Posture Watcher Final Report

Senior Design Spring 2021

By Phil Johnson, Sean McHale, Evan Kutney, & Gerald Kiliany

**Introduction 3**

**Background 4**

**System Requirements 6**

**Design Constraints 8**

**Summary of Design 10**

**Team and Timeline 15**

**Testing, Data, and Results 16**

**New Skills Acquired and Learning Strategies 21**

**Conclusions and Future Work 23**

**References 25**

**Appendix 27**

**Introduction**

Our Senior Design project was focused on helping users identify and correct incorrect sitting posture. Office workers routinely sit for hours a day, and sitting with incorrect posture can create many long term health conditions. Especially with the outbreak of Covid-19, more people than ever are working remotely, often in less than ergonomic chairs. Two key system requirements the chair satisfies are that it can correctly identify the quality of posture with which the user is sitting and alert the user if they are sitting with poor posture for an extended period of time. These two system requirements make up the main purpose behind the Posture Watcher and its functionality. Without these requirements our project would look drastically different than it does now.

The chair was constructed using a Raspberry Pi connected to an external power bank and our custom pcb board. The board would read inputs from eight separate embedded force sensitive resistors and convert these analog readings into digital information for the Pi, as well as connecting powering the two vibration motors. The main software was a python program written on the Raspberry Pi, a corresponding Android application, and a machine learning model deployed on the Google cloud platform. The Pi would be in charge of reading inputs from the sensors, sending classification requests to the machine learning model, interpreting the returned classification, vibrating the motors when appropriate, sending posture data to the mobile application, and updating settings that were changed in the mobile application.

We performed various tests with several different participants. Most of the tests consisted of having users sit in a way that they would characterize as “good posture,” “left leaning,” “right leaning,” “forward leaning,” and “back leaning” and comparing the classification from the algorithm to the posture the user is exhibiting. We found that the model accurately characterized the users posture roughly 90% of the time across all users, which we viewed as a success because these users were not necessarily used to train the model.

Our design requirements were updated over time as we made various important decisions about the Posture Watcher. The way our system requirements stand now, we have satisfied all of the requirements, which allows for a good user experience. However, there are some issues that are present in our final product. Because we did not want to destroy the chair we used in the process of implementing the project, the final design is a little unwieldy and bulky. However, for a prototype it does what it is designed to do, and if commercialization were to happen in the future, a more sleek, slimmed down design could be employed.

**Background**

Bad posture is a problem that may not be getting the attention it deserves in society today, as people may not know what having good posture really means. One definition of good posture is “distributing the force of gravity through [the] body so no one structure is overstressed” [1]. This tends to happen when a person is in an upright position, not sitting too far forward or reclined (see Figure 1 below). What tends to happen when a person is sitting is the head and neck move forward towards the object of attention (whether it be a phone, computer, or tv) which forces muscles to work harder to support the neck and upper body and stretches spinal ligaments [1]. This problem has been exacerbated by the coronavirus’s continuous spread, which has caused more people to work from home in sometimes sub-optimal sitting conditions. As many as 58% of Americans are working from home at least part of the time, and two-thirds of those working remotely would like to continue doing so even after the pandemic ends [2]. Choosing to work from a couch, sofa, or bed for eight hours a day every day for the foreseeable future could be detrimental in instilling bad habits for sitting anywhere else.

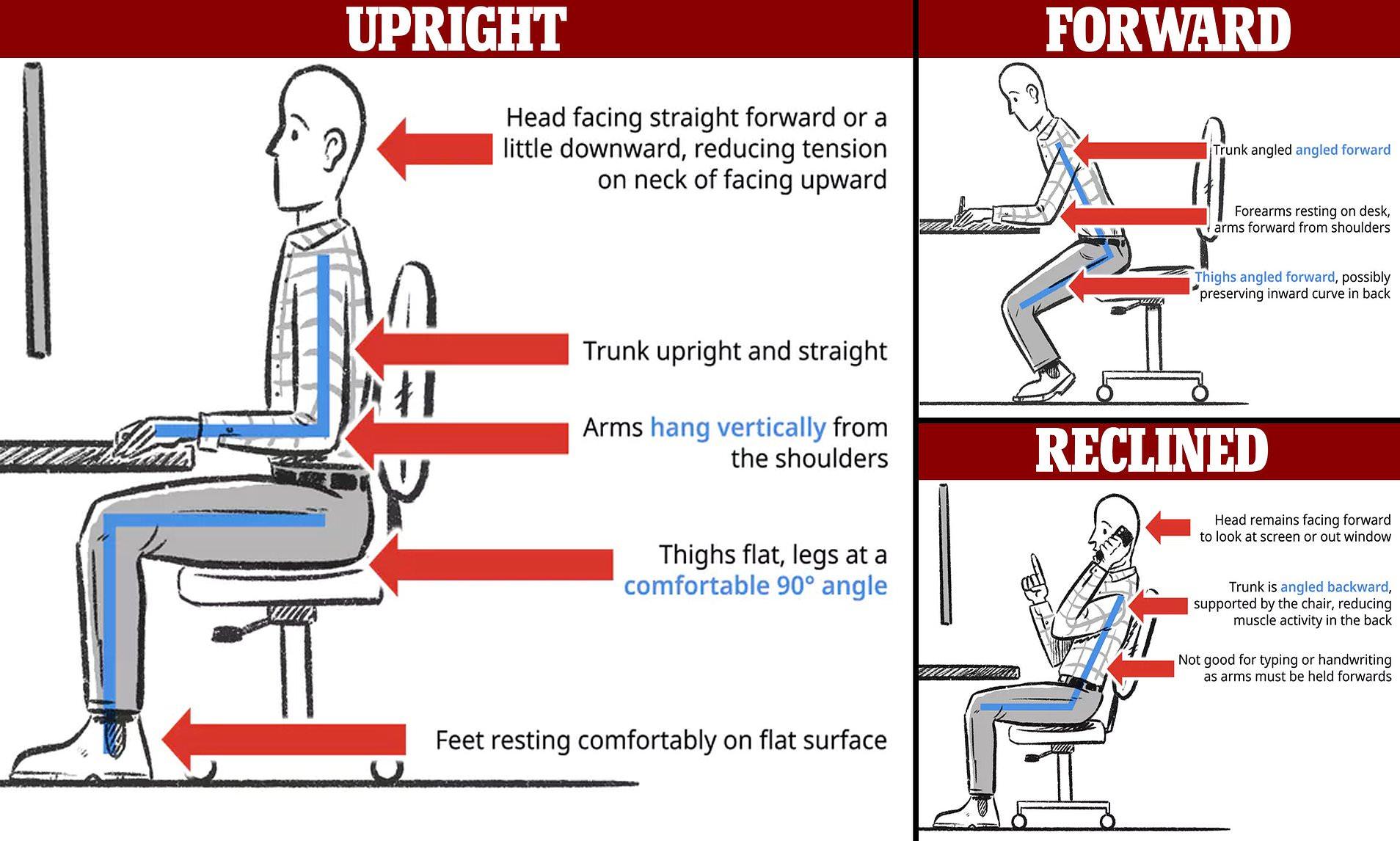


Figure 1: Sitting Posture Guide [3]

