StrainLess

B7: Lilly Das, Cora Marcet, Kaitlyn Vitkin
18-500 Capstone Design, Spring 2025
Electrical and Computer Engineering Department
Carnegie Mellon University

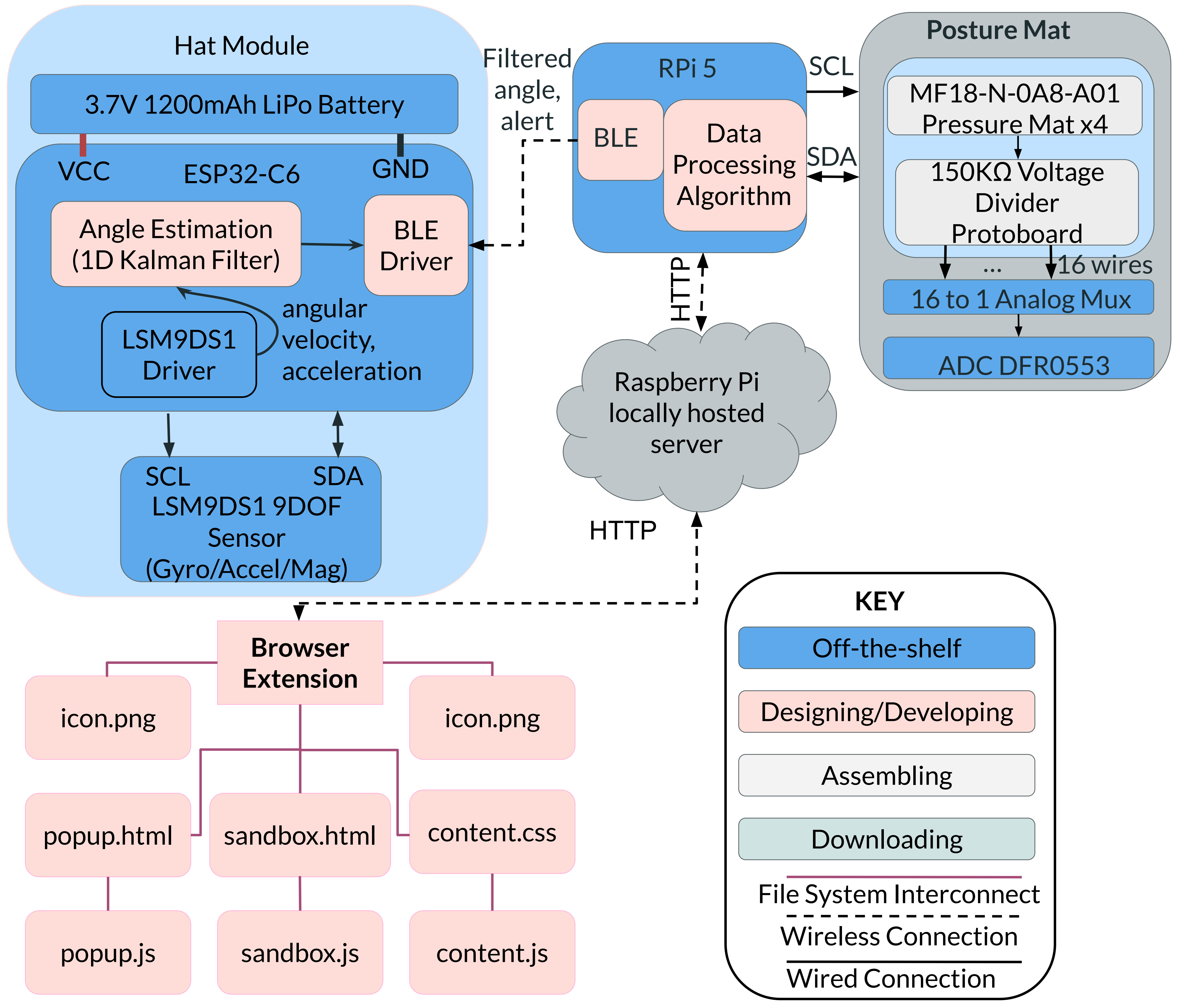
Product Pitch

StrainLess helps reduce the negative health effects of working at a desk for an extended period of time, specifically back and neck pain due to poor posture and eye strain from staring at a computer screen for long stretches.

