The primary driver of the device is two Arduino nano 33 BLE micro controllers. The ecg signal is captured by a MAX86150 board. The MAX has both PPG and ECG capabilities.
The PPG is done using a single finger but the output signal is very shaky, the ECG signal uses the leads which looks a lot better. While the leads are labeled for right arm, left arm, and right leg they do not have to be placed on those exact spots to work, you can put them on your torso or just back and it will give very similar outputs. The pads are disposable and do not last longer than a couple days once opened, if you try to use them after a while from opening it then it will kill the output's accuracy. In addition, the right leg/ground electrode is not needed to get a signal but it reduces the noise of the signal substantially.

The two Arduinos are separated into a transmitter and receiver. The transmitter does no calculations in order to minimize power usage. The receiver is able to be plugged into a wall or a usb port to get power while in use. The two devices use bluetooth low energy to connect to each other, it uses very little power and gives a range of 10-15 feet at max.

<u>The housing impacts the distance that the bluetooth signal can travel significantly. Down from 10-15 feet to around 5 feet maximum.</u>

- Travis DiDonato

When designing the ecg, we needed a series of filters to remove any distortion of signal. Our first thought was to use a series of analog filters. These included a notch filter to remove the common 60Hz frequency, and a bandpass filter to attenuate frequencies outside of the 0.5-150 Hz range which is the industry standard.. However, the added passive active components to the overall design would make the device less practical and create more room for error. To simplify this problem we decided to simply go with an FIR and lowpass filter. The purpose of the lowpass filter was to remove high frequency noise by any outside interference such as body movement or breathing. The purpose of the FIR filter was to pass signals that are 0.5-150Hz and further amplify them so that they can be observable.

A FIR was presigned in the example arduino code that came with the device, we simply repurposed it for our design. However, you can use the website down below to design filters from scratch.
http://t-filter.engineerjs.com/

We designed the digital low pass filter through a trial and error process when finding the coefficients. However, a more mathematical process can be done by using the reverse euler method.

https://math.libretexts.org/Bookshelves/Differential_Equations/Numerically_Solving_Ordinary_Differential_Equations_(Brorson)/01%3A_Chapters/1.03%3A_Backward_Euler_method#:~:text=The%20backward%20Euler%20method%20is,%2B1)%3Dyn.

- William Friend

When designing the form-factor of the device, an issue that had to be addressed was the compatibility between multiple hospital ECG devices. Since every manufacturer of ECG devices has different cables/ports, a cable converter would simply not be feasible.

A solution to the problem was to connect the hospital's ECG leads directly to the Wireless ECG Receiver by a means of wiring the conductive, metal electrode buttons to the Wireless ECG Receiver's Arduino Nano BLE 33.
When receiving data from the Wireless ECG Transmitter, it will send the processed signals (FIR and Low-Pass filters) to the hospital monitor.

One issue that has come to attention is how the hospital ECG interprets the signal. It currently reads a relatively steady PQRS wave from the patient, but there exists noise such that the monitor cannot interpret a heart rate from the signal.
A possible cause is either from the MAX86150 board's library not supporting differentiating (more than one) signals expected from most ECG devices.
Another possible cause is due to the hospital ECG expecting a true analog signal and the signal being sent from the Wireless ECG Receiver is Digital-to-Analog.

- Krister Lawlor