There are several reasons, both short-term and long-term, that sitting with bad posture is potentially harmful. Most obviously bad posture can cause back, neck, and shoulder pain, but it has also been found to cause spinal deformity, worsened lung performance, poor circulation, and digestive issues [4]. Putting aside the other health detriments, as many as 80% of Americans suffer from back pain alone at some point in their lives, and bad posture can be a major component of that [5].  
 Surprisingly, there are not many good technological solutions to this problem. Many chairs offer lumbar support in an attempt to correct a person’s posture while sitting in them, but effortlessly having good posture in one chair will not correct a person’s posture when they are sitting anywhere else. Therefore, an ideal solution would instill a good habit of having correct posture so that they will sit properly in any type of seat.  
 The current state of posture-correcting technology is severely lacking. Three distinct types of posture trainers currently exist: devices you wear, devices you adhere to yourself, and devices you sit on. None of these categories offer products that match the convenience, effectiveness, and accuracy of the proposed Posture Watcher.  
 The Lumo Lift is a device that is attached to the inside of the user’s shirt near their collarbone. The app that was needed in order for the posture trainer to function is no longer supported by Lumo BodyTech after it was acquired by another company, but even when the app was working the Lumo Lift still had its limitations. The technology requires a “snug fitting shirt that allows the sensor to stay close to your body,” so the user is restricted in what they can wear if they want the device to work at its best [6]. The Lumo Lift is also limited by its lack of sensors, one sensor around the collarbone cannot give the same advanced insight into a person’s posture that the multiple sensors of the Posture Watcher will provide.  
 The Upright GO Posture Trainer is a similar device that is attached to the upper back using a special adhesive. It has multiple sensors, but because it is only located in one spot on the user’s back it is still limited in the same way as the Lumo Lift in not being able to provide as much detailed sensing as the Posture Watcher will be able to. In addition, users may find sticking the device to their bare skin to be annoying and intrusive, and adhesives need to be repeatedly purchased with continued use of the product, meaning it is not a one-time purchase like the Posture Watcher will be [7].  
 Perhaps most similar to the proposed smart chair is the Darma Cushion, another “posture-training” device that the user sits on. While having the convenience advantage of not having to wear certain clothes or attach a device to one’s bare skin, the Darma Cushion falls short in its actual posture-tracking ability. The product is one cushion that goes on the top of a chair, so it is not able to track back or neck movements that could result in bad posture. The Posture Watcher will fix this problem by having sensors on the seat and the back of the chair in order to paint a more full picture of the user’s posture [8].  
 The potential downsides of our approach is that if it were made into a product available to customers, it would require either retrofitting an existing chair with our system or providing the user with an entirely new chair that includes our system. Retrofitting existing chairs would be difficult with our approach as a new posture detection algorithm would likely need to be used for a different chair, so the most reasonable approach would be to offer our system with an entirely new chair.

It is our goal to provide users with unparalleled posture-tracking technology without compromising on comfort. Once this is accomplished users will be able to work towards reversing the bad posture habits that the coronavirus pandemic has caused and prevent the potential chronic back pain that may result from them.

**System Requirements:**

## **Hardware Requirements**

### **Sensor Function**

The most fundamental requirement of the system is to determine whether or not a user is actually sitting in the chair as this is necessary before measuring the time sitting or user posture can be done. Using a sensor such as the force sensitive resistors will send a digital signal back to the microcontroller.

**Self-contained**

Part of the idea behind a “smart” product is ease of use. For this project, that would include the system being able to communicate with external devices and for the system to receive power without physical external connections.

**Battery Life**

The minimum battery life needed for the chair is ideally a full work day of 8 hours. The microcontroller is capable of drawing a max of 1 amp and powering all of the sensors will give a minimum capacity of 8 amp-hours. Aiming for multi-day use such as 20 amp-hours is desirable, and the microcontroller should not be pulling the max amount regardless, giving further battery life.

**Rechargeable Battery**

Giving the Posture Watcher a rechargeable battery will allow the user to not worry about the chair having extra cost after purchase as a product, along with allowing a high max capacity to work with. The rechargeable Li-ion battery design choice will allow charging over micro-USB or USB-C.

**Vibration Motor**

The chair is designed with two vibration motors on the armrests of the chair. They are used to notify the user when they have been sitting for too long or if they have been sitting with bad posture.

**Housing for Components**

The battery and microcontroller are components that will need to be protected and preferably out of plain sight during normal operation. The solution is to give these components housing to be added to the bottom of the chair, which will be drafted and 3D printed.

### **Team Designed PCB**

This is required of the system because it is a project for ECE 1896 Senior Design and all such projects must incorporate a PCB. If this were not the case, this prototype may be possible without fabricating a team created PCB. This will likely be used for the strain gauge pressure sensor.

## **Software Requirements**

### **Record and display user’s sitting statistics**

The chair will collect each user’s sitting data and use it to create a history of that user’s sitting habits. These will be viewable in the app, and will enable the user to see their progress sitting in the Posture Watcher over time.

### **Alert for sitting duration**

If the system determines that a user has been continuously sitting for a duration of time that is determined to be too long, it will notify the user via vibrations in the chair.

### **Alert for incorrect posture**

If the system detects that a user has been sitting with poor posture for a length of time that is determined to be too long, it will trigger the vibration motors to notify the user of this information.

### **Ability to modify settings**

The user will be able to change the settings regarding the frequency of the vibrations of the chair. Default settings will set the frequency of these until the user changes them from the mobile app. The user will be able to modify the bad posture threshold, consecutive sitting threshold, and inactivity threshold.

### **Ability to detect “good” and “bad” posture**

The algorithm must be correct in identifying whether or not the user is sitting with correct posture enough to provide accurate and useful information to the user.

### **Ability to function while disconnected**

In the event that no user is currently connected to the chair via the android app, the system will still be able to notify the user of posture or sitting alerts through vibrations in the chair.

## **Ability to classify the posture in real time**

Since the chair will be in charge of continually monitoring a person's posture, it must be able to classify the sitting position of its user as fast as it gathers such information. A backlogue cannot be allowed to form, as this would cripple the chair’s ability to identify and correct bad posture as it happens.

**Design Constraints:**

## **The product must still function as a chair**

### **The sensors in the chair do not impede the user experience**

Since the chair is designed for the user to use often over a long period of time, it must be comfortable to sit in. This project would not solve any problem if the users find the chair too uncomfortable to sit on long enough for the chair to monitor and report on their posture or if the sensors cause a user to sit in a bad posture position to avoid discomfort from the sensors. The sensors in the chair then must be minimally invasive as to not cause discomfort to the user.

**System must work with people of sizes ranging from the 5th percentile female through the 95th percentile male body dimensions**

These values are from 5’ tall to 6’ 1” and from 110 to 216 lbs. This is based on the business and institutional furniture manufacturers (BIFMA) standard for office chairs. As our product is intended to be used for working individuals sitting in office chairs, our product should follow this standard. [15]

**Limitations caused by this constraint:**

The sensors chosen had to be relatively flat so as not to create an unpleasant sitting experience for the user. Also, we chose an office chair that would fit the people in our size range, and needed to train the machine learning algorithm on data from people of different percentiles in our range to make it effective at classifying our target user.

## **The machine learning algorithm must be adequately trained**

**The algorithm will be trained on 5000+ data points from multiple users**

The proper training of the algorithm will allow for accurate classifications of posture using the data provided from the chair. Multiple users contributing data will help to create a robust model that is less biased to any particular user.

**Limitations caused by this constraint:**

A large amount of time must be devoted to gathering this training data, which could limit the time that is able to be spent implementing other aspects of the chair.

## **The chair goals must be achievable within the scope of this class**

### **The chair must be able to be produced for under $200**

Since we are only allocated a budget of $200 to work on this project, we must make sure its design and construction do not exceed this budget.

### **The chair must be completed by April 19th 2021**

Since the project must be graded and be ready for presentation at the senior design expo, total completion of the project must be done by the final due date.

### **The chair must be able to be completed by the four team members**

Since this is still a graded project, university policies regarding cheating and plagiarism still apply. While some of this project might be influenced by advisors and started from cited pre-existing frameworks, the majority of the work must be completed by the four team members who are tasked to work on it.