StrainLess produces an alert via a browser extension when it finds the user is exhibiting poor posture and/or eyestrain. Those who frequently work at a desk including students, those who work at an office or from home, and gamers can therefore build healthier habits and have a better quality of life. Our project prioritizes the comfort of the user, with a goal of not impeding their work in any way or making them uncomfortable. Our tests ensure that StrainLess works for a variety of people, regardless of individual characteristics like weight and height. A notification will be sent only at appropriate times and in a timely manner so that it is actually effective in instilling better habits.

Our tests verify that our posture sensing is **over 95% accurate**, with a false positive rate of **3%** over 10 minutes. The neck angle tracking has an **RMS (root mean squared) error of <5°** over 10- and 30-minute tests. The latency (delay) between when our sensors find poor posture/eye strain to when a notification is received is **<1 minute**.

System Architecture



System Evaluation

Use-Case Requirements

Metric	Target	Actual
Alert Latency	< 1 minute	3.09 s (worst-case)
Neck Angle Accuracy	< 5° RMS Error	~3.5°
Lean Alert Accuracy	< 5% false positives	~3.5% false positives
Blink Rate Accuracy	<= 1 blink/min error	1 blink/min (avg. brightness)
Hat Weight	< 150g	115g
Seat Battery Life	> 8 hrs	> 25 hrs
Visor Battery Life	> 8 hrs	At least 8 hrs

Design Tradeoffs:
On the hardware side, the main tradeoffs included using a multiplexer for the sensor selection instead of directly wiring to multiple ADCs. This was done to decrease the runtime complexity of the python script on the RPi, in exchange for higher overhead time of switching channels. Another tradeoff lied in the choice of IMU location, which involved balancing user comfort, system weight, component cooling, and ease of layout. We ultimately decided on a visor over a neck band or cuff to improve weight distribution around the user's neck/head and decrease overheating. Software tradeoffs included the functionality of the browser extension vs. the security of the user, i.e. the extension having too much access to the user's system may make the user more vulnerable to attacks.

