposed sleep duration to align better to their sleep preferences.

References

[1] B. Bei, J. F. Wiley, J. Trinder, and R. Manber, "Beyond the mean: a systematic review on the correlates of daily intraindividual variability of sleep/wake patterns," *Sleep Medicine Reviews*, vol. 28, pp. 108–124, 2016.

[2] D. Fischer, E. B. Klerman, and A. J. Phillips, "Measuring sleep regularity: theoretical properties and practical usage of existing metrics," *Sleep*, vol. 44, no. 10, p. zsab103, 2021.

[3] J. C. Lo, J. L. Ong, R. L. Leong, J. J. Gooley, and M. W. Chee, "Cognitive performance, sleepiness, and mood in partially sleep deprived adolescents: the need for sleep study," *Sleep*, vol. 39, no. 3, pp. 687–698, 2016.

[4] W. Zhao, X. Wu, and W. Xiao, "Sleep stage classification based on heart rate variability and cardiopulmonary coupling," in *Cyberspace Data and Intelligence, and Cyber-Living, Syndrome, and Health*. Springer, 2019, pp. 521–527.

[5] MathWorks, "Learn more about thingspeak," https://thingspeak.mathworks.com/pages/learn_more, accessed: 2024-12-12.

[6] T. Goetz, "Garmindb: A library for accessing garmin connect data," https://github.com/tcgoetz/GarminDB, accessed: 2024-12-12.

[7] Z. A. Caddick and K. Gregory, "A review of the environmental parameters necessary for an optimal sleep environment," *Fatigue: Biomedicine, Health & Behavior*, vol. 6, no. 1, pp. 1–14, 2018.

[8] Adafruit, "Ens160 component datasheet," https://cdn-learn.adafruit.com/downloads/pdf/adafruit-ens160-mox-gas-sensor.pdf, accessed: 2024-12-12.

[9] D. Fischer, E. B. Klerman, and A. J. K. Phillips, "Measuring sleep regularity: theoretical properties and practical usage of existing metrics," *Sleep*, pp. 1–16, 2021, advance Access Publication Date: 17 April 2021.