**Limitations caused by this constraint:**

The budget will limit the types of sensors and other parts we can afford to buy. Without top of the line sensors performance may not be optimal. The timetable could potentially rush some aspects of the chair and not allow for them to be fully completed by the April 19th deadline.

**Summary of Design:**

**Design Alternatives:**

There were several other designs the team considered before deciding on our current implementation. We looked into using a simpler posture detection software than machine learning, but decided that machine learning would give us the most accurate classifications and that was the priority. We originally wanted to implement two different alert modes that the user could choose from: app notifications and vibrations. In the interest of time we decided to only pursue the vibration motor implementation. The notifications would have been a nice feature for users to be able to choose from, but we felt that the vibration motors were more essential to our design. The biggest change in our product would have resulted from choosing to implement our project as a cushion instead of a chair. This would have had many benefits, such as portability and adaptability. Users would have been able to use it while driving, and carry it with them from chair to chair to gather information about their posture habits throughout an entire day. Ultimately, we determined that it was near impossible to gather accurate data from only sensors placed on the bottom cushion. Because we prioritized accuracy over most every other metric, we decided that although the cushion had many benefits, the chair design was going to give us a better chance to create a project that achieved the goals we have set for it.

Originally for the weight sensors, using strain gauges in a whetstone half-bridge configuration seemed like the best plan of action. Taking the voltage across the center and feeding the input back to the HX711 ADC/amplifier would allow precise measurements with a large range of values available. Upon testing the strain gauges proved quite fragile when not adhered and epoxied to a surface, causing them to tear. This led the team to switch to using FSRs in a voltage divider configuration instead, which proved more resilient, simpler to assemble, and when fed into the MCP3008 ADC the team used in the final design, allowed more inputs with less GPIO pins needed.

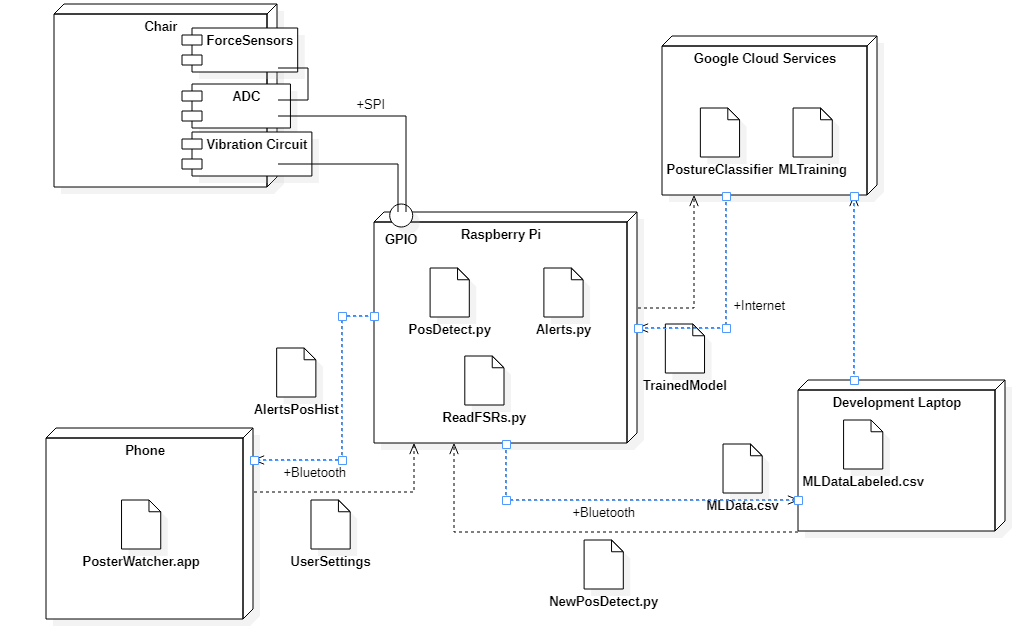


Figure 2: Deployment Diagram for posture watcher system

**Hardware Design:**

The chair has 8 force sensitive resistors (FSRs) attached to it. These are connected to PCB and each is set up to create a voltage divider with a 2.2kΩ resistor. The resistance of the FSRs start at a near infinite value, and decreases exponentially to a 1kΩ with a significant force on it (~40 lbs). The system runs the center of the voltage divider back the input pins of the MCP3008 ADC, which outputs a 10 bit number to the raspberry pi to process as the system input.



Figure 3: Smart Chair with all 8 FSRs attached

To alert the user, vibration motors are attached to the arms of the chair, with the final design including two small 3V motors. Both motors require a start-up current of 120mA, which is generated by using both of the 5V pins of the Pi, with a 18Ω resistance to drop the voltage, and a diode reversed and in parallel with the motor to prevent a kick-back voltage from ruining the motor over a long period of usage. 2N2222 transistors are used as switching transistors for the motor circuit, with both turning on when the gate pin is powered by the GPIO.

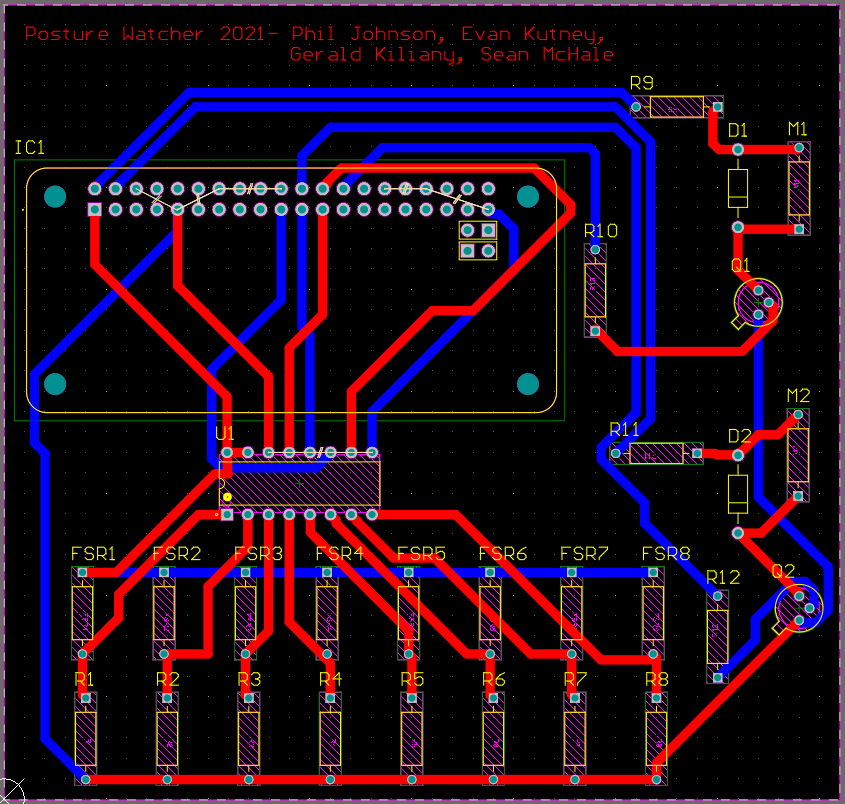


Figure 4: The PCB schematic, blue showing the bottom traces and red as the top traces

The power supply chosen was a 16 amp-hour power bank with a 2 amp USB-A output, chargeable by USB-C. The purpose of this was to allow the chair to run off of a single power source for extended periods of time without needing to recharge, and allow the user to charge it with a common cable. Both the power supply, PCB, and microcontroller were attached to the back of the chair using fitted 3D printed enclosures and 3M velcro strips. This allows the chair to maintain the ability to rotate, with all adjustment handles underneath unobscured.

. 