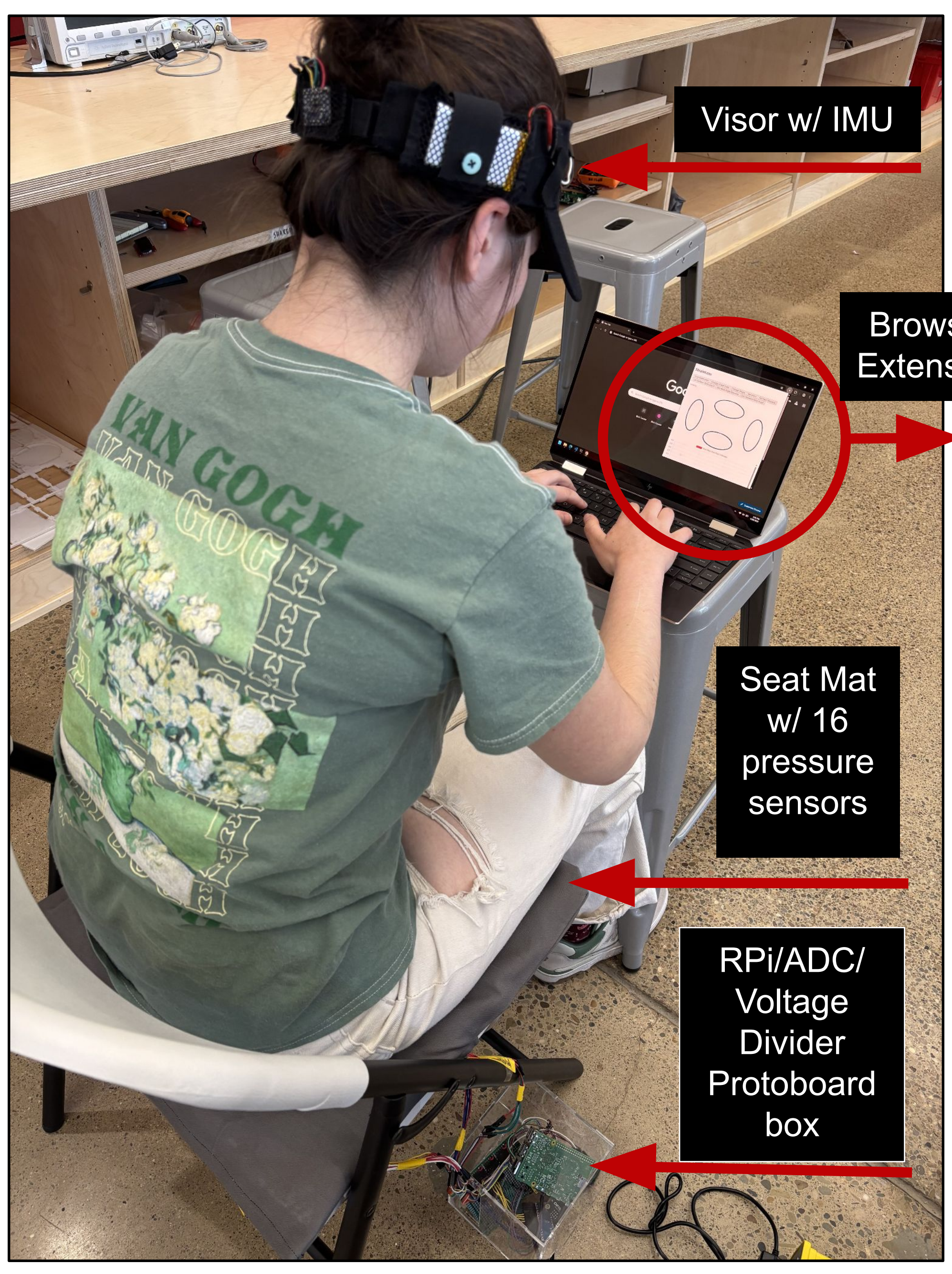
System Description

To calculate when a user's posture has deviated from their ideal, 16 pressure sensors' outputs are muxed, then passed through an 16-bit precision ADC after being sent through a 3.3V 150KΩ voltage divider. This data is passed through an I2C connection to an RPi5. The data is then averaged over the span of a minute, and a custom algorithm weights the sensor outputs to determine whether a deviation of over 5% from the baseline has been detected. This is sent to the user's display via HTTP requests to the extension.