Figure 5: The back of the chair using both 3D printed enclosures.

**Software Design:**

The Raspberry Pi has two of our programs launched as part of its boot process. The first searches in the background for a bluetooth connection to the app if one is not yet established. The other is responsible for carrying out the rest of the system’s functions. It reads the sensor values from the ADC via SPI, it determines if there is a user seated, it sends and receives data from google cloud services to classify posture, it handles the timing for notifying the user of bad posture and prolonged sitting.

**Flow of Data**

When a user applies pressure to the FSRs by sitting in the chair, the voltage will change. The voltage values at these nodes are read through our ADC, the MCP3008, which will send these values to our raspberry pi through an SPI connection. The values that are read are used to first determine if a user is seated in the chair at the moment. If that is the case, then the sensor values are sent to the posture classifier algorithm that we trained using and deployed on google cloud services. What is then received through this wifi connection, is one of five posture classifications, being sitting upright (good posture) or leaning forward, back, left or right (bad posture) and a confidence percentage for this classification. This classification is written to a file which can be sent via Bluetooth serial connection to and the app we created on the user’s phone. In the app the user can view graphs generated with this data to show the amount of time they spend in good posture or bad posture and see how much time they spent in each of the five categories of posture. The app is also where the user can change several settings for the system. One setting is the amount of time before the vibration motors on the chair will notify the user that they have been sitting for too long. Likewise, the user can change the amount of time spent seated with bad posture that will notify them in the same way. They can also change how much time can go by with nobody sitting in the chair before the system shuts down to save power.

**Posture Detection**

Throughout the class over fifteen thousand data points were gathered to train the various iterations of machine learning models. For our final model, 6758 data points were collected from eight college age individuals, seven of which were males. Not all of the volunteers offered their body dimensions but we had a range of weights from 110 to 240 pounds which enables our system to operate within the BIFMA office chair constraint 1.1.2. The final model was trained to minimize log loss with 6052 of the data points dedicated to training and the remaining 706 used for model testing. Based on these 706 test points, 695 were correctly classified giving a theoretical precision of 98.6%. The model was then deployed off the Google Cloud Platform to enable the chair to make online posture classifications based on its current sensor inputs.

**Team and Timeline:**

Posture Watcher’s ambitious goals required a variety of skill sets The team’s Hardware Design specialist and business admin was Phil Johnson. His responsibilities included circuit and PCB design and testing, researching and ordering parts, organizational work, and prototype assembly. Phil did a lot of troubleshooting involved with the PCB-microcontroller interfacing, along with designing the enclosures and prototyping them.

Sean McHale was responsible for sensor construction, testing and placement. After ensuring all sensors were reporting accurate data consistently, he worked on multiple phases of volunteer posture data collection, cleaned and labeled the gathered data, trained several machine learning models, and implemented the online deployment of the model for real time chair posture classification.

Evan Kutney was primarily responsible for the Bluetooth connection between android app and raspberry pi, and the development of the app itself. He worked with Gerald to refine outputs from the pi and determine how to best display the data in the app.

Gerald Kiliany was responsible for the Raspberry Pi programming. This included: reading sensor values by creating an SPI connection to the ADC, detecting if a user is seated using these values, system timing for alerts, generating alerts by powering the vibration circuit through the Pi GPIO, reading inputs from Bluetooth with the help of Evan, writing sensor values to a csv file for ML training, writing and sending posture information to the phone app, and power save settings.

**Timeline**:

The major milestones that were completed in order to realize the final prototype were:

* Week 1: Strain Gauge Sensor testing
* Week 2-3: Designing and assembling the initial prototype on a breadboard.
* Week 4: Force Sensitive Resistor testing
* Week 5: Seating detection working with values read from the sensors.
* Week 6: Bluetooth connection to device
* Week 7: PCB assembled with software running
* Week 8: Volunteer data collection and model training
* Week 9: Connecting deployed model to chair for classifications
* Week 10: Improved Sensor construction and regathering of data
* Week 11: Retraining of machine learning model and final model deployment
* Week 12: Final chair testing and housing construction

**Testing, Data Analysis and Results:**

## **Software Tests**

The chair will require a few different software tests, especially given the team plans to implement a companion app to monitor and input info from the user.

**TEST-1.0**

The system shall determine if the user’s posture is good or bad above 80% of the time.

**RESULTS**

Over the tests that we ran with 500 samples of posture, 100 being good posture and 400 being bad posture, 488 of these samples were correctly categorized into good vs bad posture giving an accuracy of 97.6%. The graphs showing all test results can be found in the results from test 1.1 and the appendix.

**DATA ANALYSIS**

The results of our testing show an accuracy of well over 80% for determining if posture is good or bad. Since these tests were done over a relatively small period of time in a laboratory setting, they cannot be completely generalized to users over a long period of time in a work environment. Despite this, our results should translate well based on the assumption that an average user does not change positions many times in a short period of time. With the accuracy in the lab being so high, there is strong evidence that the real-world accuracy will be high enough that a user sitting with poor posture for a prolonged time will be accurately detected and notified with our system. This allows them to be aware of their poor posture and correct it, solving the poor posture problem that we are focused on.

### **TEST-1.1**

User’s posture position should be correctly identified above 80% of the time for each position considered.

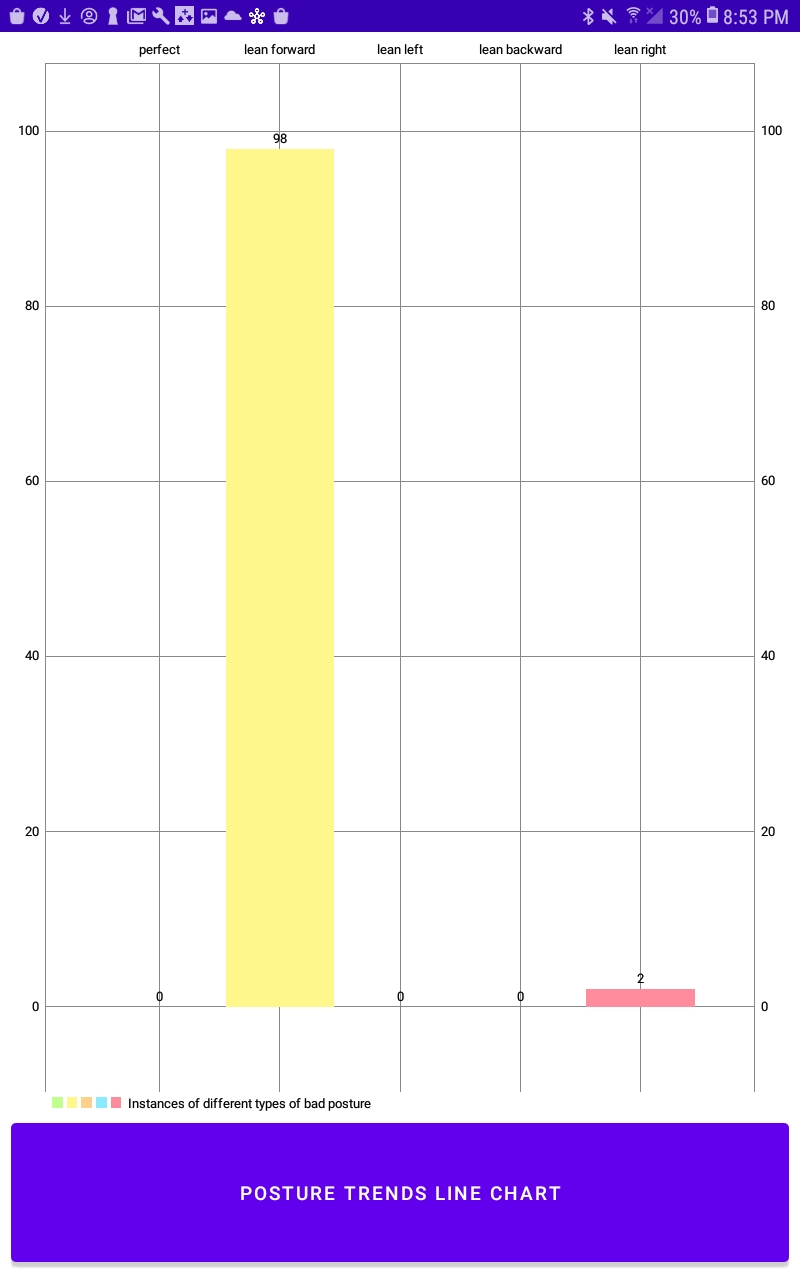


Figure 6: Graph showing results of leaning forward test which was classified 98% correctly (graph generated in the app)

**RESULTS**

The remaining graphs showing how the posture was classified for 100 samples taken while a participant was sitting in each position can be found in the appendix. Each test had the following percentage of correctly classified samples. Upright/ good posture: 88%, Leaning forward: 98%, Leaning backward: 100%, Leaning right: 100%, Leaning left: 94%.

**DATA ANALYSIS**

The results from our tests are very promising. In each test our classifier worked with above 80% accuracy. It is unclear if these correctness values would perfectly translate to a random user of the system in a real world work environment over a long period of time. Testing under these conditions was not possible due to several factors but most importantly time and personnel restrictions due to the COVID pandemic. Despite this, with the accuracy of results gathered in lab testing, it will be very unlikely that a user having bad posture over time would not be notified of this by the system. This evidence supports it will work as intended to solve the problem of bad posture.

### **TEST-1.2**

User will be alerted by a vibration after the set amount of time spent sitting consecutively is reached.

**RESULTS**

User is successfully alerted by the vibrations on the chair at the specified consecutive sitting time.

### **TEST-1.3**

User will be alerted by a vibration after the threshold for consecutive bad posture is reached

**RESULTS**

The user is successfully alerted by vibration motors attached to the chair when the bad posture threshold is reached.

**TEST-1.4**

Chair successfully connects to the user device via bluetooth.

**RESULTS**

Upon booting, the chair successfully pairs to the user device. When booted unpaired, will successfully pair and update user setting accordingly.

### **TEST-1.5**

Chair is able to function normally (without a bluetooth connection) from startup with no external intervention.

**RESULTS**

The chair successfully starts up and begins notifying the user via vibrations when the default consecutive bad posture and consecutive time sitting thresholds are met.

**TEST-1.6**

The user will be able to change alert timing settings while the system is running and they will be updated.

**RESULTS**

The user can successfully update the alert timing settings and changes will be reflected in the system within 15 seconds.

**TEST-1.7**

Chair successfully connects to the google cloud model.

**RESULTS**

Upon booting, the chair successfully connects to the cloud hosted model which can be seen by the classification label and confidence returned to the chair when it makes a request.

**TEST-1.8**

The machine learning model was successfully trained with over 80% of testing data being correctly classified

**RESULTS**

Based on 706 randomly chosen test points that were separated from the training data, 695 were correctly classified giving a model precision of 98.6%. These results are visualized in the table below.

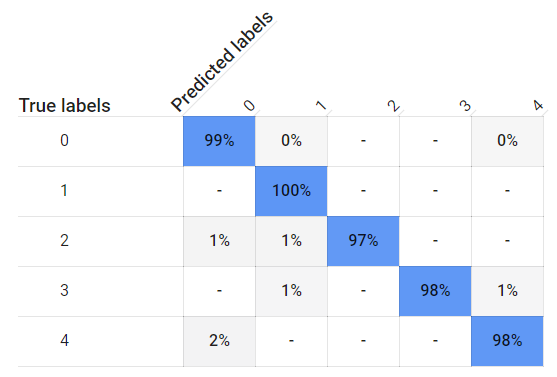


Figure 7: Theoretical calculated accuracy of our posture classification from google cloud services. The labels are as follows: Upright/Good posture: 0, Leaning forward: 1, Leaning left: 2, Leaning back 3, Leaning right 4

**DATA ANALYSIS**

The machine learning model was able to not only identify good posture 99% of the time based on testing, but it also correctly identities the four other incorrect posture positions at least 97% of the time based on the testing portion of the training data. This is indicative that we gathered enough data to create an accurate posture sensing model, and that the training of said model was done correctly.

**TEST-1.9**

The posture can be correctly classified for a user who was not part of the training data.

**RESULTS**

We collected 100 sample classifications from a volunteer who was not part of the original data set, of which 94 were correctly classified.

## **Hardware Tests**

### **TEST-2.1**

Users sit in the chair in different positions and check which sensors are going

off. May require test variations of the software.

**RESULTS**

We were able to have all sensor inputs print out in real time and had all 8 sensors successfully display a 0 when no force was applied and increasing numbers as we pressed on them with increasing force.

### **TEST-2.2**

System is able to be started and shut down with no major issues.

**RESULTS**

The system starts up and powers down with no issues.

### **TEST-2.4**

Random Users will be asked to sit in the chair improperly, and upon setting off the vibration motor, surveyed on whether vibrations from the motor are possibly too harsh, or not noticeable enough.

**RESULTS**

All eight people we tested noticed the vibrations when they were active on the chair.

**New Skills Acquired and Learning Strategies:**

**Gerald**

One learning strategy I started using during this course was to consider multiple possible approaches for achieving the same solution to evaluate which is best and then incorporate it into the final product. I utilized this strategy when I was going through the process of learning how to run programs and commands as part of the boot process for the Raspberry Pi. The purpose of this was to enable the system to be used in our without any display or other external hardware. This involved seeing several common approaches to solve this problem online and creating implementations of them early in the development process [9]. At this point it was unclear which approach would be best. However, as the complexity of the system grew it became clear that certain implementations would not work. The simplest to implement had issues as certain commands, such as those that require root privilege, could not be run at this stage of the boot process. Other methods had issues with starting the bluetooth connection. The approach that was finally decided on solved both these issues and also allowed running the program in a terminal on startup so for hardware debugging we could simply connect to a display and see the output values right away.

**Sean**

One of the most complicated and significant new technical skills I developed over the course of this project was how to create and manage machine learning models using Google Cloud Platform. I had some experience with machine learning, but had never had to create an entire project end to end before. To accomplish this I devised a strategy of scanning through what I would need to do for the entire process, and then focusing on the specifics as I implemented each part. I feel that by taking a shallow dive of the entire process, I could get a grasp for the general scope of what creating the machine learning algorithm would entail and help me understand the general concepts of how it works. Then after establishing a roadmap of what I thought I needed to do, I could focus in greater detail on the current step I was on, allowing me to make sure I did not make any errors in my implementation.

I first watched some video tutorials, which were less than helpful, and then moved on to read google’s provided documentation [13], which was actually helpful. I found that unlike many other programs I had used before, there was very little outside resources to help me troubleshoot the errors I encountered and I regularly found that Google’s own documentation was the most helpful guide. I did on several occasions run into errors with google’s method that were not touched upon in the documentation, such as when I was trying to enable the chair to receive predictions from the model. To solve this, I opened the module’s source code [14] and read through the code and comments to understand how it worked and what was causing the error. As it turned out I needed to export the credentials to the model as a json file, and include this as an argument when calling for online predictions.