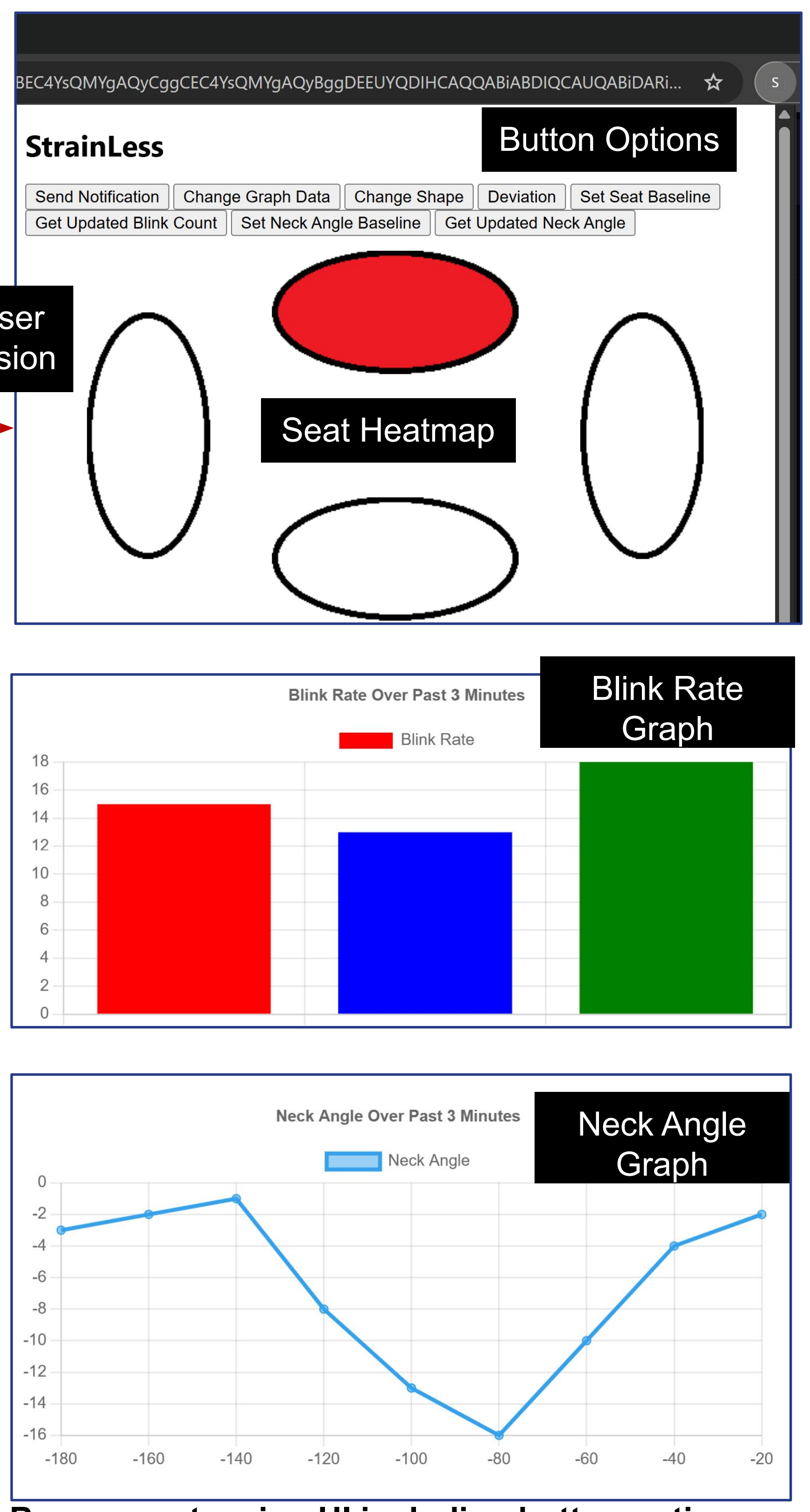
To calculate the user's neck angle, we have outfitted a visor with a 9-DOF IMU and an ESP32 to process the gyroscope and accelerometer readings through a 1-D Kalman filter. The filtered angle and the associated alert are sent to the user's display via a BLE connection to the RPi, and HTTP requests to the extension.

In order to detect eye strain, we are using CV (computer vision) to count the user's blinks per minute. Our CV algorithm is in Python and uses the OpenCV Python library.

The user can view their data in a Google Chrome browser extension which is built with JavaScript, HTML, and CSS. In the extension, they can also set their baseline (goal posture) by clicking a button. They can view the direction in which they are currently leaning through a heatmap of the seat which lights up red in the areas which they are leaning too heavily in relation to their baseline. They can view their blink rate and neck angle over the past three minutes via graph displays. The graphs are built with the chart.js npm module and are iframed into the HTML of the extension to ensure security with third-party code.


Visor w/ IMU
Browser Extension
Seat Mat w/ 16 pressure sensors
RPi/ADC/ Voltage Divider Protoboard box

Overall System: Browser Extension is connected to the RPi Server, which interacts with the hat module and the seat module



Browser extension UI including button options, seat heatmap, and blink rate and neck angle graphs.

Conclusions & Additional Information

Our final product fulfills our original goal of providing a posture/eye-tracking system that builds healthier habits for desk users. We met our use-case requirements along with our goal of providing well-timed and meaningful alerts to our users.
Lessons learned include the importance of early integration and how to fit algorithms for a diverse range of people vs a singular user. We learned a lot about hardware-software interaction, as well as how to collaborate to end up with a better result.
Possible extensions to our project include adding more sensors onto the back of the chair for additional posture tracking. In addition, we could continue working on creating a more robust algorithm to detect a wider variety of people, and creating a more professional-looking product by reducing the amount of sensors/wiring needed, which should further improve user comfort in the long run.

https://course.ece.cmu.edu/~ece500/projects/s25-teamb7/