After I had read up on what to do, I was successfully able to navigate Google autoML. I was able to create a new project, enable billing, apply for a free trial to gain credits, create a storage bucket in the appropriate region, and import some test data. Because I took the time to read through the process and understand the general concepts along each part of my roadmap, I was able to have the data properly formatted and labeled before importing it, which I am sure helped avoid unnecessary debugging in the later steps in the process. Continuing with the actual training of the model, I was able to successfully train six models throughout the project, and ended up deploying two of them. Accessing the deployed model from the chair did prove difficult as previously mentioned, but because of my strategy I was daily confident that the deployment up to that point had been correct, which led me to believe that it was the way I was calling the model that was causing the error, which was ultimately correct. Overall, I feel that my strategy helped me understand and correctly implement an entire machine learning model despite being unfamiliar with how to do so in a project as advanced as this.

**Evan**

This project required the use of a lot of new technologies that I had not experienced before and needed to learn before I could implement my part. Prior to this semester, I had no experience with android app development, Android Studio, connecting to a Bluetooth Serial Port, and very limited experience working with Raspberry Pis at all. Thankfully I was already proficient in Java, which I was able to use as the coding language for everything involved in the app, but using frames, activities, and other android concepts was completely new to me and took a lot of learning to be able to integrate it into the project.

The main strategy that I employed in order to learn these things I was unfamiliar with was gathering as much information as I could before blindly attempting to “jump in” and start coding in a naive fashion. I used a developer.android guide geared towards creating a first android app [10], as well as an instructables tutorial [11] with a similar purpose to walk through and create an unrelated app to gain knowledge for creating the real one. After finishing this, I gathered information specifically about taking information that users had entered from various other guides in the android developer website [10]. During this process, I was researching different methods to connect my app to the Raspberry Pi via Bluetooth on several raspberrypi.org[12] forums of people who had similar goals to my own. There were many helpful posts about both Bluetooth Low Energy (BLE) and Bluetooth Serial Port connections, but I eventually went with the Bluetooth Serial Port method after seeing advantages in performance and wealth of information compared to BLE.

Once gathered, I was able to apply this knowledge to my portion of the project with varying degrees of success. The Bluetooth connection, while a little finicky at times, was able to be instantiated with the tips and methods I had learned from my research. For the most part the app was able to be implemented successfully as well, with one caveat: I ran out of time towards the end and was not able to get the android notifications working correctly. The reason for this was not necessarily that the information I had gathered was not sufficient, but rather that I had spent a little too much time gathering information and simply ran out towards the end. The majority of the sensors coming in late and the resulting lack of knowledge about how the data would look coming from the chair did not help, but I should have taken steps to make sure notifications would be working in time. However, as far as the rest of the app, everything worked as intended and I believe overall my strategy was a success. It can be intimidating to begin to use a new technology, so the more background information a person can learn beforehand the better.

**Phil**

As the team’s singular EE, I felt a lot of pressure on my circuit design skills and prototyping skills. As I learned, what works well on paper does not alway correlate well to reality. The strain gauges originally intended did not translate well to prototyping, which required going back to researching another method. I had never needed to use an ADC and had to thoroughly research that out before ordering any. PCB design was an area I was familiar with but this project was a big step up in terms of scale. I also decided that getting it manufactured by SERC would be the fastest route, since delay in parts shipping led to me refusing to commit to my PCB design til it was completely tested out. Circuit design with transistors is an area I was very unfamiliar with, as we talk about it a lot but rarely actually design with them. That required a lot of datasheet digging, and overall a lot of improving my non-linear circuit design.

The biggest skill I got to work on this semester was leadership, as the remote environment throughout the semester required a lot of tracking down people for meetings, being flexible when things went wrong or people were missing before a deadline, and adapting to the circumstances. My team and I spent a lot of time together in the design lab, and we found that working together in person was our best method of holding one another accountable in terms of progress, as we often needed to work ahead of what the very next step was a group in order to get the project progressing as it needed to.

**Conclusions and Future Work:**

In conclusion, the Posture Watcher worked as designed and achieved the goals we had set for it. It classifies types of posture well, and alerts the user in the case of bad posture or sitting too long. The Bluetooth Serial connection works well as a system of transferring data to the android app, which displays user data nicely. Additionally, we finished with $37.88 of our budget remaining. Full details of our budget can be found in the appendix. On a less positive note, there were several things we learned that could have been implemented in a better way. The sensors, which were secured to the chair via duct tape, were not fastened in place as well as we had thought they should be. They came loose on a couple different occasions, resulting in us having to retrain the model to be able to obtain accurate results. The wires that run from the sensors to the back of the chair take up a lot of space and get tangled easily. The vibration motors are secured to the arms, but they can sometimes be too small to be felt or noticed, and the vibration does not travel well from the arms to the rest of the chair. The chair we chose was intentionally not able to lean back, as we feared this would mess up our data. This sacrificed user comfort for accuracy, which we were willing to do in this instance, but was not ideal. On the software side, notifications not being implemented hurts the functionality of the chair a lot, as some users would most definitely prefer notifications to vibration. In addition, while the ways that the data is currently analyzed paint a clear picture of the user’s posture improvement/regression over time, there is much more room for more data analysis in the form of other graphs and data displayed. As much as we feel the Posture Watcher accomplishes what it set out to do, there are definitely some improvements that can be made.

If we were to start from scratch and create a new Posture Watcher with more of a mind to commercialize the product, among the first of the changes to be made would be embedding the wires, sensors, PCB, etc inside of the chair. This would create a more appealing look to the user, and also help to hold the sensors in place better. The new placement for the vibration motors could be in either the back or bottom of the chair, and we would upgrade to stronger motors in our new version so that the vibration is always noticed. To address the chair not being able to lean back without invalidating our obtained data, we would add a gyroscope to our implementation and machine learning algorithm so that we could track how far back the user is leaning and more accurately determine posture while allowing for maximum user comfort. Notifications would definitely be implemented in our new version, along with other user friendly graphs displaying more relevant data. We had the idea to potentially allow users to “compete” against their friends to see who can sit with the most consistent good posture, so that feature may have made it in in a commercial implementation. Overall, there are definitely changes we would make if we were to build the Poster Watcher again, but we do not see the need for major overhauls as we feel we have a solid base for a useful, innovative product.

**References:**

[1] D’Ambrosio, Frank J. “The Importance of Good Posture.” Southern California Orthopedic Institute, https://www.scoi.com/services/physical-therapy/importance-good-posture.

[2] Brenan, Megan. “COVID-19 and Remote Work: An Update.” Gallup, 13 Oct. 2020, https://news.gallup.com/poll/321800/covid-remote-work-update.aspx.

[3] All My Family Care. https://allmyfamilycare.com/health-news/physiotherapist-reveals-exactly-how-you-should-sit-on-your-chair/

[4] “5 Long-Term Complications of Poor Posture.” Corner Chiropractic Center, 23 Jul. 2020, https://www.cornerchiropractic.com/5-long-term-complications-of-poor-posture. 18

[5] Steussy, Lauren. “Experts say your posture is terrible and wrecking your health.” New York Post, 24 Sept. 2018, https://nypost.com/2018/09/24/experts-say-your-posture-is-terrible-and-wrecking-your-health/.

[6] “Lumo Lift: The First Wearable Posture Coach. You slouch, it vibrates! A posture corrector that’s perfect for sitting or working at computers. Comfortable & easy to use. Improve your posture today!” Amazon, 5 Feb. 2021, https://www.amazon.com/Lumo-Lift-corrector-computers-Comfortable/dp/B00N9P8GMW.

[7] “Upright GO 2 NEW Posture Trainer and Corrector for Back Strapless, Discreet and Easy to Use Complete with App and Training Plan Back Health Benefits and Confidence Builder.” Amazon, 5 Feb. 2021, https://www.amazon.com/Upright-GO-Corrector-Strapless-Confidence/dp/B07SRW2D38.

[8] Raphael, Rina. “I Tried Five High-Tech Solutions That Promised To Fix My Terrible Posture.” Fast Company, 12 Jul. 2017, https://www.fastcompany.com/40438857/i-tried-five-high-tech-solutions-that-promised-to-fix-m y-terrible-posture-did-they-work.

[9] Dexter Industries, “Five Ways To Run a Program On Your Raspberry Pi At Startup” <https://www.dexterindustries.com/howto/run-a-program-on-your-raspberry-pi-at-startup/>

[10] “Build your first app.” Android Developers. <https://developer.android.com/training/basics/firstapp>

[11] TheZachBales. “How to Create an Android App with Android Studio.” Instructables.

https://www.instructables.com/How-To-Create-An-Android-App-With-Android-Studio/

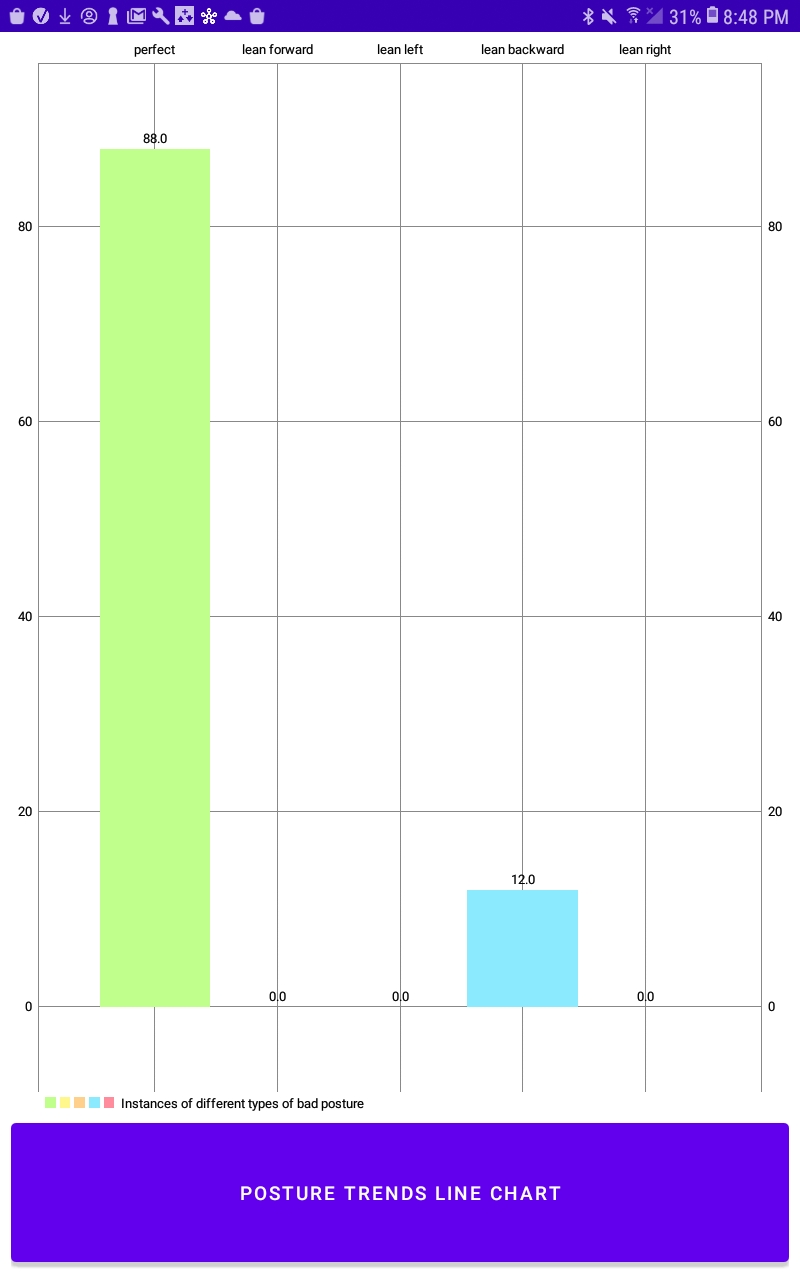
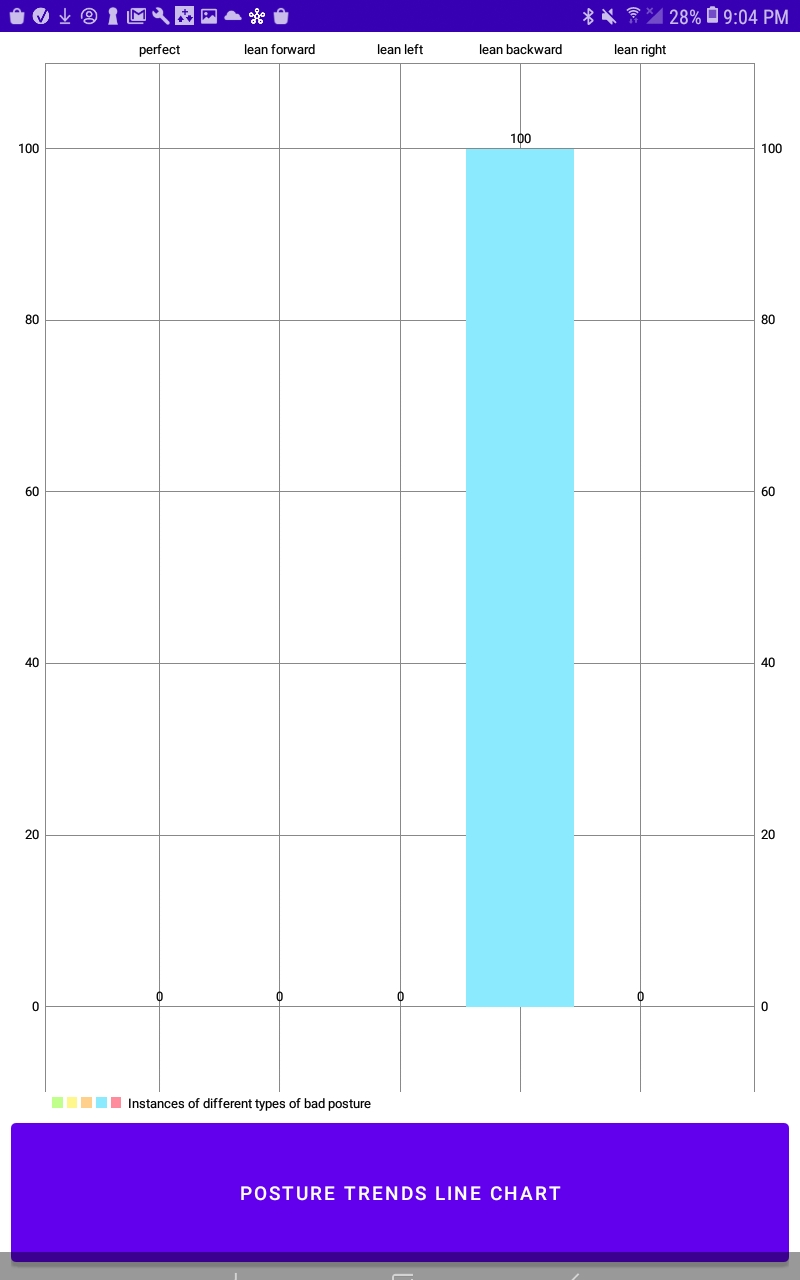
[12] Forum Posts. Raspberry Pi. <https://www.raspberrypi.org/forums/>

[13] Google Cloud AutoML Documentation. <https://cloud.google.com/automl-tables/docs>

[14] Source code for Google Cloud module <https://googleapis.dev/python/automl/latest/_modules/google/cloud/automl_v1beta1/tables/tables_client.html>

[15] BIFMA Product Conformance Requirements <https://cdn.ymaws.com/www.bifma.org/resource/resmgr/compliance/bifma_pc-2020.pdf>

**Appendix**

Figure 8 - Lean Backward accuracy 100% Figure 9 - Perfect Posture Accuracy 88%

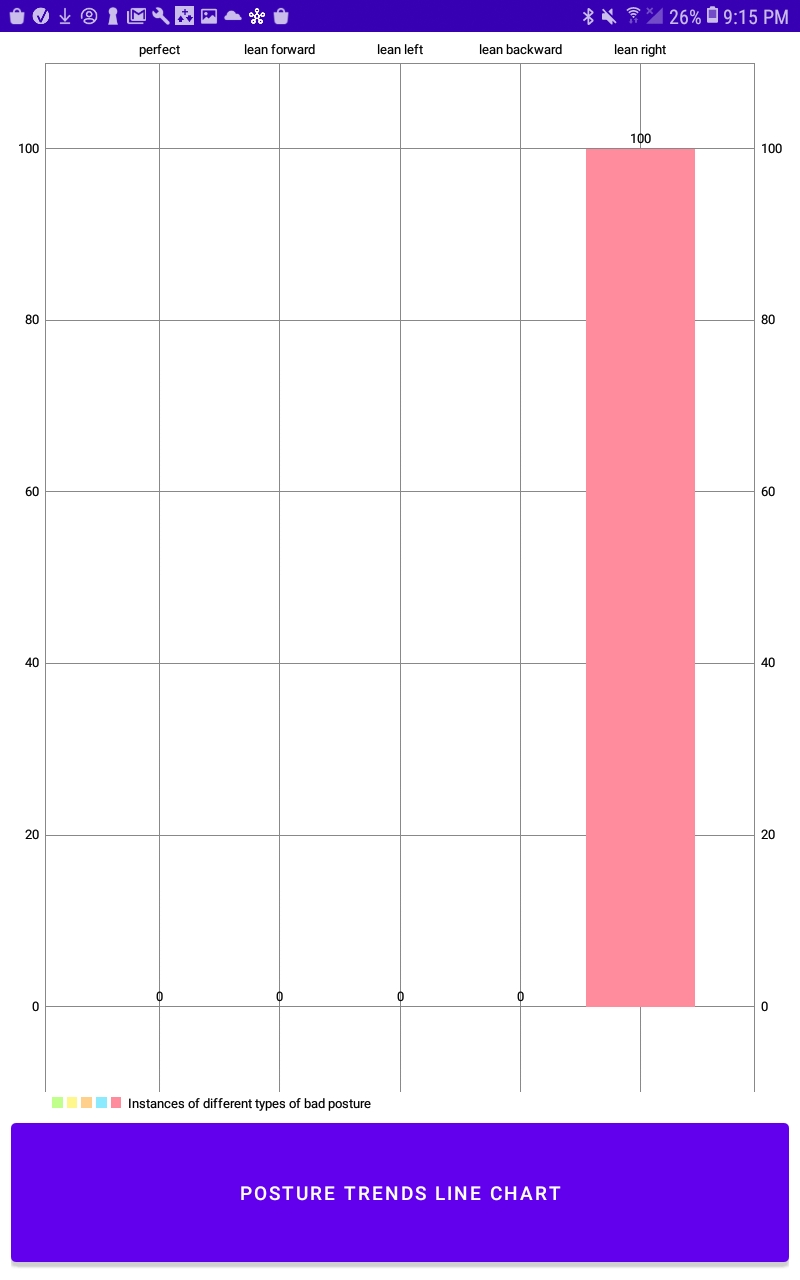
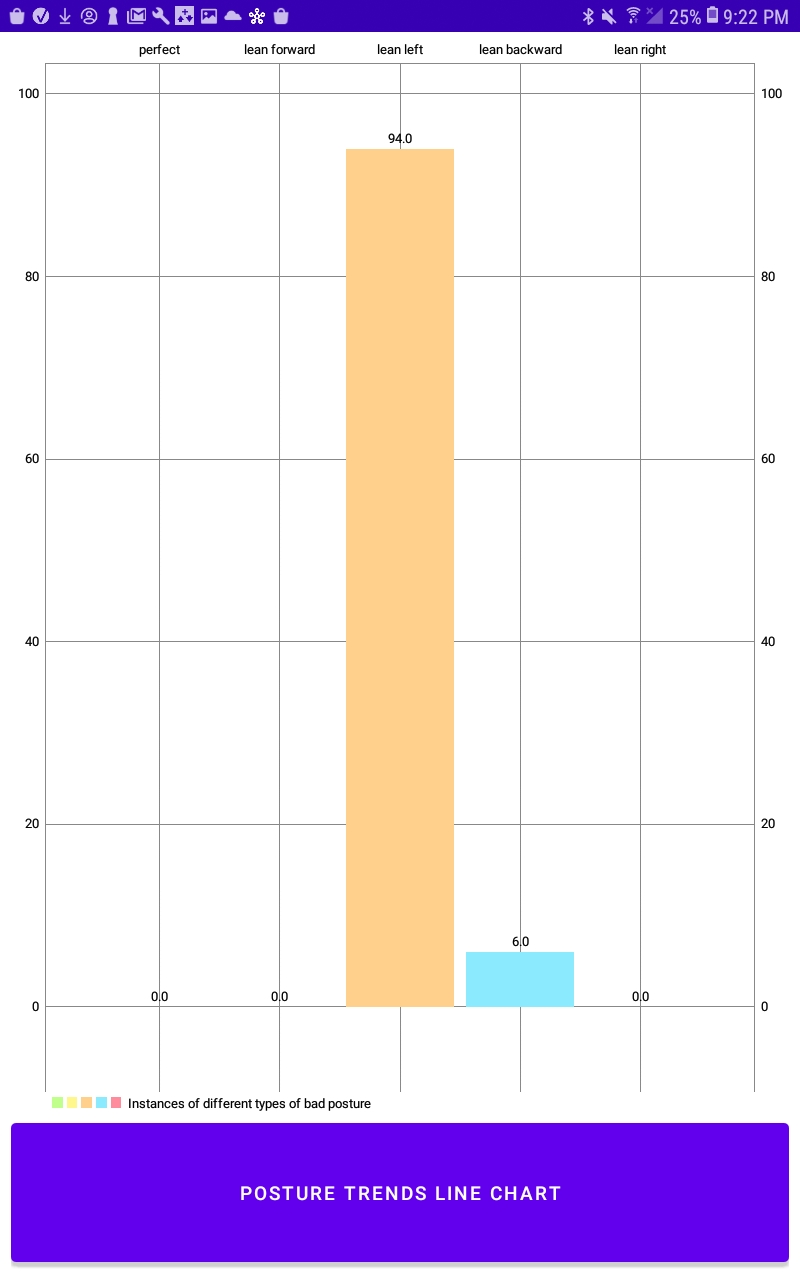
****

Figure 10 - Lean Right accuracy 100% Figure 11 - Lean Left accuracy 94%

| Vendor | Purpose | Order Cost | Total Budget Remaining |
| --- | --- | --- | --- |
|  | Initial Budget | 0 | 200 |
| Amazon | Strain Gauge/ADC | 17.98 | 180.02 |
| Adafruit | Microcontroller  +Components | 45.45 | 136.57 |
| Interlink | Initial FSRs | 27.95 | 108.62 |
| Digikey | Parts/Motors | 20.83 | 87.79 |
| Interlink | Final FSR Order | 49.91 | 37.88 |

Table 1: Budget for our project, we finished the project with $37.88 remaining of our initial $200